Generated Documentation
Generated using Doxygen v 1.8.13

Prepared by
Colan Biemer,
Allison Frauenpreis,
Gabrielle Getz,
Jasmine Marcial,
Laura Mo,
Shreya Patel

Advisor
Jeff Salvage

Stakeholder
Frank Lee
Highwater Generated Documentation

Purpose
This 820-page document specifies the relations between each class (a total of 220 classes), and the detailed descriptions of each method and property. It allows testers and Highwater reviewers to look up each class and its details and can support the Design Document and Testing Acceptance Plan.

Description
The team of developers were able to generate this thorough document because the team followed the C# coding guidelines (CSharpStyleGuide.pdf also submitted in case the specified link below does not work):

https://goo.gl/ucYsb9

Each submitted code change had to be reviewed through a Github pull request by at least two other team members before submitting, and commented code was expected – allowing us to generate such a thorough document.
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.3.2.1</td>
<td>AnnouncementYAMLParser()</td>
<td>23</td>
</tr>
<tr>
<td>4.3.3</td>
<td>Member Function Documentation</td>
<td>24</td>
</tr>
<tr>
<td>4.3.3.1</td>
<td>LoadAnnouncements()</td>
<td>24</td>
</tr>
<tr>
<td>4.4</td>
<td>Attribute Class Reference</td>
<td>24</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Detailed Description</td>
<td>25</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>25</td>
</tr>
<tr>
<td>4.4.2.1</td>
<td>Attribute()</td>
<td>25</td>
</tr>
<tr>
<td>4.4.3</td>
<td>Member Function Documentation</td>
<td>25</td>
</tr>
<tr>
<td>4.4.3.1</td>
<td>GetDuplicate()</td>
<td>25</td>
</tr>
<tr>
<td>4.4.4</td>
<td>Property Documentation</td>
<td>25</td>
</tr>
<tr>
<td>4.4.4.1</td>
<td>Name</td>
<td>25</td>
</tr>
<tr>
<td>4.4.4.2</td>
<td>Value</td>
<td>26</td>
</tr>
<tr>
<td>4.5</td>
<td>Barometer Class Reference</td>
<td>26</td>
</tr>
<tr>
<td>4.5.1</td>
<td>Detailed Description</td>
<td>28</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Member Function Documentation</td>
<td>28</td>
</tr>
<tr>
<td>4.5.2.1</td>
<td>Equip()</td>
<td>28</td>
</tr>
<tr>
<td>4.5.2.2</td>
<td>SetUpTool()</td>
<td>28</td>
</tr>
<tr>
<td>4.5.2.3</td>
<td>Unequip()</td>
<td>28</td>
</tr>
<tr>
<td>4.5.2.4</td>
<td>Use()</td>
<td>29</td>
</tr>
<tr>
<td>4.5.3</td>
<td>Property Documentation</td>
<td>29</td>
</tr>
<tr>
<td>4.5.3.1</td>
<td>CheckBarometer</td>
<td>29</td>
</tr>
<tr>
<td>4.6</td>
<td>BaseItem Class Reference</td>
<td>29</td>
</tr>
<tr>
<td>4.6.1</td>
<td>Detailed Description</td>
<td>33</td>
</tr>
<tr>
<td>4.6.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>33</td>
</tr>
<tr>
<td>4.6.2.1</td>
<td>BaseItem() [1/3]</td>
<td>33</td>
</tr>
<tr>
<td>4.6.2.2</td>
<td>BaseItem() [2/3]</td>
<td>33</td>
</tr>
<tr>
<td>4.6.2.3</td>
<td>BaseItem() [3/3]</td>
<td>34</td>
</tr>
<tr>
<td>4.6.3</td>
<td>Member Function Documentation</td>
<td>34</td>
</tr>
<tr>
<td>4.6.3.1</td>
<td>AddItemCategory()</td>
<td>34</td>
</tr>
<tr>
<td>4.6.3.2</td>
<td>ChangeName()</td>
<td>34</td>
</tr>
</tbody>
</table>
4.6.3.3 Discard() ............................................ 35  
4.6.3.4 FireItemNameChangedEvent() .......................... 35  
4.6.3.5 FireItemSpriteChangedEvent() .......................... 35  
4.6.3.6 GetItemAttribute() ...................................... 35  
4.6.3.7 GetItemAttributes() ..................................... 36  
4.6.3.8 GetItemCategories() ..................................... 36  
4.6.3.9 GetItemCategoryByClass() .............................. 36  
4.6.3.10 GetItemCategoryByParentClass() ......................... 37  
4.6.3.11 GetItemToModify() ...................................... 37  
4.6.3.12 GetNumberOfActionsCompleted() ......................... 38  
4.6.3.13 GetPossibleActions() .................................... 38  
4.6.3.14 InitializeBaseItem() ...................................... 38  
4.6.3.15 RemoveAttributes() ...................................... 38  
4.6.3.16 RemoveCategoriesExcluding() .......................... 39  
4.6.3.17 RemoveCategoriesSpecified() .......................... 39  
4.6.3.18 SetNewModel() .......................................... 39  
4.6.3.19 SetUpBaseItem() ......................................... 40  
4.6.3.20 UpdateItemEvent() ....................................... 40  

4.6.4 Member Data Documentation ................................ 40  
4.6.4.1 DirtyFlag .................................................. 40  
4.6.4.2 DiscardFlag ............................................... 40  
4.6.4.3 RemovalFlag ............................................... 40  
4.6.4.4 UpdateExistingFlag ....................................... 41  

4.6.5 Property Documentation .................................... 41  
4.6.5.1 ActionModifiedModels .................................... 41  
4.6.5.2 ActionModifiedSprites ................................... 41  
4.6.5.3 FlavorText .................................................. 41  
4.6.5.4 InventorySprite .......................................... 42  
4.6.5.5 ItemName ................................................... 42  
4.6.5.6 ModifyingActionNames .................................... 42
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.6.5.7</td>
<td>Rarity</td>
<td>42</td>
</tr>
<tr>
<td>4.6.5.8</td>
<td>Types</td>
<td>43</td>
</tr>
<tr>
<td>4.6.5.9</td>
<td>WorldModel</td>
<td>43</td>
</tr>
<tr>
<td>4.6.6</td>
<td>Event Documentation</td>
<td>43</td>
</tr>
<tr>
<td>4.6.6.1</td>
<td>UpdateItemName</td>
<td>43</td>
</tr>
<tr>
<td>4.6.6.2</td>
<td>UpdateItemSprite</td>
<td>43</td>
</tr>
<tr>
<td>4.7</td>
<td>BaseItemPopup Class Reference</td>
<td>44</td>
</tr>
<tr>
<td>4.7.1</td>
<td>Detailed Description</td>
<td>44</td>
</tr>
<tr>
<td>4.7.2</td>
<td>Member Function Documentation</td>
<td>45</td>
</tr>
<tr>
<td>4.7.2.1</td>
<td>OnGUI()</td>
<td>45</td>
</tr>
<tr>
<td>4.8</td>
<td>BaseItemPopupAttribute Class Reference</td>
<td>45</td>
</tr>
<tr>
<td>4.9</td>
<td>BeaconInteractable Class Reference</td>
<td>46</td>
</tr>
<tr>
<td>4.9.1</td>
<td>Detailed Description</td>
<td>47</td>
</tr>
<tr>
<td>4.9.2</td>
<td>Member Function Documentation</td>
<td>47</td>
</tr>
<tr>
<td>4.9.2.1</td>
<td>TurnLightOn()</td>
<td>47</td>
</tr>
<tr>
<td>4.10</td>
<td>BezierLine Class Reference</td>
<td>48</td>
</tr>
<tr>
<td>4.10.1</td>
<td>Detailed Description</td>
<td>49</td>
</tr>
<tr>
<td>4.10.2</td>
<td>Property Documentation</td>
<td>49</td>
</tr>
<tr>
<td>4.10.2.1</td>
<td>ControlPoint</td>
<td>49</td>
</tr>
<tr>
<td>4.10.2.2</td>
<td>EndPoint</td>
<td>49</td>
</tr>
<tr>
<td>4.10.2.3</td>
<td>RendererComponent</td>
<td>49</td>
</tr>
<tr>
<td>4.10.2.4</td>
<td>StartPoint</td>
<td>50</td>
</tr>
<tr>
<td>4.11</td>
<td>Block Class Reference</td>
<td>50</td>
</tr>
<tr>
<td>4.11.1</td>
<td>Detailed Description</td>
<td>51</td>
</tr>
<tr>
<td>4.11.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>51</td>
</tr>
<tr>
<td>4.11.2.1</td>
<td>Block()</td>
<td>51</td>
</tr>
<tr>
<td>4.11.3</td>
<td>Member Function Documentation</td>
<td>51</td>
</tr>
<tr>
<td>4.11.3.1</td>
<td>ContainsPoint()</td>
<td>51</td>
</tr>
<tr>
<td>4.11.4</td>
<td>Property Documentation</td>
<td>52</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>4.14.3.7</td>
<td>Position</td>
<td>61</td>
</tr>
<tr>
<td>4.15</td>
<td>City.BuildingEnumerator Class Reference</td>
<td>61</td>
</tr>
<tr>
<td>4.15.1</td>
<td>Detailed Description</td>
<td>62</td>
</tr>
<tr>
<td>4.15.2</td>
<td>Member Function Documentation</td>
<td>62</td>
</tr>
<tr>
<td>4.15.2.1</td>
<td>Dispose()</td>
<td>63</td>
</tr>
<tr>
<td>4.15.2.2</td>
<td>MoveNext()</td>
<td>63</td>
</tr>
<tr>
<td>4.15.2.3</td>
<td>Reset()</td>
<td>63</td>
</tr>
<tr>
<td>4.15.3</td>
<td>Property Documentation</td>
<td>63</td>
</tr>
<tr>
<td>4.15.3.1</td>
<td>Current</td>
<td>63</td>
</tr>
<tr>
<td>4.16</td>
<td>BuildingGenerator Class Reference</td>
<td>64</td>
</tr>
<tr>
<td>4.16.1</td>
<td>Detailed Description</td>
<td>65</td>
</tr>
<tr>
<td>4.16.2</td>
<td>Member Function Documentation</td>
<td>65</td>
</tr>
<tr>
<td>4.16.2.1</td>
<td>Generate()</td>
<td>65</td>
</tr>
<tr>
<td>4.16.3</td>
<td>Property Documentation</td>
<td>65</td>
</tr>
<tr>
<td>4.16.3.1</td>
<td>CityCenterBuilding</td>
<td>66</td>
</tr>
<tr>
<td>4.17</td>
<td>buildingIndex Struct Reference</td>
<td>66</td>
</tr>
<tr>
<td>4.17.1</td>
<td>Detailed Description</td>
<td>66</td>
</tr>
<tr>
<td>4.17.2</td>
<td>Member Data Documentation</td>
<td>66</td>
</tr>
<tr>
<td>4.17.2.1</td>
<td>BuildingSize</td>
<td>66</td>
</tr>
<tr>
<td>4.17.2.2</td>
<td>PartNumber</td>
<td>67</td>
</tr>
<tr>
<td>4.18</td>
<td>DistrictConfiguration.BuildingTemplatePlacement Class Reference</td>
<td>67</td>
</tr>
<tr>
<td>4.18.1</td>
<td>Detailed Description</td>
<td>67</td>
</tr>
<tr>
<td>4.18.2</td>
<td>Member Data Documentation</td>
<td>67</td>
</tr>
<tr>
<td>4.18.2.1</td>
<td>Building</td>
<td>68</td>
</tr>
<tr>
<td>4.18.2.2</td>
<td>PlacementFrequency</td>
<td>68</td>
</tr>
<tr>
<td>4.19</td>
<td>CameraController Class Reference</td>
<td>68</td>
</tr>
<tr>
<td>4.19.1</td>
<td>Detailed Description</td>
<td>69</td>
</tr>
<tr>
<td>4.19.2</td>
<td>Member Function Documentation</td>
<td>70</td>
</tr>
<tr>
<td>4.19.2.1</td>
<td>RotateLeft()</td>
<td>70</td>
</tr>
<tr>
<td>4.19.2.2</td>
<td>RotateRight()</td>
<td>70</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>4.19.2.3</td>
<td>Zoom()</td>
<td>70</td>
</tr>
<tr>
<td>4.19.3</td>
<td>Member Data Documentation</td>
<td>70</td>
</tr>
<tr>
<td>4.19.3.1</td>
<td>CameraPositions</td>
<td>70</td>
</tr>
<tr>
<td>4.19.3.2</td>
<td>EndingTriggered</td>
<td>71</td>
</tr>
<tr>
<td>4.19.4</td>
<td>Property Documentation</td>
<td>71</td>
</tr>
<tr>
<td>4.19.4.1</td>
<td>CurrentView</td>
<td>71</td>
</tr>
<tr>
<td>4.19.4.2</td>
<td>ZoomLevel</td>
<td>71</td>
</tr>
<tr>
<td>4.20</td>
<td>ChooseItemAmountPanelBehavior Class Reference</td>
<td>71</td>
</tr>
<tr>
<td>4.20.1</td>
<td>Detailed Description</td>
<td>73</td>
</tr>
<tr>
<td>4.20.2</td>
<td>Member Function Documentation</td>
<td>73</td>
</tr>
<tr>
<td>4.20.2.1</td>
<td>CancelChosenAmount()</td>
<td>73</td>
</tr>
<tr>
<td>4.20.2.2</td>
<td>Change()</td>
<td>73</td>
</tr>
<tr>
<td>4.20.2.3</td>
<td>ChooseNumOfItemsToAffect()</td>
<td>74</td>
</tr>
<tr>
<td>4.20.2.4</td>
<td>CloseEntirePanel()</td>
<td>74</td>
</tr>
<tr>
<td>4.20.2.5</td>
<td>FinalizeAction()</td>
<td>74</td>
</tr>
<tr>
<td>4.20.2.6</td>
<td>OpenItemAmountPanel()</td>
<td>74</td>
</tr>
<tr>
<td>4.20.2.7</td>
<td>OpenItemDetailPanel()</td>
<td>74</td>
</tr>
<tr>
<td>4.20.3</td>
<td>Member Data Documentation</td>
<td>75</td>
</tr>
<tr>
<td>4.20.3.1</td>
<td>ItemNameDisplay</td>
<td>75</td>
</tr>
<tr>
<td>4.20.4</td>
<td>Property Documentation</td>
<td>75</td>
</tr>
<tr>
<td>4.20.4.1</td>
<td>SelectedItem</td>
<td>75</td>
</tr>
<tr>
<td>4.21</td>
<td>City Class Reference</td>
<td>75</td>
</tr>
<tr>
<td>4.21.1</td>
<td>Detailed Description</td>
<td>76</td>
</tr>
<tr>
<td>4.21.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>76</td>
</tr>
<tr>
<td>4.21.2.1</td>
<td>City()</td>
<td>76</td>
</tr>
<tr>
<td>4.21.3</td>
<td>Member Function Documentation</td>
<td>77</td>
</tr>
<tr>
<td>4.21.3.1</td>
<td>GetEnumerator()</td>
<td>77</td>
</tr>
<tr>
<td>4.21.4</td>
<td>Property Documentation</td>
<td>77</td>
</tr>
<tr>
<td>4.21.4.1</td>
<td>BoundingBox</td>
<td>77</td>
</tr>
<tr>
<td>4.21.4.2</td>
<td>Center</td>
<td>77</td>
</tr>
</tbody>
</table>
4.21.4.3 Districts ......................................................... 77
4.21.4.4 TallestBuilding ............................................. 78

4.22 CityBoundaries Class Reference ........................................ 78
  4.22.1 Detailed Description ........................................... 79
  4.22.2 Constructor & Destructor Documentation ......................... 79
    4.22.2.1 CityBoundaries() ....................................... 79
  4.22.3 Member Function Documentation .................................. 79
    4.22.3.1 BoundVector2d() ...................................... 79
    4.22.3.2 BoundVector3d() ...................................... 80
  4.22.4 Member Data Documentation ..................................... 80
    4.22.4.1 CityBounds ........................................... 80
  4.22.5 Property Documentation ........................................ 80
    4.22.5.1 RandomVector2d ....................................... 80
    4.22.5.2 RandomVector3d ....................................... 81

4.23 CityChunk Class Reference ........................................... 81
  4.23.1 Detailed Description ........................................... 82
  4.23.2 Constructor & Destructor Documentation .......................... 82
    4.23.2.1 CityChunk() ........................................ 82
  4.23.3 Member Function Documentation .................................. 82
    4.23.3.1 Load() ............................................. 83
    4.23.3.2 Unload() ........................................... 83
  4.23.4 Property Documentation ........................................ 83
    4.23.4.1 BoundingBox ......................................... 83
    4.23.4.2 Buildings .......................................... 83
    4.23.4.3 CurrentCoroutine ................................... 83
    4.23.4.4 IsLoaded ......................................... 84
    4.23.4.5 Location ........................................... 84

4.24 CityChunkManager Class Reference .................................... 84
  4.24.1 Detailed Description ........................................... 85
  4.24.2 Member Function Documentation .................................. 86

Generated by Doxygen
4.29.2.7 SetTimeScale() .................................................. 105
4.29.2.8 TeleportToEnding() .......................................... 105
4.29.2.9 Test() .......................................................... 106

4.30 ContainerCategory Class Reference ........................................ 106
  4.30.1 Detailed Description ............................................. 108
  4.30.2 Member Function Documentation .................................. 108
      4.30.2.1 GetDuplicate() ........................................... 108
      4.30.2.2 ReadyCategory() ......................................... 108
      4.30.2.3 SetDown() ................................................. 108
  4.30.3 Property Documentation ......................................... 109
      4.30.3.1 Size .......................................................... 109

4.31 ControlScheme Class Reference .......................................... 109
  4.31.1 Detailed Description ............................................. 110
  4.31.2 Member Data Documentation ...................................... 110
      4.31.2.1 Action ...................................................... 110
      4.31.2.2 Back ....................................................... 110
      4.31.2.3 BackSecondary .......................................... 110
      4.31.2.4 CameraLeft .............................................. 111
      4.31.2.5 CameraRight ............................................. 111
      4.31.2.6 CameraZoomAxis ......................................... 111
      4.31.2.7 CameraZoomInKey ....................................... 111
      4.31.2.8 CameraZoomOutKey ..................................... 111
      4.31.2.9 Crafting .................................................. 111
      4.31.2.10 Forward .................................................. 112
      4.31.2.11 ForwardSecondary ...................................... 112
      4.31.2.12 Inventory .............................................. 112
      4.31.2.13 Jump ..................................................... 112
      4.31.2.14 Left ...................................................... 112
      4.31.2.15 LeftSecondary ......................................... 112
      4.31.2.16 Pause .................................................... 113
4.34.4.1 CreatureCount

4.34.4.2 IdealCreatureCount

4.35 CreatureSpawner Class Reference

4.35.1 Detailed Description

4.35.2 Member Function Documentation

4.35.2.1 CreatureKilled()

4.35.2.2 findSpawnLocation()

4.35.2.3 Init()

4.35.3 Member Data Documentation

4.35.3.1 creatureManager

4.35.3.2 maxCreatureCount

4.35.3.3 meanCreatureCount

4.35.3.4 minCreatureCount

4.35.4 Property Documentation

4.35.4.1 Count

4.36 CreatureTracker Class Reference

4.36.1 Detailed Description

4.36.2 Member Data Documentation

4.36.2.1 Index

4.36.2.2 IsDead

4.37 DayNight Class Reference

4.37.1 Detailed Description

4.38 DayNightSkybox Class Reference

4.39 DeathFlavorText Class Reference

4.39.1 Detailed Description

4.39.2 Member Data Documentation

4.39.2.1 flavorText

4.40 DeathManager Class Reference

4.40.1 Detailed Description
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.40.2 Member Function Documentation</td>
<td>131</td>
</tr>
<tr>
<td>4.40.2.1 Death()</td>
<td>131</td>
</tr>
<tr>
<td>4.41 DebrisController Class Reference</td>
<td>131</td>
</tr>
<tr>
<td>4.41.1 Detailed Description</td>
<td>132</td>
</tr>
<tr>
<td>4.41.2 Member Data Documentation</td>
<td>132</td>
</tr>
<tr>
<td>4.41.2.1 Debris</td>
<td>133</td>
</tr>
<tr>
<td>4.41.2.2 DebrisStartThreshold</td>
<td>133</td>
</tr>
<tr>
<td>4.41.2.3 UseCustomValues</td>
<td>133</td>
</tr>
<tr>
<td>4.41.3 Property Documentation</td>
<td>133</td>
</tr>
<tr>
<td>4.41.3.1 DebrisLevel</td>
<td>133</td>
</tr>
<tr>
<td>4.41.3.2 WindVectorXZ</td>
<td>133</td>
</tr>
<tr>
<td>4.42 DebugCity Class Reference</td>
<td>134</td>
</tr>
<tr>
<td>4.42.1 Detailed Description</td>
<td>134</td>
</tr>
<tr>
<td>4.43 Dial Class Reference</td>
<td>135</td>
</tr>
<tr>
<td>4.43.1 Detailed Description</td>
<td>135</td>
</tr>
<tr>
<td>4.43.2 Member Data Documentation</td>
<td>136</td>
</tr>
<tr>
<td>4.43.2.1 knobDegree</td>
<td>136</td>
</tr>
<tr>
<td>4.44 District Class Reference</td>
<td>136</td>
</tr>
<tr>
<td>4.44.1 Detailed Description</td>
<td>137</td>
</tr>
<tr>
<td>4.44.2 Constructor &amp; Destructor Documentation</td>
<td>137</td>
</tr>
<tr>
<td>4.44.2.1 District()</td>
<td>137</td>
</tr>
<tr>
<td>4.44.3 Member Function Documentation</td>
<td>137</td>
</tr>
<tr>
<td>4.44.3.1 ContainsPoint()</td>
<td>138</td>
</tr>
<tr>
<td>4.44.4 Property Documentation</td>
<td>138</td>
</tr>
<tr>
<td>4.44.4.1 Blocks</td>
<td>138</td>
</tr>
<tr>
<td>4.44.4.2 BoundingBox</td>
<td>138</td>
</tr>
<tr>
<td>4.44.4.3 Configuration</td>
<td>138</td>
</tr>
<tr>
<td>4.44.4.4 EdgeVertices</td>
<td>139</td>
</tr>
<tr>
<td>4.44.4.5 Name</td>
<td>139</td>
</tr>
<tr>
<td>4.44.4.6 SeedPoint</td>
<td>139</td>
</tr>
</tbody>
</table>
4.45 DistrictConfiguration Class Reference ........................................ 139

4.45.1 Detailed Description ......................................................... 141

4.45.2 Member Data Documentation ............................................. 141

4.45.2.1 districtProceduralMaterials ........................................ 141

4.45.2.2 districtProceduralWindowMaterials ................................. 142

4.45.2.3 Doors ........................................................................ 142

4.45.2.4 MaxFloors ................................................................... 142

4.45.2.5 MinFloors ................................................................... 142

4.45.2.6 Shelters ....................................................................... 142

4.45.2.7 WeenieBuildingTemplate ............................................. 143

4.45.3 Property Documentation .................................................... 143

4.45.3.1 DistrictAttachments .................................................... 143

4.45.3.2 DistrictMaterials .......................................................... 143

4.45.3.3 DistrictMaxWindowSpacings .......................................... 143

4.45.3.4 DistrictMinWindowSpacings .......................................... 143

4.45.3.5 DistrictPosters ............................................................. 144

4.45.3.6 DistrictWindowMaterials .............................................. 144

4.45.3.7 DistrictWindows ............................................................ 144

4.45.3.8 DistrictWindowWashers .............................................. 144

4.45.3.9 MaterialAlbedoColors .................................................. 144

4.45.3.10 MaxAttachmentChance .............................................. 145

4.45.3.11 MinAttachmentChance .............................................. 145

4.45.3.12 Name ...................................................................... 145

4.45.3.13 SizeXDefinitions ........................................................ 145

4.45.3.14 WindowWasherChance ................................................ 145

4.45.3.15 WindowWasherMaxLengthCurve .................................. 146

4.45.3.16 WindowWasherStartUpChance ..................................... 146

4.46 DistrictGenerator Class Reference .......................................... 146

4.46.1 Detailed Description ......................................................... 147

4.46.2 Member Function Documentation ....................................... 147
4.46.2.1 Generate() ........................................ 147

4.47 DistrictItemConfiguration Class Reference ........................................ 148

4.47.1 Detailed Description ........................................ 149

4.47.2 Constructor & Destructor Documentation ........................................ 149

4.47.2.1 DistrictItemConfiguration() .................................. 149

4.47.3 Property Documentation ........................................ 149

4.47.3.1 DoorExtents ........................................ 149

4.47.3.2 DoorTemplates ........................................ 149

4.47.3.3 ItemExtents ........................................ 150

4.47.3.4 ItemNames ........................................ 150

4.47.3.5 ItemTemplates ........................................ 150

4.47.3.6 ShelterExtents ........................................ 150

4.47.3.7 ShelterTemplates ........................................ 151

4.48 DistrictItemRarityConfiguration Class Reference ........................................ 151

4.48.1 Detailed Description ........................................ 151

4.48.2 Constructor & Destructor Documentation ........................................ 152

4.48.2.1 DistrictItemRarityConfiguration() .................................. 152

4.48.3 Member Function Documentation ........................................ 152

4.48.3.1 GetWeightedRandomItemIndex() .................................. 152

4.48.3.2 setUpVoseAlias() ........................................ 152

4.49 DistrictPopup Class Reference ........................................ 153

4.49.1 Detailed Description ........................................ 154

4.49.2 Member Function Documentation ........................................ 154

4.49.2.1 OnGUI() ........................................ 154

4.50 DistrictPopupAttribute Class Reference ........................................ 155

4.50.1 Detailed Description ........................................ 155

4.51 DiurnalTemperatureVariance Class Reference ........................................ 156

4.51.1 Detailed Description ........................................ 156

4.51.2 Property Documentation ........................................ 156

4.51.2.1 TemperatureVariance ........................................ 156
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.52 DummyGenerator Class Reference</td>
<td>157</td>
</tr>
<tr>
<td>4.52.1 Detailed Description</td>
<td>158</td>
</tr>
<tr>
<td>4.52.2 Member Data Documentation</td>
<td>158</td>
</tr>
<tr>
<td>4.52.2.1 buildings</td>
<td>158</td>
</tr>
<tr>
<td>4.52.2.2 doors</td>
<td>158</td>
</tr>
<tr>
<td>4.52.2.3 shelters</td>
<td>158</td>
</tr>
<tr>
<td>4.53 DummyNoteGenerator Class Reference</td>
<td>159</td>
</tr>
<tr>
<td>4.53.1 Detailed Description</td>
<td>159</td>
</tr>
<tr>
<td>4.54 EndCreditsBehavior Class Reference</td>
<td>160</td>
</tr>
<tr>
<td>4.54.1 Detailed Description</td>
<td>160</td>
</tr>
<tr>
<td>4.55 EndingController Class Reference</td>
<td>161</td>
</tr>
<tr>
<td>4.55.1 Detailed Description</td>
<td>162</td>
</tr>
<tr>
<td>4.55.2 Member Function Documentation</td>
<td>162</td>
</tr>
<tr>
<td>4.55.2.1 BeaconTurnedOn()</td>
<td>162</td>
</tr>
<tr>
<td>4.55.2.2 EndingTrigger()</td>
<td>162</td>
</tr>
<tr>
<td>4.55.3 Member Data Documentation</td>
<td>162</td>
</tr>
<tr>
<td>4.55.3.1 disableThese</td>
<td>163</td>
</tr>
<tr>
<td>4.55.3.2 hideThese</td>
<td>163</td>
</tr>
<tr>
<td>4.56 EndingTrigger Class Reference</td>
<td>163</td>
</tr>
<tr>
<td>4.56.1 Detailed Description</td>
<td>164</td>
</tr>
<tr>
<td>4.57 EquipableCategory Class Reference</td>
<td>165</td>
</tr>
<tr>
<td>4.57.1 Detailed Description</td>
<td>167</td>
</tr>
<tr>
<td>4.57.2 Member Function Documentation</td>
<td>167</td>
</tr>
<tr>
<td>4.57.2.1 Equip()</td>
<td>167</td>
</tr>
<tr>
<td>4.57.2.2 GetDuplicate()</td>
<td>167</td>
</tr>
<tr>
<td>4.57.2.3 ReadyCategory()</td>
<td>168</td>
</tr>
<tr>
<td>4.57.2.4 UnEquip()</td>
<td>168</td>
</tr>
<tr>
<td>4.57.3 Member Data Documentation</td>
<td>168</td>
</tr>
<tr>
<td>4.57.3.1 equipActionName</td>
<td>168</td>
</tr>
<tr>
<td>4.57.3.2 equipedAttributeName</td>
<td>168</td>
</tr>
</tbody>
</table>
4.57.3.3 unequipActionName .................................................. 168

4.57.4 Property Documentation ............................................... 169

4.57.4.1 Equipped ............................................................. 169

4.58 EquippedItemDropdown Class Reference .............................. 169

4.58.1 Detailed Description .................................................. 170

4.58.2 Member Function Documentation .................................... 171

4.58.2.1 DisplayOptions() .................................................. 171

4.58.2.2 GetSelected() ..................................................... 171

4.58.2.3 HideOptions() ..................................................... 171

4.58.2.4 SetEquipped() ..................................................... 171

4.58.2.5 ToggleDisplay() ................................................... 171

4.58.2.6 Unequipped() ..................................................... 172

4.58.3 Member Data Documentation ........................................ 172

4.58.3.1 contentPanel ..................................................... 172

4.58.3.2 equippedItemHolder ............................................. 172

4.59 EventManager Class Reference ....................................... 173

4.59.1 Detailed Description .................................................. 174

4.59.2 Member Function Documentation .................................... 174

4.59.2.1 PlayerBoardRaftDelegate() ...................................... 174

4.59.2.2 RadioMusicDelegate() ........................................... 174

4.59.2.3 RadioMusicTurnedOff() ......................................... 174

4.59.2.4 RadioMusicTurnedOn() .......................................... 174

4.59.2.5 RaftBoarded() ................................................... 175

4.59.2.6 StormStart() ..................................................... 175

4.59.2.7 StormStartedDelegate() ......................................... 175

4.59.2.8 StormStop() ..................................................... 175

4.59.2.9 StormStoppedDelegate() ........................................ 175

4.59.2.10 WeatherUpdated() .............................................. 176

4.59.2.11 WeatherUpdatedDelegate() .................................... 176

4.59.3 Event Documentation .................................................. 176
4.59.3.1 PlayerBoardRaftSubscription ........................................ 176
4.59.3.2 RadioMusicSubscription ............................................. 176
4.59.3.3 StormStartedSubscription ......................................... 176
4.59.3.4 StormStoppedSubscription ......................................... 176
4.59.3.5 WeatherUpdatedSubscription ....................................... 177
4.60 FaderManager Class Reference ........................................... 177
  4.60.1 Detailed Description ................................................... 178
  4.60.2 Member Function Documentation ..................................... 178
    4.60.2.1 EndGameFade() ..................................................... 178
    4.60.2.2 GoToMain() .......................................................... 179
    4.60.2.3 ShowMainMenuButton() .......................................... 179
4.61 FileManager Class Reference ............................................. 179
  4.61.1 Detailed Description ................................................... 180
  4.61.2 Member Function Documentation ..................................... 180
    4.61.2.1 BuildPath() ....................................................... 180
    4.61.2.2 GetDocument() .................................................... 180
    4.61.2.3 SaveAllDocuments() ............................................. 181
    4.61.2.4 SaveFile() .......................................................... 181
4.62 FireBaseCategory Class Reference ....................................... 182
  4.62.1 Detailed Description ................................................... 184
  4.62.2 Member Function Documentation ..................................... 184
    4.62.2.1 AddFuel() .......................................................... 184
    4.62.2.2 CalculateRemainingFuel() ..................................... 184
    4.62.2.3 GetDuplicate() .................................................... 185
    4.62.2.4 ReadyCategory() .................................................. 185
    4.62.2.5 SetDown() .......................................................... 185
  4.62.3 Property Documentation ................................................ 185
    4.62.3.1 BurnRateMultiplier .............................................. 186
    4.62.3.2 FuelRemaining ..................................................... 186
    4.62.3.3 StartingFuel ........................................................ 186
4.71.2 Member Function Documentation .............................................. 219

4.71.2.1 SetFloatHeight() ............................................................. 219

4.72 FloodWater Class Reference ....................................................... 219

4.72.1 Detailed Description ............................................................ 220

4.72.2 Property Documentation ....................................................... 220

4.72.2.1 WaterLevelHeight .......................................................... 220

4.73 FuelCategory Class Reference ............................................... 221

4.73.1 Detailed Description ............................................................ 223

4.73.2 Member Function Documentation ............................................ 223

4.73.2.1 GetDuplicate() ............................................................... 223

4.73.2.2 ReadyCategory() ............................................................. 223

4.73.3 Property Documentation ....................................................... 223

4.73.3.1 BurnTime ..................................................................... 224

4.74 FXSplash Class Reference .......................................................... 224

4.74.1 Detailed Description ............................................................... 225

4.74.2 Member Function Documentation ............................................ 225

4.74.2.1 setAnimationSpeed() ....................................................... 225

4.74.3 Member Data Documentation ................................................ 226

4.74.3.1 SplashAnimator ............................................................... 226

4.75 FXSplashManager Class Reference ............................................. 226

4.75.1 Detailed Description ............................................................... 227

4.75.2 Member Function Documentation ............................................ 227

4.75.2.1 CreateSplash(1/3) ............................................................ 228

4.75.2.2 CreateSplash(2/3) ............................................................ 228

4.75.2.3 CreateSplash(3/3) ............................................................ 228

4.75.2.4 FillPool() ................................................................. 229

4.75.3 Property Documentation ....................................................... 229

4.75.3.1 SplashPoolSize ............................................................... 229

4.76 Game Class Reference .............................................................. 229

4.76.1 Detailed Description ............................................................... 231
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.88.1</td>
<td>Detailed Description</td>
<td>265</td>
</tr>
<tr>
<td>4.88.2</td>
<td>Member Function Documentation</td>
<td>266</td>
</tr>
<tr>
<td>4.88.2.1</td>
<td>Equip()</td>
<td>266</td>
</tr>
<tr>
<td>4.88.2.2</td>
<td>SetUpTool()</td>
<td>266</td>
</tr>
<tr>
<td>4.88.2.3</td>
<td>Unequip()</td>
<td>266</td>
</tr>
<tr>
<td>4.88.2.4</td>
<td>Use()</td>
<td>267</td>
</tr>
<tr>
<td>4.89</td>
<td>IdolCategory Class Reference</td>
<td>268</td>
</tr>
<tr>
<td>4.89.1</td>
<td>Detailed Description</td>
<td>270</td>
</tr>
<tr>
<td>4.89.2</td>
<td>Member Function Documentation</td>
<td>270</td>
</tr>
<tr>
<td>4.89.2.1</td>
<td>ApplyBenefit()</td>
<td>270</td>
</tr>
<tr>
<td>4.89.2.2</td>
<td>Equip()</td>
<td>270</td>
</tr>
<tr>
<td>4.89.2.3</td>
<td>ReadyCategory()</td>
<td>270</td>
</tr>
<tr>
<td>4.89.2.4</td>
<td>RemoveBenefit()</td>
<td>271</td>
</tr>
<tr>
<td>4.90</td>
<td>Ingredient Struct Reference</td>
<td>271</td>
</tr>
<tr>
<td>4.90.1</td>
<td>Detailed Description</td>
<td>272</td>
</tr>
<tr>
<td>4.90.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>272</td>
</tr>
<tr>
<td>4.90.2.1</td>
<td>Ingredient()</td>
<td>272</td>
</tr>
<tr>
<td>4.90.3</td>
<td>Property Documentation</td>
<td>272</td>
</tr>
<tr>
<td>4.90.3.1</td>
<td>Amount</td>
<td>272</td>
</tr>
<tr>
<td>4.90.3.2</td>
<td>AssociatedStackId</td>
<td>272</td>
</tr>
<tr>
<td>4.90.3.3</td>
<td>IngredientName</td>
<td>273</td>
</tr>
<tr>
<td>4.90.3.4</td>
<td>UseType</td>
<td>273</td>
</tr>
<tr>
<td>4.91</td>
<td>IngredientButtonBehavior Class Reference</td>
<td>273</td>
</tr>
<tr>
<td>4.91.1</td>
<td>Detailed Description</td>
<td>274</td>
</tr>
<tr>
<td>4.91.2</td>
<td>Member Function Documentation</td>
<td>274</td>
</tr>
<tr>
<td>4.91.2.1</td>
<td>DeselectIngredient()</td>
<td>275</td>
</tr>
<tr>
<td>4.91.2.2</td>
<td>SelectIngredient()</td>
<td>275</td>
</tr>
<tr>
<td>4.91.2.3</td>
<td>SetUpIngredient()</td>
<td>275</td>
</tr>
<tr>
<td>4.91.2.4</td>
<td>UpdateIngredientEvent()</td>
<td>275</td>
</tr>
<tr>
<td>4.91.3</td>
<td>Event Documentation</td>
<td>275</td>
</tr>
</tbody>
</table>
4.91.3.1 UpdateIngredientSelection ........................................ 276

4.92 InteractableObject Class Reference .................................. 276

4.92.1 Detailed Description .................................................. 277

4.92.2 Member Function Documentation .................................. 277

4.92.2.1 PerformAction() .................................................. 277

4.92.2.2 SetAction() ....................................................... 277

4.92.2.3 SetUp() .......................................................... 278

4.92.3 Member Data Documentation ........................................ 278

4.92.3.1 setupComplete .................................................... 278

4.92.4 Property Documentation .............................................. 278

4.92.4.1 Show ............................................................. 278

4.92.4.2 Text ............................................................... 279

4.93 InteractableRadioModel Class Reference ............................... 279

4.93.1 Detailed Description .................................................. 280

4.93.2 Member Function Documentation .................................. 281

4.93.2.1 ActivateRadio() .................................................. 281

4.93.2.2 DeactivateRadio() ............................................... 281

4.93.2.3 PushPowerButton() .............................................. 281

4.93.2.4 PushVolumeDownButton() .................................... 281

4.93.2.5 PushVolumeUpButton() ........................................ 281

4.93.2.6 SetKnobRotation() .............................................. 281

4.93.2.7 SetSlider() ....................................................... 282

4.93.2.8 SetUpRadioCanvas() ............................................ 282

4.94 InteractableSetUp Class Reference .................................... 283

4.94.1 Detailed Description .................................................. 283

4.94.2 Member Data Documentation ........................................ 284

4.94.2.1 ItemName ........................................................ 284

4.95 Inventory Class Reference ............................................... 284

4.95.1 Detailed Description .................................................. 286

4.95.2 Constructor & Destructor Documentation ........................... 286
4.95.2.1 Inventory() .................................................. 286
4.95.2.2 Inventory() .................................................. 287
4.95.3 Member Function Documentation ................................ 287
  4.95.3.1 AddItem() .................................................. 287
  4.95.3.2 CheckRecipePossible() .................................... 288
  4.95.3.3 CheckRequirementMet() ................................... 288
  4.95.3.4 CombineStacks() .......................................... 288
  4.95.3.5 GetAllItemsWithTag() ..................................... 289
  4.95.3.6 GetInventory() ............................................ 289
  4.95.3.7 GetInventoryBaseItem() .................................. 289
  4.95.3.8 GetItemsByType() ......................................... 290
  4.95.3.9 GetNextOpenSlot() ...................................... 290
  4.95.3.10 GetStacks() ............................................. 290
  4.95.3.11 LoadInventory() ......................................... 291
  4.95.3.12 RemoveStack() .......................................... 291
  4.95.3.13 UpdateTypeAmount() .................................... 291
  4.95.3.14 UseItem() ............................................... 292
4.95.4 Member Data Documentation .................................... 292
  4.95.4.1 contents .................................................. 292
4.95.5 Property Documentation ........................................ 292
  4.95.5.1 InventorySize ............................................ 292
4.96 InventoryInteractable Class Reference .......................... 293
  4.96.1 Detailed Description ....................................... 294
  4.96.2 Member Function Documentation ............................. 294
    4.96.2.1 SetUp() ................................................ 295
  4.96.3 Property Documentation .................................... 295
    4.96.3.1 AttachedInventory .................................... 295
4.97 InventoryItemYAMLModel Class Reference ........................ 295
  4.97.1 Detailed Description ....................................... 296
  4.97.2 Property Documentation .................................... 296
<table>
<thead>
<tr>
<th>Section</th>
<th>Content</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.97.2.1</td>
<td>Item</td>
</tr>
<tr>
<td>4.97.2.2</td>
<td>ItemAmount</td>
</tr>
<tr>
<td>4.97.2.3</td>
<td>ItemCategories</td>
</tr>
<tr>
<td>4.97.2.4</td>
<td>StackId</td>
</tr>
<tr>
<td>4.98</td>
<td>InventoryUI Class Reference</td>
</tr>
<tr>
<td>4.98.1</td>
<td>Detailed Description</td>
</tr>
<tr>
<td>4.98.2</td>
<td>Member Function Documentation</td>
</tr>
<tr>
<td>4.98.2.1</td>
<td>DisplayInventory()</td>
</tr>
<tr>
<td>4.98.2.2</td>
<td>GetStackUI()</td>
</tr>
<tr>
<td>4.98.2.3</td>
<td>LoadNewInventory()</td>
</tr>
<tr>
<td>4.98.2.4</td>
<td>RefreshInventoryPanel()</td>
</tr>
<tr>
<td>4.98.3</td>
<td>Member Data Documentation</td>
</tr>
<tr>
<td>4.98.3.1</td>
<td>TargetInventory</td>
</tr>
<tr>
<td>4.98.4</td>
<td>Property Documentation</td>
</tr>
<tr>
<td>4.98.4.1</td>
<td>ItemSpriteManager</td>
</tr>
<tr>
<td>4.98.4.2</td>
<td>ItemsToDiscard</td>
</tr>
<tr>
<td>4.99</td>
<td>InventoryYAMLModel Class Reference</td>
</tr>
<tr>
<td>4.99.1</td>
<td>Detailed Description</td>
</tr>
<tr>
<td>4.99.2</td>
<td>Property Documentation</td>
</tr>
<tr>
<td>4.99.2.1</td>
<td>InventoryName</td>
</tr>
<tr>
<td>4.99.2.2</td>
<td>InventorySize</td>
</tr>
<tr>
<td>4.99.2.3</td>
<td>Items</td>
</tr>
<tr>
<td>4.100</td>
<td>InventoryYamlParser Class Reference</td>
</tr>
<tr>
<td>4.100.1</td>
<td>Detailed Description</td>
</tr>
<tr>
<td>4.100.2</td>
<td>Constructor &amp; Destructor Documentation</td>
</tr>
<tr>
<td>4.100.2.1</td>
<td>InventoryYamlParser()</td>
</tr>
<tr>
<td>4.100.3</td>
<td>Member Function Documentation</td>
</tr>
<tr>
<td>4.100.3.1</td>
<td>AddNewInventory()</td>
</tr>
<tr>
<td>4.100.3.2</td>
<td>GetInventoryContents()</td>
</tr>
<tr>
<td>4.100.3.3</td>
<td>LoadInventories()</td>
</tr>
</tbody>
</table>
4.109.2.1 AddTemplatesToItemPool() ........................................... 335
4.109.2.2 calculateBounds() .................................................... 335
4.109.2.3 GetItemExtents() ..................................................... 336
4.109.2.4 getRarityInformation() ............................................. 336
4.109.2.5 SetSeed() ................................................................ 336
4.109.3 Member Data Documentation ........................................... 337
4.109.3.1 districtItemInfo ......................................................... 337
4.109.3.2 poolManager ........................................................... 337
4.110 ItemPlacementSamplePoint Class Reference ......................... 337
4.110.1 Detailed Description ...................................................... 338
4.110.2 Member Enumeration Documentation ................................ 338
4.110.2.1 PointType ............................................................... 338
4.110.3 Property Documentation ................................................ 338
4.110.3.1 District ................................................................. 339
4.110.3.2 GridPoint ............................................................... 339
4.110.3.3 ItemIndex .............................................................. 339
4.110.3.4 LocalTargetSurfaceLocation ...................................... 339
4.110.3.5 MinDistance .......................................................... 340
4.110.3.6 Size ................................................................ 340
4.110.3.7 Type ................................................................ 340
4.110.3.8 WorldSpaceLocation ............................................... 340
4.111 ItemPoolInfo Class Reference ............................................. 341
4.111.1 Detailed Description ...................................................... 341
4.111.2 Constructor & Destructor Documentation ......................... 341
4.111.2.1 ItemPoolInfo() ....................................................... 342
4.111.3 Member Function Documentation .................................... 342
4.111.3.1 AddItemInfo() ....................................................... 342
4.111.3.2 RemoveItemInfo() .................................................. 342
4.111.4 Property Documentation ............................................... 343
4.111.4.1 Activated .............................................................. 343
4.111.4.2 ItemNames ............................................. 343
4.111.4.3 Items ................................................. 343
4.111.4.4 Locations ............................................. 343

4.112 ItemPoolManager Class Reference ............................................. 344
4.112.1 Detailed Description ............................................. 346
4.112.2 Member Function Documentation ............................................. 346
  4.112.2.1 AddItemFromWorld() ............................................. 346
  4.112.2.2 AddItemToPool() ............................................. 346
  4.112.2.3 AddToGrid() ............................................. 347
  4.112.2.4 GetItemPool() ............................................. 347
  4.112.2.5 PointToGrid() ............................................. 347
  4.112.2.6 RemoveItemFromWorld() ............................................. 348
  4.112.2.7 SetUpItemPoolManager() ............................................. 348
  4.112.2.8 StartManagingPool() ............................................. 348

4.112.3 Member Data Documentation ............................................. 349
  4.112.3.1 grid ............................................. 349

4.113 ItemRarity Class Reference ............................................. 349
4.113.1 Detailed Description ............................................. 350
4.113.2 Member Function Documentation ............................................. 350
  4.113.2.1 GetRarity() ............................................. 350

4.113.3 Member Data Documentation ............................................. 350
  4.113.3.1 Common ............................................. 350
  4.113.3.2 Legendary ............................................. 350
  4.113.3.3 Rare ............................................. 351
  4.113.3.4 Uncommon ............................................. 351

4.114 ItemSerializer Class Reference ............................................. 351
4.114.1 Detailed Description ............................................. 352
4.114.2 Constructor & Destructor Documentation ............................................. 352
  4.114.2.1 ItemSerializer() ............................................. 352

4.114.3 Member Function Documentation ............................................. 353
4.117.2 Member Function Documentation ................................................. 363
  4.117.2.1 ClearAttributesAndActions() ............................................. 363
  4.117.2.2 ClearSubActionPanel() ...................................................... 363
  4.117.2.3 ClosePanel() ..................................................................... 363
  4.117.2.4 CreateSubAction() .............................................................. 363
  4.117.2.5 DisplayAttributesAndAction() .............................................. 364
  4.117.2.6 RefreshItemActions() ......................................................... 364
  4.117.2.7 RefreshItemPanel() ............................................................. 364
  4.117.2.8 SetItemData() ................................................................... 364
  4.117.2.9 SetSelectedItem() .................................................................. 365
  4.117.2.10 UpdateSelectedAmount() ................................................... 365

4.117.3 Property Documentation .......................................................... 365
  4.117.3.1 Attributes ........................................................................... 365
  4.117.3.2 PossibleActions ................................................................. 366
  4.117.3.3 PossibleSubActions ............................................................ 366

4.118 ItemStackUI Class Reference ...................................................... 367
  4.118.1 Detailed Description ............................................................... 369
  4.118.2 Member Function Documentation ............................................ 369
    4.118.2.1 CheckForModification() ..................................................... 369
    4.118.2.2 GetItemAttributes() .......................................................... 370
    4.118.2.3 GetItemName() ................................................................ 370
    4.118.2.4 GetMaxAmount() ................................................................. 370
    4.118.2.5 GetPossibleActions() ......................................................... 370
    4.118.2.6 GetStack() ....................................................................... 371
    4.118.2.7 HandleItemAmountTextChangeEvent() .................................. 371
    4.118.2.8 HandleItemIconChangeEvent() ............................................. 371
    4.118.2.9 HandleItemNameTextChangeEvent() ...................................... 371
    4.118.2.10 PreserveOriginal() ............................................................ 372
    4.118.2.11 RefreshInventoryItem() .................................................... 372
    4.118.2.12 SetHoverPanelActive() ...................................................... 372
4.120.2 Property Documentation

4.120.2.1 BaseItem

4.120.2.2 ItemCategories

4.120.2.3 ItemName

4.121 JellyFishSpawner Class Reference

4.121.1 Detailed Description

4.122 KeyCodeConfigMenuBehavior Class Reference

4.122.1 Detailed Description

4.122.2 Member Function Documentation

4.122.2.1 OnCloseClick()

4.122.2.2 OnKeyButtonClick()

4.122.2.3 SetKey()

4.123 LandMovement Class Reference

4.123.1 Detailed Description

4.123.2 Member Function Documentation

4.123.2.1 Climb()

4.123.2.2 GetClimbHeight()

4.123.2.3 GetRaycastHeight()

4.123.2.4 Idle()

4.123.2.5 Jump()

4.123.2.6 JumpForce()

4.123.2.7 JumpLand()

4.123.2.8 Move()

4.123.2.9 OnStateEnter()

4.123.2.10 OnStateExit()

4.124 LightCategory Class Reference

4.124.1 Detailed Description

4.124.2 Member Function Documentation

4.124.2.1 AddFuel()

4.124.2.2 CalculateRemainingFuel()
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.130.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>404</td>
</tr>
<tr>
<td>4.130.2.1</td>
<td>LoadingScreenFlavorTextManager()</td>
<td>404</td>
</tr>
<tr>
<td>4.130.3</td>
<td>Member Function Documentation</td>
<td>405</td>
</tr>
<tr>
<td>4.130.3.1</td>
<td>GetRandom()</td>
<td>405</td>
</tr>
<tr>
<td>4.131</td>
<td>LoadingScreenFlavorTextYAMLParser Class Reference</td>
<td>405</td>
</tr>
<tr>
<td>4.131.1</td>
<td>Detailed Description</td>
<td>405</td>
</tr>
<tr>
<td>4.131.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>406</td>
</tr>
<tr>
<td>4.131.2.1</td>
<td>LoadingScreenFlavorTextYAMLParser()</td>
<td>406</td>
</tr>
<tr>
<td>4.131.3</td>
<td>Member Function Documentation</td>
<td>406</td>
</tr>
<tr>
<td>4.131.3.1</td>
<td>Load()</td>
<td>406</td>
</tr>
<tr>
<td>4.132</td>
<td>MainMenuBehavior Class Reference</td>
<td>407</td>
</tr>
<tr>
<td>4.132.1</td>
<td>Detailed Description</td>
<td>408</td>
</tr>
<tr>
<td>4.132.2</td>
<td>Member Function Documentation</td>
<td>408</td>
</tr>
<tr>
<td>4.132.2.1</td>
<td>OnBackToMainClick()</td>
<td>408</td>
</tr>
<tr>
<td>4.132.2.2</td>
<td>OnExitClick()</td>
<td>408</td>
</tr>
<tr>
<td>4.132.2.3</td>
<td>OnHelpClick()</td>
<td>408</td>
</tr>
<tr>
<td>4.132.2.4</td>
<td>OnSettingsClick()</td>
<td>409</td>
</tr>
<tr>
<td>4.132.2.5</td>
<td>OnStartClick()</td>
<td>409</td>
</tr>
<tr>
<td>4.133</td>
<td>MedicineCategory Class Reference</td>
<td>410</td>
</tr>
<tr>
<td>4.133.1</td>
<td>Detailed Description</td>
<td>412</td>
</tr>
<tr>
<td>4.133.2</td>
<td>Member Function Documentation</td>
<td>412</td>
</tr>
<tr>
<td>4.133.2.1</td>
<td>CureAll()</td>
<td>412</td>
</tr>
<tr>
<td>4.133.2.2</td>
<td>CureFoodPoisoning()</td>
<td>412</td>
</tr>
<tr>
<td>4.133.2.3</td>
<td>CurePneumonia()</td>
<td>413</td>
</tr>
<tr>
<td>4.133.2.4</td>
<td>GetDuplicate()</td>
<td>413</td>
</tr>
<tr>
<td>4.133.2.5</td>
<td>Heal()</td>
<td>413</td>
</tr>
<tr>
<td>4.133.2.6</td>
<td>ReadyCategory()</td>
<td>413</td>
</tr>
<tr>
<td>4.133.3</td>
<td>Property Documentation</td>
<td>413</td>
</tr>
<tr>
<td>4.133.3.1</td>
<td>HealthGain</td>
<td>414</td>
</tr>
<tr>
<td>4.133.3.2</td>
<td>Sickness</td>
<td>414</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>4.134</td>
<td>Movement Class Reference</td>
<td>415</td>
</tr>
<tr>
<td>4.134.1</td>
<td>Detailed Description</td>
<td>417</td>
</tr>
<tr>
<td>4.134.2</td>
<td>Member Function Documentation</td>
<td>417</td>
</tr>
<tr>
<td>4.134.2.1</td>
<td>Climb()</td>
<td>417</td>
</tr>
<tr>
<td>4.134.2.2</td>
<td>GetClimbHeight()</td>
<td>417</td>
</tr>
<tr>
<td>4.134.2.3</td>
<td>GetRaycastHeight()</td>
<td>417</td>
</tr>
<tr>
<td>4.134.2.4</td>
<td>Idle()</td>
<td>417</td>
</tr>
<tr>
<td>4.134.2.5</td>
<td>Jump()</td>
<td>417</td>
</tr>
<tr>
<td>4.134.2.6</td>
<td>Move()</td>
<td>418</td>
</tr>
<tr>
<td>4.134.2.7</td>
<td>OnStateEnter()</td>
<td>418</td>
</tr>
<tr>
<td>4.134.2.8</td>
<td>OnStateExit()</td>
<td>418</td>
</tr>
<tr>
<td>4.134.3</td>
<td>Member Data Documentation</td>
<td>418</td>
</tr>
<tr>
<td>4.134.3.1</td>
<td>AccumulatedFallDamage</td>
<td>418</td>
</tr>
<tr>
<td>4.134.3.2</td>
<td>climbHeight</td>
<td>418</td>
</tr>
<tr>
<td>4.134.3.3</td>
<td>RigidBody</td>
<td>419</td>
</tr>
<tr>
<td>4.134.4</td>
<td>Property Documentation</td>
<td>419</td>
</tr>
<tr>
<td>4.134.4.1</td>
<td>CurrentFallDamage</td>
<td>419</td>
</tr>
<tr>
<td>4.134.4.2</td>
<td>Speed</td>
<td>419</td>
</tr>
<tr>
<td>4.135</td>
<td>MysteryAnnouncement Class Reference</td>
<td>419</td>
</tr>
<tr>
<td>4.135.1</td>
<td>Detailed Description</td>
<td>420</td>
</tr>
<tr>
<td>4.135.2</td>
<td>Property Documentation</td>
<td>420</td>
</tr>
<tr>
<td>4.135.2.1</td>
<td>Category</td>
<td>420</td>
</tr>
<tr>
<td>4.135.2.2</td>
<td>EventPath</td>
<td>420</td>
</tr>
<tr>
<td>4.135.2.3</td>
<td>Speed</td>
<td>421</td>
</tr>
<tr>
<td>4.135.2.4</td>
<td>Text</td>
<td>421</td>
</tr>
<tr>
<td>4.135.2.5</td>
<td>Voice</td>
<td>421</td>
</tr>
<tr>
<td>4.136</td>
<td>Note Class Reference</td>
<td>422</td>
</tr>
<tr>
<td>4.136.1</td>
<td>Detailed Description</td>
<td>423</td>
</tr>
<tr>
<td>4.136.2</td>
<td>Member Function Documentation</td>
<td>424</td>
</tr>
<tr>
<td>4.136.2.1</td>
<td>Read()</td>
<td>424</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>4.136.2.2</td>
<td>SetUpNote()</td>
<td>424</td>
</tr>
<tr>
<td>4.136.3</td>
<td>Property Documentation</td>
<td>424</td>
</tr>
<tr>
<td>4.136.3.1</td>
<td>DisplayText</td>
<td>424</td>
</tr>
<tr>
<td>4.137</td>
<td>NoteData Class Reference</td>
<td>425</td>
</tr>
<tr>
<td>4.137.1</td>
<td>Detailed Description</td>
<td>425</td>
</tr>
<tr>
<td>4.137.2</td>
<td>Property Documentation</td>
<td>425</td>
</tr>
<tr>
<td>4.137.2.1</td>
<td>interactable</td>
<td>426</td>
</tr>
<tr>
<td>4.137.2.2</td>
<td>InventorySprite</td>
<td>426</td>
</tr>
<tr>
<td>4.137.2.3</td>
<td>Text</td>
<td>426</td>
</tr>
<tr>
<td>4.137.2.4</td>
<td>Title</td>
<td>426</td>
</tr>
<tr>
<td>4.137.2.5</td>
<td>Type</td>
<td>427</td>
</tr>
<tr>
<td>4.137.2.6</td>
<td>WorldModel</td>
<td>427</td>
</tr>
<tr>
<td>4.138</td>
<td>NoteFactory Class Reference</td>
<td>427</td>
</tr>
<tr>
<td>4.138.1</td>
<td>Detailed Description</td>
<td>428</td>
</tr>
<tr>
<td>4.138.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>428</td>
</tr>
<tr>
<td>4.138.2.1</td>
<td>NoteFactory()</td>
<td>428</td>
</tr>
<tr>
<td>4.138.3</td>
<td>Member Function Documentation</td>
<td>428</td>
</tr>
<tr>
<td>4.138.3.1</td>
<td>CreateInteractableItem()</td>
<td>428</td>
</tr>
<tr>
<td>4.138.3.2</td>
<td>LoadTemplates()</td>
<td>428</td>
</tr>
<tr>
<td>4.139</td>
<td>NoteUIController Class Reference</td>
<td>429</td>
</tr>
<tr>
<td>4.139.1</td>
<td>Detailed Description</td>
<td>430</td>
</tr>
<tr>
<td>4.139.2</td>
<td>Member Function Documentation</td>
<td>430</td>
</tr>
<tr>
<td>4.139.2.1</td>
<td>DisableUI()</td>
<td>430</td>
</tr>
<tr>
<td>4.139.2.2</td>
<td>SetText()</td>
<td>430</td>
</tr>
<tr>
<td>4.139.3</td>
<td>Member Data Documentation</td>
<td>431</td>
</tr>
<tr>
<td>4.139.3.1</td>
<td>Instance</td>
<td>431</td>
</tr>
<tr>
<td>4.139.3.2</td>
<td>NoteText</td>
<td>431</td>
</tr>
<tr>
<td>4.139.3.3</td>
<td>NoteUIPanel</td>
<td>431</td>
</tr>
<tr>
<td>4.140</td>
<td>YAMLParser Class Reference</td>
<td>432</td>
</tr>
<tr>
<td>4.140.1</td>
<td>Detailed Description</td>
<td>432</td>
</tr>
</tbody>
</table>
## 4.155.1 Detailed Description

- 482

## 4.156 ProceduralBuilding Class Reference

- 482

### 4.156.1 Detailed Description

- 485

### 4.156.2 Constructor & Destructor Documentation

#### 4.156.2.1 ProceduralBuilding()

- 485

### 4.156.3 Member Function Documentation

#### 4.156.3.1 Load()

- 485

### 4.156.4 Member Data Documentation

#### 4.156.4.1 BuildingScale

- 485

#### 4.156.4.2 Generator

- 485

### 4.156.5 Property Documentation

#### 4.156.5.1 Configuration

- 485

#### 4.156.5.2 NumberOfFloors

- 485

#### 4.156.5.3 Seed

- 485

## 4.157 ProceduralBuildingAttachment Class Reference

- 487

### 4.157.1 Detailed Description

- 487

### 4.157.2 Member Data Documentation

#### 4.157.2.1 AttachmentPoint

- 487

#### 4.157.2.2 HasRoof

- 487

### 4.157.3 Property Documentation

#### 4.157.3.1 AddWindows

- 487

#### 4.157.3.2 RoofLocation

- 487

#### 4.157.3.3 WindowPoints

- 487

## 4.158 ProceduralBuildingBase Class Reference

- 490

### 4.158.1 Detailed Description

- 490

### 4.158.2 Property Documentation

#### 4.158.2.1 AddWindows

- 490

#### 4.158.2.2 AttachmentPoints

- 490

#### 4.158.2.3 HasRoof

- 490

#### 4.158.2.4 HeightType

- 490
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.164.3.9</td>
<td>StaticDefaultPath</td>
<td>510</td>
</tr>
<tr>
<td>4.165</td>
<td>RadioButton Class Reference</td>
<td>511</td>
</tr>
<tr>
<td>4.165.1</td>
<td>Detailed Description</td>
<td>511</td>
</tr>
<tr>
<td>4.166</td>
<td>RaftCategory Class Reference</td>
<td>512</td>
</tr>
<tr>
<td>4.166.1</td>
<td>Detailed Description</td>
<td>514</td>
</tr>
<tr>
<td>4.166.2</td>
<td>Member Function Documentation</td>
<td>514</td>
</tr>
<tr>
<td>4.166.2.1</td>
<td>GetDuplicate()</td>
<td>514</td>
</tr>
<tr>
<td>4.166.2.2</td>
<td>ReadyCategory()</td>
<td>514</td>
</tr>
<tr>
<td>4.166.2.3</td>
<td>SetDown()</td>
<td>514</td>
</tr>
<tr>
<td>4.166.3</td>
<td>Property Documentation</td>
<td>515</td>
</tr>
<tr>
<td>4.166.3.1</td>
<td>Speed</td>
<td>515</td>
</tr>
<tr>
<td>4.167</td>
<td>RaftInteractable Class Reference</td>
<td>515</td>
</tr>
<tr>
<td>4.167.1</td>
<td>Detailed Description</td>
<td>516</td>
</tr>
<tr>
<td>4.167.2</td>
<td>Member Function Documentation</td>
<td>517</td>
</tr>
<tr>
<td>4.167.2.1</td>
<td>BoardRaft()</td>
<td>517</td>
</tr>
<tr>
<td>4.167.2.2</td>
<td>DisembarkRaft()</td>
<td>517</td>
</tr>
<tr>
<td>4.167.2.3</td>
<td>SetUp()</td>
<td>517</td>
</tr>
<tr>
<td>4.167.3</td>
<td>Property Documentation</td>
<td>517</td>
</tr>
<tr>
<td>4.167.3.1</td>
<td>Raft</td>
<td>517</td>
</tr>
<tr>
<td>4.168</td>
<td>RaftMovement Class Reference</td>
<td>518</td>
</tr>
<tr>
<td>4.168.1</td>
<td>Detailed Description</td>
<td>520</td>
</tr>
<tr>
<td>4.168.2</td>
<td>Member Function Documentation</td>
<td>520</td>
</tr>
<tr>
<td>4.168.2.1</td>
<td>Climb()</td>
<td>520</td>
</tr>
<tr>
<td>4.168.2.2</td>
<td>GetClimbHeight()</td>
<td>521</td>
</tr>
<tr>
<td>4.168.2.3</td>
<td>GetRaycastHeight()</td>
<td>521</td>
</tr>
<tr>
<td>4.168.2.4</td>
<td>Idle()</td>
<td>521</td>
</tr>
<tr>
<td>4.168.2.5</td>
<td>Jump()</td>
<td>521</td>
</tr>
<tr>
<td>4.168.2.6</td>
<td>Move()</td>
<td>521</td>
</tr>
<tr>
<td>4.168.2.7</td>
<td>OnStateEnter()</td>
<td>522</td>
</tr>
<tr>
<td>4.168.2.8</td>
<td>OnStateExit()</td>
<td>522</td>
</tr>
</tbody>
</table>
4.168.2.9 SetMaxSpeed() ............................................. 522

4.168.3 Member Data Documentation .................................. 522
  4.168.3.1 BoardRaftText ........................................... 523
  4.168.3.2 DisembarkRaftText ...................................... 523
  4.168.3.3 IsActive .................................................. 523
  4.168.3.4 PlayerAnimator .......................................... 523

4.168.4 Property Documentation ...................................... 523
  4.168.4.1 PlayerStandHeight ...................................... 523

4.169 RaftWake Class Reference ..................................... 524
  4.169.1 Detailed Description ..................................... 524

4.170 RainController Class Reference ................................. 525
  4.170.1 Detailed Description ..................................... 526
  4.170.2 Member Data Documentation ................................ 526
    4.170.2.1 FogStartThreshold .................................. 526
    4.170.2.2 MainRain ............................................. 526
    4.170.2.3 RainFog ............................................... 526
    4.170.2.4 UseCustomValues ...................................... 527

  4.170.3 Property Documentation .................................. 527
    4.170.3.1 RainLevel ............................................ 527
    4.170.3.2 WindVectorXZ ....................................... 527

4.171 RandomUtility Class Reference ................................ 527
  4.171.1 Detailed Description ..................................... 528
  4.171.2 Member Function Documentation ............................ 528
    4.171.2.1 RandomVector2d() .................................. 528
    4.171.2.2 RandomVector3d() .................................. 529

  4.171.3 Property Documentation .................................. 529
    4.171.3.1 RandomBinomial ...................................... 529
    4.171.3.2 RandomBool .......................................... 529
    4.171.3.3 RandomHundredPercent ................................. 530
    4.171.3.4 RandomPercent ....................................... 530
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.177 RecipeYamlSerializer Class Reference</td>
<td>549</td>
</tr>
<tr>
<td>4.177.1 Detailed Description</td>
<td>550</td>
</tr>
<tr>
<td>4.177.2 Constructor &amp; Destructor Documentation</td>
<td>550</td>
</tr>
<tr>
<td>4.177.2.1 RecipeYamlSerializer()</td>
<td>550</td>
</tr>
<tr>
<td>4.177.3 Member Function Documentation</td>
<td>550</td>
</tr>
<tr>
<td>4.177.3.1 LoadRecipes()</td>
<td>550</td>
</tr>
<tr>
<td>4.178 Regression Class Reference</td>
<td>551</td>
</tr>
<tr>
<td>4.178.1 Detailed Description</td>
<td>551</td>
</tr>
<tr>
<td>4.178.2 Member Function Documentation</td>
<td>551</td>
</tr>
<tr>
<td>4.178.2.1 Prediction()</td>
<td>551</td>
</tr>
<tr>
<td>4.179 Requirement Class Reference</td>
<td>552</td>
</tr>
<tr>
<td>4.179.1 Detailed Description</td>
<td>553</td>
</tr>
<tr>
<td>4.179.2 Constructor &amp; Destructor Documentation</td>
<td>553</td>
</tr>
<tr>
<td>4.179.2.1 Requirement()</td>
<td>553</td>
</tr>
<tr>
<td>4.179.3 Member Function Documentation</td>
<td>553</td>
</tr>
<tr>
<td>4.179.3.1 isFullfilled()</td>
<td>553</td>
</tr>
<tr>
<td>4.179.3.2 SubmitItem()</td>
<td>553</td>
</tr>
<tr>
<td>4.179.4 Property Documentation</td>
<td>554</td>
</tr>
<tr>
<td>4.179.4.1 AmountRequired</td>
<td>554</td>
</tr>
<tr>
<td>4.179.4.2 ItemType</td>
<td>554</td>
</tr>
<tr>
<td>4.180 RestartBehavior Class Reference</td>
<td>554</td>
</tr>
<tr>
<td>4.180.1 Detailed Description</td>
<td>555</td>
</tr>
<tr>
<td>4.180.2 Member Function Documentation</td>
<td>555</td>
</tr>
<tr>
<td>4.180.2.1 Restart()</td>
<td>555</td>
</tr>
<tr>
<td>4.181 RooftopGeneration Class Reference</td>
<td>556</td>
</tr>
<tr>
<td>4.181.1 Detailed Description</td>
<td>558</td>
</tr>
<tr>
<td>4.181.2 Member Function Documentation</td>
<td>558</td>
</tr>
<tr>
<td>4.181.2.1 AddDoorsToDistrict()</td>
<td>558</td>
</tr>
<tr>
<td>4.181.2.2 AddSheltersToDistrict()</td>
<td>558</td>
</tr>
<tr>
<td>4.181.2.3 PopulateRoof()</td>
<td>559</td>
</tr>
</tbody>
</table>
CONTENTS

4.186.1 Detailed Description ................................................. 571

4.187 SettingsMenuBehavior Class Reference ................................. 572

  4.187.1 Detailed Description .................................................. 573

  4.187.2 Member Function Documentation ..................................... 573

    4.187.2.1 changeVolume() .................................................. 573

    4.187.2.2 OnConfigKeysClick() ............................................ 573

    4.187.2.3 OnSeedValueChanged() ........................................... 573

    4.187.2.4 turnSoundOnOff() ............................................... 574

4.188 ShelterCategory Class Reference ....................................... 574

  4.188.1 Detailed Description .................................................. 576

  4.188.2 Member Function Documentation ..................................... 576

    4.188.2.1 GetDuplicate() .................................................. 576

    4.188.2.2 ReadyCategory() ................................................ 576

    4.188.2.3 SetDown() ....................................................... 576

  4.188.3 Property Documentation .............................................. 577

    4.188.3.1 WarmthRate ..................................................... 577

4.189 ShelterInteractable Class Reference .................................... 577

  4.189.1 Detailed Description .................................................. 578

  4.189.2 Member Function Documentation ..................................... 579

    4.189.2.1 OnTriggerEnter() ............................................... 579

    4.189.2.2 OnTriggerExit() ................................................ 579

    4.189.2.3 SetUp() ....................................................... 579

  4.189.3 Property Documentation .............................................. 579

    4.189.3.1 Shelter ....................................................... 579

4.190 SicknessNotification Class Reference ................................. 580

  4.190.1 Detailed Description .................................................. 580

  4.190.2 Member Function Documentation ..................................... 581

    4.190.2.1 CloseNotification() ............................................ 581

4.191 SolidCategory Class Reference ......................................... 581

  4.191.1 Detailed Description .................................................. 584

Generated by Doxygen
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.191.2</td>
<td>Member Function Documentation</td>
<td>584</td>
</tr>
<tr>
<td>4.191.2.1</td>
<td>GetDuplicate()</td>
<td>584</td>
</tr>
<tr>
<td>4.191.2.2</td>
<td>ReadyCategory()</td>
<td>585</td>
</tr>
<tr>
<td>4.191.2.3</td>
<td>Sharpen()</td>
<td>585</td>
</tr>
<tr>
<td>4.191.2.4</td>
<td>WeaveBasket()</td>
<td>585</td>
</tr>
<tr>
<td>4.191.2.5</td>
<td>WeaveRope()</td>
<td>585</td>
</tr>
<tr>
<td>4.191.3</td>
<td>Property Documentation</td>
<td>585</td>
</tr>
<tr>
<td>4.191.3.1</td>
<td>Durability</td>
<td>586</td>
</tr>
<tr>
<td>4.191.3.2</td>
<td>Elasticity</td>
<td>586</td>
</tr>
<tr>
<td>4.191.3.3</td>
<td>Flexibility</td>
<td>586</td>
</tr>
<tr>
<td>4.191.3.4</td>
<td>Sharpness</td>
<td>586</td>
</tr>
<tr>
<td>4.191.3.5</td>
<td>Stickiness</td>
<td>587</td>
</tr>
<tr>
<td>4.191.3.6</td>
<td>Thickness</td>
<td>587</td>
</tr>
<tr>
<td>4.192</td>
<td>SplashScreenBehavior Class Reference</td>
<td>587</td>
</tr>
<tr>
<td>4.192.1</td>
<td>Detailed Description</td>
<td>588</td>
</tr>
<tr>
<td>4.193</td>
<td>SpriteManager Class Reference</td>
<td>588</td>
</tr>
<tr>
<td>4.193.1</td>
<td>Detailed Description</td>
<td>589</td>
</tr>
<tr>
<td>4.193.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>589</td>
</tr>
<tr>
<td>4.193.2.1</td>
<td>SpriteManager()</td>
<td>589</td>
</tr>
<tr>
<td>4.193.3</td>
<td>Member Function Documentation</td>
<td>589</td>
</tr>
<tr>
<td>4.193.3.1</td>
<td>GetSprite()</td>
<td>589</td>
</tr>
<tr>
<td>4.194</td>
<td>StatRate Class Reference</td>
<td>590</td>
</tr>
<tr>
<td>4.194.1</td>
<td>Detailed Description</td>
<td>591</td>
</tr>
<tr>
<td>4.194.2</td>
<td>Constructor &amp; Destructor Documentation</td>
<td>591</td>
</tr>
<tr>
<td>4.194.2.1</td>
<td>StatRate()</td>
<td>591</td>
</tr>
<tr>
<td>4.194.2.2</td>
<td>StatRate()</td>
<td>591</td>
</tr>
<tr>
<td>4.194.3</td>
<td>Member Function Documentation</td>
<td>591</td>
</tr>
<tr>
<td>4.194.3.1</td>
<td>ApplyRateToStat()</td>
<td>592</td>
</tr>
<tr>
<td>4.194.3.2</td>
<td>ChangeRateValues()</td>
<td>592</td>
</tr>
<tr>
<td>4.194.4</td>
<td>Property Documentation</td>
<td>592</td>
</tr>
</tbody>
</table>
4.194.1 CurrentStat .................................................. 592
4.194.2 MaxStat ..................................................... 592
4.194.3 PerSeconds ................................................ 593
4.194.4 Units ......................................................... 593

4.195 TeleportPlayer.TeleportLocation Struct Reference ........................................... 593
4.195.1 Detailed Description ........................................ 594
4.195.2 Member Data Documentation ............................ 594
  4.195.2.1 CanTeleportTo ........................................ 594
4.195.3 Property Documentation .................................. 594
  4.195.3.1 Location .............................................. 594

4.196 TeleportPlayer Class Reference .................................. 595
4.196.1 Detailed Description ........................................ 596
4.196.2 Member Function Documentation ........................ 596
  4.196.2.1 TeleportToA() ......................................... 596
  4.196.2.2 TeleportToB() ......................................... 596

4.197 TemplateBuilding Class Reference .................................. 597
4.197.1 Detailed Description ........................................ 598
4.197.2 Constructor & Destructor Documentation ............... 598
  4.197.2.1 TemplateBuilding() ................................... 599
4.197.3 Member Function Documentation ........................ 599
  4.197.3.1 Load() ............................................. 599
4.197.4 Property Documentation .................................. 599
  4.197.4.1 Template ............................................ 599

4.198 TestGUI Class Reference ......................................... 600
4.198.1 Detailed Description ........................................ 602
4.198.2 Member Function Documentation ........................ 602
  4.198.2.1 Music_Clicked() ...................................... 602
  4.198.2.2 ShowRadioButtons() .................................. 602
  4.198.2.3 Weather_Clicked() .................................... 602
4.198.3 Member Data Documentation ................................ 602
4.206.2.1 Voroni() ............................................. 620
4.206.3 Member Function Documentation ............................................. 621
  4.206.3.1 GetEdges() ............................................. 621
4.206.4 Property Documentation .................................................. 621
  4.206.4.1 Center .................................................. 621

4.207 WarmthIdolCategory Class Reference ............................................. 622
  4.207.1 Detailed Description .................................................. 624
  4.207.2 Member Function Documentation ............................................. 624
    4.207.2.1 ApplyBenefit() ............................................. 624
    4.207.2.2 GetDuplicate() ............................................. 624
    4.207.2.3 ReadyCategory() ............................................. 625
    4.207.2.4 RemoveBenefit() ............................................. 625
  4.207.3 Property Documentation .................................................. 625
    4.207.3.1 WarmthBenefit ............................................. 625

4.208 WarmthRateManager Class Reference ............................................. 626
  4.208.1 Detailed Description .................................................. 627
  4.208.2 Constructor & Destructor Documentation ............................................. 628
    4.208.2.1 WarmthRateManager() ............................................. 628
  4.208.3 Member Function Documentation ............................................. 628
    4.208.3.1 SetUnitsInShelter() ............................................. 628
    4.208.3.2 UseClothRate() ............................................. 628
    4.208.3.3 UseDefaultWarmthReductionRate() ............................................. 628
    4.208.3.4 UseHeatSourceWarmthIncreaseRate() ............................................. 629
    4.208.3.5 UseWaterWarmthReductionRate() ............................................. 629

4.209 WaterItemGeneration Class Reference ............................................. 629
  4.209.1 Detailed Description .................................................. 631
  4.209.2 Member Function Documentation ............................................. 631
    4.209.2.1 AddBuildingToWaterGenerationMap() ............................................. 631
    4.209.2.2 GenerateInWater() ............................................. 631
    4.209.2.3 SetCityInformation() ............................................. 631
4.215.1 Detailed Description .............................................. 653
4.215.2 Member Function Documentation .......................... 653
  4.215.2.1 findSpawnLocation() .................................. 654

4.216 WindMovement Class Reference .............................. 654
  4.216.1 Detailed Description ....................................... 655

4.217 WindowWasher Class Reference ............................... 655
  4.217.1 Detailed Description ....................................... 657
  4.217.2 Member Function Documentation ......................... 657
    4.217.2.1 Start() ............................................. 657
    4.217.2.2 TeleportPlayer() .................................. 658
    4.217.2.3 Update() .......................................... 658
    4.217.2.4 UpdateRopePosition() ................................ 658
  4.217.3 Member Data Documentation .............................. 658
    4.217.3.1 MaxLowerDistance .................................. 658
    4.217.3.2 StartUp ............................................. 658
  4.217.4 Property Documentation .................................. 659
    4.217.4.1 PlacementCenter ................................... 659
    4.217.4.2 WasherBaseLength ................................... 659

4.218 WorldItemFactory Class Reference ......................... 659
  4.218.1 Detailed Description ..................................... 660
  4.218.2 Constructor & Destructor Documentation ............... 660
    4.218.2.1 WorldItemFactory() ................................ 660
  4.218.3 Member Function Documentation ........................ 660
    4.218.3.1 CreateGenericInteractableItem() ................... 660
    4.218.3.2 CreatePickUpInteractableItem() .................... 661
    4.218.3.3 CreateRandomPickupInteractableItem() ........... 661
    4.218.3.4 GetAllInteractableItemsByDistrict() ............. 661
    4.218.3.5 GetRandomItemIndex() ............................... 662
    4.218.3.6 LoadTemplates() .................................... 662

4.219 WorldSelectionButtonBehavior Class Reference ........... 663
  4.219.1 Detailed Description ..................................... 664
  4.219.2 Member Function Documentation ........................ 664
    4.219.2.1 SetAction() ........................................ 664
  4.219.3 Property Documentation .................................. 664
    4.219.3.1 ButtonName ......................................... 664

4.220 WorldSelectionGUIDirector Class Reference ............... 665
  4.220.1 Detailed Description ..................................... 666
  4.220.2 Member Function Documentation ........................ 666
    4.220.2.1 ClearOptions() ..................................... 666
    4.220.2.2 CloseSelection() ................................... 666
    4.220.2.3 DisplayActions() ................................... 667
    4.220.2.4 DisplayItemOptions() ................................ 667
    4.220.2.5 FinishSelection() ................................... 667
5 File Documentation

5.1 Assets/Scripts/AmbientSoundManager.cs File Reference .......................... 669
5.2 Assets/Scripts/Camera/CameraController.cs File Reference ....................... 669
5.3 Assets/Scripts/City/Block.cs File Reference ........................................ 669
5.4 Assets/Scripts/City/BlockGenerator.cs File Reference ......................... 669
5.5 Assets/Scripts/City/Building.cs File Reference .................................. 669
5.6 Assets/Scripts/City/BuildingGenerator.cs File Reference ..................... 670
5.7 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingAttachment.cs File Reference 670
5.8 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingBase.cs File Reference 670
5.9 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingCreator.cs File Reference 670
5.9.1 Enumeration Type Documentation ..................................................... 670
5.9.1.1 BaseSize ................................................................................. 670
5.9.1.2 BuildingParts ........................................................................... 671
5.9.1.3 HeightType .............................................................................. 671
5.10 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingInstance.cs File Reference 671
5.11 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingRoof.cs File Reference 672
5.12 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingWindow.cs File Reference 672
5.13 Assets/Scripts/City/City.cs File Reference .......................................... 672
5.14 Assets/Scripts/City/CityBoundaries.cs File Reference ......................... 672
5.15 Assets/Scripts/City/CityController.cs File Reference ......................... 672
5.16 Assets/Scripts/City/DebugCity.cs File Reference ................................. 672
5.17 Assets/Scripts/City/District.cs File Reference .................................... 673
5.18 Assets/Scripts/City/DistrictDefinition.cs File Reference ...................... 673
5.19 Assets/Scripts/City/DistrictGenerator.cs File Reference ...................... 673
5.20 Assets/Scripts/City/DistrictPopupAttribute.cs File Reference ............... 673
5.21 Assets/Scripts/City/GenerationUtility.cs File Reference ...................... 673
5.22 Assets/Scripts/City/Management/CityChunk.cs File Reference ............... 673
5.23 Assets/Scripts/City/Management/CityChunkManager.cs File Reference .... 674
5.24 Assets/Scripts/City/ProceduralBuilding.cs File Reference .................... 674
5.25 Assets/Scripts/City/TemplateBuilding.cs File Reference ....................... 674
5.26 Assets/Scripts/City/Voroni.cs File Reference ................................................. 674
5.27 Assets/Scripts/Creatures/CreatureManager.cs File Reference .......................... 674
5.28 Assets/Scripts/Creatures/CreatureSpawner.cs File Reference .......................... 674
5.29 Assets/Scripts/Creatures/CreatureTracker.cs File Reference .......................... 675
5.30 Assets/Scripts/Creatures/FishSpawner.cs File Reference ............................... 675
5.31 Assets/Scripts/Creatures/JellyFishSpawner.cs File Reference ........................... 675
5.32 Assets/Scripts/Creatures/UI/SetCreatureGUI.cs File Reference ......................... 675
5.33 Assets/Scripts/Creatures/WhaleSpawner.cs File Reference ............................. 675
5.34 Assets/Scripts/DeathScreen/DeathFlavorText.cs File Reference ........................ 675
5.35 Assets/Scripts/DeathScreen/DeathManager.cs File Reference ........................... 675
5.36 Assets/Scripts/DeathScreen/RestartBehavior.cs File Reference .......................... 676
5.37 Assets/Scripts/Editor/BaseItemPopup.cs File Reference .................................. 676
5.38 Assets/Scripts/Editor/DistrictPopup.cs File Reference ................................... 676
5.39 Assets/Scripts/Editor/uGUITools.cs File Reference ...................................... 676
5.40 Assets/Scripts/Effects/DayNightSkybox.cs File Reference ............................... 676
5.41 Assets/Scripts/Effects/DebrisController.cs File Reference .............................. 676
5.42 Assets/Scripts/Effects/FXSplash.cs File Reference ........................................ 677
5.43 Assets/Scripts/Effects/FXSplashManager.cs File Reference ............................. 677
5.44 Assets/Scripts/Effects/RainController.cs File Reference ................................ 677
5.45 Assets/Scripts/Ending/BeaconInteractable.cs File Reference ......................... 677
5.46 Assets/Scripts/Ending/EndingController.cs File Reference ............................ 677
5.47 Assets/Scripts/Ending/EndingTrigger.cs File Reference ................................ 677
5.48 Assets/Scripts/Events/EventManager.cs File Reference .................................. 677
5.49 Assets/Scripts/Game.cs File Reference ....................................................... 678
5.50 Assets/Scripts/Graffiti/TutorialGraffiti.cs File Reference .............................. 678
5.51 Assets/Scripts/GUI/CraftingGUI/IngredientButtonBehavior.cs File Reference .... 678
5.52 Assets/Scripts/GUI/CraftingGUI/RecipeBookBehavior.cs File Reference ............. 678
5.53 Assets/Scripts/GUI/CraftingGUI/RecipeButtonGUIBehavior.cs File Reference .... 678
5.54 Assets/Scripts/GUI/CraftingGUI/RecipePageBehavior.cs File Reference ............. 678
5.55 Assets/Scripts/GUI/CraftingGUI/RecipeRequirementsUI.cs File Reference ........... 678

Generated by Doxygen
CONTENTS

5.56 Assets/Scripts/GUI/CraftingGUI/SelectedIngredientButton.cs File Reference .................. 679
5.57 Assets/Scripts/GUI/GameViewScene/GameViewBehavior.cs File Reference .................. 679
5.58 Assets/Scripts/GUI/GameViewScene/HUDBehavior.cs File Reference .................. 679
5.59 Assets/Scripts/GUI/GameViewScene/SicknessNotification.cs File Reference ............... 679
5.60 Assets/Scripts/GUI/InstanceManagerGUI.cs File Reference ................................. 679
5.61 Assets/Scripts/GUI/InventoryGUI/ChooseItemAmountPanelBehavior.cs File Reference .... 679
5.62 Assets/Scripts/GUI/InventoryGUI/InventoryUI.cs File Reference ............................. 680
5.63 Assets/Scripts/GUI/InventoryGUI/ItemActionButtonUI.cs File Reference .................. 680
5.64 Assets/Scripts/GUI/InventoryGUI/ItemAttributeUI.cs File Reference ...................... 680
5.65 Assets/Scripts/GUI/InventoryGUI/ItemStackDetailPanelBehavior.cs File Reference ...... 680
5.66 Assets/Scripts/GUI/InventoryGUI/ItemStackUI.cs File Reference ............................ 680
5.67 Assets/Scripts/GUI/MainMenuScene/EndCreditsBehavior.cs File Reference ................. 680
5.68 Assets/Scripts/GUI/MainMenuScene/KeyCodeConfigMenuBehavior.cs File Reference .... 680
5.69 Assets/Scripts/GUI/MainMenuScene/MainMenuBehavior.cs File Reference ................. 681
5.70 Assets/Scripts/GUI/MainMenuScene/SettingsMenuBehavior.cs File Reference ............... 681
5.71 Assets/Scripts/GUI/MainMenuScene/SplashScreenBehavior.cs File Reference ............... 681
5.72 Assets/Scripts/GUI/Utility/FaderManager.cs File Reference .................................. 681
5.73 Assets/Scripts/GUI/Utility/GridLayoutManager.cs File Reference ........................... 681
5.74 Assets/Scripts/GUI/Utility/SpriteManager.cs File Reference .................................. 681
5.75 Assets/Scripts/GUI/WorldInteraction/EquippedItemDropdown.cs File Reference ........... 681
5.76 Assets/Scripts/GUI/WorldInteraction/WorldSelectionButtonBehavior.cs File Reference .... 682
5.77 Assets/Scripts/GUI/WorldInteraction/WorldSelectionGUIDirector.cs File Reference ...... 682
5.78 Assets/Scripts/Item Crafting/Backend/Attribute.cs File Reference ............................ 682
5.79 Assets/Scripts/Item Crafting/Backend/BaseItem.cs File Reference ............................ 682
5.80 Assets/Scripts/Item Crafting/Backend/ClothCategory.cs File Reference ....................... 682
5.81 Assets/Scripts/Item Crafting/Backend/CollectableItem.cs File Reference .................... 682
5.82 Assets/Scripts/Item Crafting/Backend/ContainerCategory.cs File Reference ............... 683
5.83 Assets/Scripts/Item Crafting/Backend/CraftingStat.cs File Reference ........................ 683
5.84 Assets/Scripts/Item Crafting/Backend/EquipableCategory.cs File Reference ................. 683
5.85 Assets/Scripts/Item Crafting/Backend/FireBaseCategory.cs File Reference .................... 683
<p>| 5.116 | Assets/Scripts/ItemProceduralGeneration/ItemPoolInfo.cs File Reference | 688 |
| 5.117 | Assets/Scripts/ItemProceduralGeneration/ItemPoolManager.cs File Reference | 688 |
| 5.118 | Assets/Scripts/ItemProceduralGeneration/ItemSpawner.cs File Reference | 688 |
| 5.119 | Assets/Scripts/ItemProceduralGeneration/RooftopGeneration.cs File Reference | 688 |
| 5.120 | Assets/Scripts/ItemProceduralGeneration/RooftopPointGenerator.cs File Reference | 689 |
| 5.121 | Assets/Scripts/ItemProceduralGeneration/SamplingPointGenerator.cs File Reference | 689 |
| 5.122 | Assets/Scripts/ItemProceduralGeneration/WaterItemGenerator.cs File Reference | 689 |
| 5.123 | Assets/Scripts/ItemProceduralGeneration/WaterPointGenerator.cs File Reference | 689 |
| 5.124 | Assets/Scripts/ItemProceduralGeneration/WeightedPair.cs File Reference | 689 |
| 5.125 | Assets/Scripts/ItemProceduralGeneration/WorldItemFactory.cs File Reference | 689 |
| 5.126 | Assets/Scripts/Items/Barometer.cs File Reference | 690 |
| 5.127 | Assets/Scripts/Items/FireInteractable.cs File Reference | 690 |
| 5.128 | Assets/Scripts/Items/FishingLure.cs File Reference | 690 |
| 5.129 | Assets/Scripts/Items/FishingRod.cs File Reference | 690 |
| 5.130 | Assets/Scripts/Items/FloatBehavior.cs File Reference | 690 |
| 5.131 | Assets/Scripts/Items/HasSelectionInterface.cs File Reference | 690 |
| 5.132 | Assets/Scripts/Items/Idol.cs File Reference | 690 |
| 5.133 | Assets/Scripts/Items/InteractableObject.cs File Reference | 691 |
| 5.134 | Assets/Scripts/Items/InteractableSetUp.cs File Reference | 691 |
| 5.135 | Assets/Scripts/Items/InventoryInteractable.cs File Reference | 691 |
| 5.136 | Assets/Scripts/Items/LightItem.cs File Reference | 691 |
| 5.137 | Assets/Scripts/Items/OverworldItemOptionSelection.cs File Reference | 691 |
| 5.138 | Assets/Scripts/Items/PickUpItem.cs File Reference | 691 |
| 5.139 | Assets/Scripts/Items/RaftInteractable.cs File Reference | 691 |
| 5.140 | Assets/Scripts/Items/ShelterInteractable.cs File Reference | 692 |
| 5.141 | Assets/Scripts/Items/Thermometer.cs File Reference | 692 |
| 5.142 | Assets/Scripts/Items/Tool.cs File Reference | 692 |
| 5.143 | Assets/Scripts/Loading/GameLoader.cs File Reference | 692 |
| 5.144 | Assets/Scripts/Loading/GameLoaderTask.cs File Reference | 692 |
| 5.145 | Assets/Scripts/Loading&gt;LoadingScreen.cs File Reference | 692 |</p>
<table>
<thead>
<tr>
<th>5.172.1.1 AnnouncementCategory</th>
<th>697</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.173 Assets/Scripts/Radio/Announcements/AnnouncementFactory.cs File Reference</td>
<td>698</td>
</tr>
<tr>
<td>5.174 Assets/Scripts/Radio/Announcements/MysteryAnnouncement.cs File Reference</td>
<td>698</td>
</tr>
<tr>
<td>5.175 Assets/Scripts/Radio/Dial.cs File Reference</td>
<td>698</td>
</tr>
<tr>
<td>5.177 Assets/Scripts/Radio/Radio.cs File Reference</td>
<td>698</td>
</tr>
<tr>
<td><strong>5.177.1 Enumeration Type Documentation</strong></td>
<td>699</td>
</tr>
<tr>
<td><strong>5.177.1.1 RadioChannel</strong></td>
<td>699</td>
</tr>
<tr>
<td><strong>5.178.1 Enumeration Type Documentation</strong></td>
<td>699</td>
</tr>
<tr>
<td><strong>5.178.1.1 ButtonType</strong></td>
<td>699</td>
</tr>
<tr>
<td>5.179 Assets/Scripts/Radio/TestGUI.cs File Reference</td>
<td>700</td>
</tr>
<tr>
<td>5.180 Assets/Scripts/Teleporters/TeleportPlayer.cs File Reference</td>
<td>700</td>
</tr>
<tr>
<td>5.181 Assets/Scripts/Teleporters/WindowWasher.cs File Reference</td>
<td>700</td>
</tr>
<tr>
<td>5.182 Assets/Scripts/Utility/BezierLine.cs File Reference</td>
<td>700</td>
</tr>
<tr>
<td>5.183 Assets/Scripts/Utility/BooleanOperator.cs File Reference</td>
<td>700</td>
</tr>
<tr>
<td>5.184 Assets/Scripts/Utility/ConsoleCommandRouter.cs File Reference</td>
<td>700</td>
</tr>
<tr>
<td>5.185 Assets/Scripts/Utility/File/FileManager.cs File Reference</td>
<td>701</td>
</tr>
<tr>
<td>5.186 Assets/Scripts/Utility/File/GoogleDrive.cs File Reference</td>
<td>701</td>
</tr>
<tr>
<td>5.187 Assets/Scripts/Utility/GameSettings.cs File Reference</td>
<td>701</td>
</tr>
<tr>
<td>5.188 Assets/Scripts/Utility/NullParent.cs File Reference</td>
<td>701</td>
</tr>
<tr>
<td>5.189 Assets/Scripts/Utility/PauseSystem.cs File Reference</td>
<td>701</td>
</tr>
<tr>
<td>5.191 Assets/Scripts/Utility/Regression.cs File Reference</td>
<td>701</td>
</tr>
<tr>
<td>5.192 Assets/Scripts/Utility/SystemLogger.cs File Reference</td>
<td>702</td>
</tr>
<tr>
<td>5.194 Assets/Scripts/Utility/Tuple.cs File Reference</td>
<td>702</td>
</tr>
<tr>
<td>5.196 Assets/Scripts/Weather/Clock.cs File Reference</td>
<td>702</td>
</tr>
<tr>
<td>5.197 Assets/Scripts/Weather/DayNight.cs File Reference</td>
<td>702</td>
</tr>
<tr>
<td>Section</td>
<td>File Name</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>5.198</td>
<td>Assets/Scripts/Weather/DiurnalTemperatureVariance.cs</td>
</tr>
<tr>
<td>5.199</td>
<td>Assets/Scripts/Weather/Effects/LightningEffect.cs</td>
</tr>
<tr>
<td>5.200</td>
<td>Assets/Scripts/Weather/FloodWater.cs</td>
</tr>
<tr>
<td>5.201</td>
<td>Assets/Scripts/Weather/GUI/PressureSystemVisualization.cs</td>
</tr>
<tr>
<td>5.202</td>
<td>Assets/Scripts/Weather/GUI/SetEnvironmentGUI.cs</td>
</tr>
<tr>
<td>5.203</td>
<td>Assets/Scripts/Weather/Lightning.cs</td>
</tr>
<tr>
<td>5.204</td>
<td>Assets/Scripts/Weather/PressureSystem.cs</td>
</tr>
<tr>
<td>5.205</td>
<td>Assets/Scripts/Weather/PressureSystems.cs</td>
</tr>
<tr>
<td>5.206</td>
<td>Assets/Scripts/Weather/WeatherSoundSystem.cs</td>
</tr>
<tr>
<td>5.207</td>
<td>Assets/Scripts/Weather/WeatherSystem.cs</td>
</tr>
<tr>
<td>5.208</td>
<td>Assets/Scripts/YamlParsing/AnnouncementYAMLParser.cs</td>
</tr>
<tr>
<td>5.209</td>
<td>Assets/Scripts/YamlParsing/CraftingSystemSerializer.cs</td>
</tr>
<tr>
<td>5.210</td>
<td>Assets/Scripts/YamlParsing/InventoryItemYAMLModel.cs</td>
</tr>
<tr>
<td>5.211</td>
<td>Assets/Scripts/YamlParsing/InventoryYAMLModel.cs</td>
</tr>
<tr>
<td>5.212</td>
<td>Assets/Scripts/YamlParsing/InventoryYamlParser.cs</td>
</tr>
<tr>
<td>5.213</td>
<td>Assets/Scripts/YamlParsing/ItemDistrictModel.cs</td>
</tr>
<tr>
<td>5.214</td>
<td>Assets/Scripts/YamlParsing/ItemSerializer.cs</td>
</tr>
<tr>
<td>5.215</td>
<td>Assets/Scripts/YamlParsing/ItemYAMLMap.cs</td>
</tr>
<tr>
<td>5.216</td>
<td>Assets/Scripts/YamlParsing/LoadingScreenFlavorTextYAMLParser.cs</td>
</tr>
<tr>
<td>5.217</td>
<td>Assets/Scripts/YamlParsing/NoteYAMLParser.cs</td>
</tr>
<tr>
<td>5.218</td>
<td>Assets/Scripts/YamlParsing/RecipeYamlSerializer.cs</td>
</tr>
</tbody>
</table>
Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

- AnnouncementFactory .......................................................... 20
- AnnouncementYAMLParser ...................................................... 23
- Attribute .............................................................................. 24
- Block ................................................................................... 50
- BooleanOperator .................................................................... 54
- Building ............................................................................... 58
  - ProceduralBuilding ............................................................ 482
  - TemplateBuilding .............................................................. 597
- buildingIndex ....................................................................... 66
- DistrictConfiguration.BuildingTemplatePlacement .................... 67
- City ..................................................................................... 75
- CityBoundaries ..................................................................... 78
- CityChunk ............................................................................ 81
- CollectableItem .................................................................... 100
- BaseItem .............................................................................. 29
- ItemCategory ......................................................................... 316
- ClothCategory ........................................................................ 95
- ContainerCategory .................................................................. 106
- EquipableCategory .................................................................. 165
  - FishingRodCategory ............................................................. 209
  - IdolCategory ....................................................................... 268
  - WarmthIdolCategory ............................................................ 622
- LightCategory ........................................................................ 391
- FireBaseCategory .................................................................... 182
- FleshCategory ........................................................................ 214
- FuelCategory ......................................................................... 221
- MedicineCategory .................................................................... 410
- PlantCategory ......................................................................... 444
- RaftCategory .......................................................................... 512
- ShelterCategory ...................................................................... 574
- SolidCategory ......................................................................... 581
- ControlScheme ...................................................................... 109
- CraftingStat .......................................................................... 114
- CraftingSystemSerializer ......................................................... 115
Index

<table>
<thead>
<tr>
<th>Class</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>InventoryYAMLParser</td>
<td>303</td>
</tr>
<tr>
<td>ItemSerializer</td>
<td>351</td>
</tr>
<tr>
<td>RecipeYAMLSerializer</td>
<td>549</td>
</tr>
<tr>
<td>CreatureManager</td>
<td>118</td>
</tr>
<tr>
<td>DeathManager</td>
<td>130</td>
</tr>
<tr>
<td>District</td>
<td>136</td>
</tr>
<tr>
<td>DistrictConfiguration</td>
<td>139</td>
</tr>
<tr>
<td>DistrictItemConfiguration</td>
<td>148</td>
</tr>
<tr>
<td>DistrictItemRarityConfiguration</td>
<td>151</td>
</tr>
<tr>
<td>DiurnalTemperatureVariance</td>
<td>156</td>
</tr>
<tr>
<td>EventManager</td>
<td>173</td>
</tr>
<tr>
<td>FileManager</td>
<td>179</td>
</tr>
<tr>
<td>Game</td>
<td>229</td>
</tr>
<tr>
<td>GameLoader</td>
<td>236</td>
</tr>
<tr>
<td>GameLoaderTask</td>
<td>238</td>
</tr>
<tr>
<td>GameSettings</td>
<td>240</td>
</tr>
<tr>
<td>GenerationUtility</td>
<td>245</td>
</tr>
<tr>
<td>GoogleDrive</td>
<td>250</td>
</tr>
<tr>
<td>HasSelectionInterface</td>
<td>254</td>
</tr>
<tr>
<td>IEnumerator</td>
<td>61</td>
</tr>
<tr>
<td>City.BuildingEnumerator</td>
<td>271</td>
</tr>
<tr>
<td>Ingredient</td>
<td>284</td>
</tr>
<tr>
<td>PlayerInventory</td>
<td>467</td>
</tr>
<tr>
<td>InventoryYAMLModel</td>
<td>295</td>
</tr>
<tr>
<td>InventoryYAMLModel</td>
<td>301</td>
</tr>
<tr>
<td>ItemAction</td>
<td>306</td>
</tr>
<tr>
<td>ItemCondition</td>
<td>321</td>
</tr>
<tr>
<td>ItemDiscarder</td>
<td>325</td>
</tr>
<tr>
<td>ItemDistrictModel</td>
<td>326</td>
</tr>
<tr>
<td>ItemFactory</td>
<td>327</td>
</tr>
<tr>
<td>ItemPlacementSamplePoint</td>
<td>337</td>
</tr>
<tr>
<td>ItemPoolInfo</td>
<td>341</td>
</tr>
<tr>
<td>ItemRarity</td>
<td>349</td>
</tr>
<tr>
<td>ItemStack</td>
<td>358</td>
</tr>
<tr>
<td>ItemTypes</td>
<td>374</td>
</tr>
<tr>
<td>ItemYAMLMap</td>
<td>379</td>
</tr>
<tr>
<td>LoadingScreenFlavorText</td>
<td>403</td>
</tr>
<tr>
<td>LoadingScreenFlavorTextManager</td>
<td>404</td>
</tr>
<tr>
<td>LoadingScreenFlavorTextYAMLParser</td>
<td>405</td>
</tr>
<tr>
<td>MonoBehaviour</td>
<td></td>
</tr>
<tr>
<td>AmbientSoundManager</td>
<td>19</td>
</tr>
<tr>
<td>BeaconInteractable</td>
<td>46</td>
</tr>
<tr>
<td>BezierLine</td>
<td>48</td>
</tr>
<tr>
<td>BlockGenerator</td>
<td>53</td>
</tr>
<tr>
<td>BuildingGenerator</td>
<td>64</td>
</tr>
<tr>
<td>CameraController</td>
<td>68</td>
</tr>
<tr>
<td>ChooseItemAmountPanelBehavior</td>
<td>71</td>
</tr>
<tr>
<td>CityChunkManager</td>
<td>84</td>
</tr>
<tr>
<td>CityController</td>
<td>88</td>
</tr>
<tr>
<td>Clock</td>
<td>89</td>
</tr>
<tr>
<td>ConsoleCommandRouter</td>
<td>101</td>
</tr>
<tr>
<td>CreatureSpawner</td>
<td>121</td>
</tr>
<tr>
<td>FishSpawner</td>
<td>212</td>
</tr>
<tr>
<td>JellyFishSpawner</td>
<td>381</td>
</tr>
<tr>
<td>WhaleSpawner</td>
<td>652</td>
</tr>
<tr>
<td>CreatureTracker</td>
<td>125</td>
</tr>
<tr>
<td>DayNight</td>
<td>127</td>
</tr>
</tbody>
</table>

Generated by Doxygen
<table>
<thead>
<tr>
<th>Class</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DayNightSkybox</td>
<td>128</td>
</tr>
<tr>
<td>DeathFlavorText</td>
<td>129</td>
</tr>
<tr>
<td>DebrisController</td>
<td>131</td>
</tr>
<tr>
<td>DebugCity</td>
<td>134</td>
</tr>
<tr>
<td>Dial</td>
<td>135</td>
</tr>
<tr>
<td>DistrictGenerator</td>
<td>146</td>
</tr>
<tr>
<td>DummyGenerator</td>
<td>157</td>
</tr>
<tr>
<td>DummyNoteGenerator</td>
<td>159</td>
</tr>
<tr>
<td>EndCreditsBehavior</td>
<td>160</td>
</tr>
<tr>
<td>EndingController</td>
<td>161</td>
</tr>
<tr>
<td>EndingTrigger</td>
<td>163</td>
</tr>
<tr>
<td>EquippedItemDropdown</td>
<td>169</td>
</tr>
<tr>
<td>FaderManager</td>
<td>177</td>
</tr>
<tr>
<td>FishAgent</td>
<td>191</td>
</tr>
<tr>
<td>FishAgentConfig</td>
<td>197</td>
</tr>
<tr>
<td>FishingLure</td>
<td>202</td>
</tr>
<tr>
<td>FloatBehavior</td>
<td>218</td>
</tr>
<tr>
<td>FloodWater</td>
<td>219</td>
</tr>
<tr>
<td>FXSplash</td>
<td>224</td>
</tr>
<tr>
<td>FXSplashManager</td>
<td>226</td>
</tr>
<tr>
<td>GameViewBehavior</td>
<td>242</td>
</tr>
<tr>
<td>GridLayoutManager</td>
<td>252</td>
</tr>
<tr>
<td>HUDBehavior</td>
<td>259</td>
</tr>
<tr>
<td>IngredientButtonBehavior</td>
<td>273</td>
</tr>
<tr>
<td>InteractableObject</td>
<td>276</td>
</tr>
<tr>
<td>FireInteractable</td>
<td>187</td>
</tr>
<tr>
<td>InventoryInteractable</td>
<td>293</td>
</tr>
<tr>
<td>ItemSpawner</td>
<td>354</td>
</tr>
<tr>
<td>Note</td>
<td>422</td>
</tr>
<tr>
<td>PickUpItem</td>
<td>441</td>
</tr>
<tr>
<td>RaftInteractable</td>
<td>515</td>
</tr>
<tr>
<td>ShelterInteractable</td>
<td>577</td>
</tr>
<tr>
<td>InteractableRadioModel</td>
<td>279</td>
</tr>
<tr>
<td>InteractableSetUp</td>
<td>283</td>
</tr>
<tr>
<td>InventoryUI</td>
<td>297</td>
</tr>
<tr>
<td>ItemActionButtonUI</td>
<td>310</td>
</tr>
<tr>
<td>ItemAttributeUI</td>
<td>313</td>
</tr>
<tr>
<td>ItemGenerator</td>
<td>333</td>
</tr>
<tr>
<td>RooftopGeneration</td>
<td>556</td>
</tr>
<tr>
<td>WaterItemGeneration</td>
<td>629</td>
</tr>
<tr>
<td>ItemPoolManager</td>
<td>344</td>
</tr>
<tr>
<td>ItemStackDetailPanelBehavior</td>
<td>361</td>
</tr>
<tr>
<td>ItemStackUI</td>
<td>367</td>
</tr>
<tr>
<td>KeyCodeConfigMenuBehavior</td>
<td>383</td>
</tr>
<tr>
<td>Lightning</td>
<td>400</td>
</tr>
<tr>
<td>LightningEffect</td>
<td>401</td>
</tr>
<tr>
<td>LoadingScreen</td>
<td>402</td>
</tr>
<tr>
<td>MainMenuBehavior</td>
<td>407</td>
</tr>
<tr>
<td>Movement</td>
<td>415</td>
</tr>
<tr>
<td>LandMovement</td>
<td>385</td>
</tr>
<tr>
<td>RaftMovement</td>
<td>518</td>
</tr>
<tr>
<td>WaterMovement</td>
<td>632</td>
</tr>
<tr>
<td>NoteUiController</td>
<td>429</td>
</tr>
<tr>
<td>NullParent</td>
<td>433</td>
</tr>
<tr>
<td>PlayerAnimationEvents</td>
<td>455</td>
</tr>
<tr>
<td>PlayerController</td>
<td>457</td>
</tr>
<tr>
<td>PlayerFocus</td>
<td>465</td>
</tr>
<tr>
<td>PressureSystemVisualization</td>
<td>481</td>
</tr>
<tr>
<td>Class Name</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>ProceduralBuildingAttachment</td>
<td>487</td>
</tr>
<tr>
<td>ProceduralBuildingBase</td>
<td>490</td>
</tr>
<tr>
<td>ProceduralBuildingCreator</td>
<td>495</td>
</tr>
<tr>
<td>ProceduralBuildingInstance</td>
<td>498</td>
</tr>
<tr>
<td>ProceduralBuildingRoof</td>
<td>502</td>
</tr>
<tr>
<td>ProceduralBuildingWindow</td>
<td>503</td>
</tr>
<tr>
<td>Radio</td>
<td>505</td>
</tr>
<tr>
<td>RadioButton</td>
<td>511</td>
</tr>
<tr>
<td>RaftWake</td>
<td>524</td>
</tr>
<tr>
<td>RainController</td>
<td>525</td>
</tr>
<tr>
<td>RecipeBookBehavior</td>
<td>534</td>
</tr>
<tr>
<td>RecipeButtonGUIBehavior</td>
<td>537</td>
</tr>
<tr>
<td>RecipePageBehavior</td>
<td>542</td>
</tr>
<tr>
<td>RecipeRequirementsUI</td>
<td>547</td>
</tr>
<tr>
<td>RestartBehavior</td>
<td>554</td>
</tr>
<tr>
<td>SelectedIngredientButton</td>
<td>566</td>
</tr>
<tr>
<td>SetCreatureGUI</td>
<td>570</td>
</tr>
<tr>
<td>SetEnvironmentGUI</td>
<td>571</td>
</tr>
<tr>
<td>SettingsMenuBehavior</td>
<td>572</td>
</tr>
<tr>
<td>SicknessNotification</td>
<td>580</td>
</tr>
<tr>
<td>SplashScreenBehavior</td>
<td>587</td>
</tr>
<tr>
<td>TeleportPlayer</td>
<td>595</td>
</tr>
<tr>
<td>TestGUI</td>
<td>600</td>
</tr>
<tr>
<td>Tool</td>
<td>607</td>
</tr>
<tr>
<td>Barometer</td>
<td>26</td>
</tr>
<tr>
<td>FishingRod</td>
<td>205</td>
</tr>
<tr>
<td>Idol</td>
<td>264</td>
</tr>
<tr>
<td>LightItem</td>
<td>396</td>
</tr>
<tr>
<td>Thermometer</td>
<td>604</td>
</tr>
<tr>
<td>TransformParent</td>
<td>611</td>
</tr>
<tr>
<td>TutorialGraffiti</td>
<td>614</td>
</tr>
<tr>
<td>uGUITools</td>
<td>616</td>
</tr>
<tr>
<td>WeatherSoundSystem</td>
<td>644</td>
</tr>
<tr>
<td>WindMovement</td>
<td>654</td>
</tr>
<tr>
<td>WindowWasher</td>
<td>655</td>
</tr>
<tr>
<td>WorldSelectionButtonBehavior</td>
<td>663</td>
</tr>
<tr>
<td>WorldSelectionGUIDirector</td>
<td>665</td>
</tr>
<tr>
<td>MysteryAnnouncement</td>
<td>419</td>
</tr>
<tr>
<td>NoteData</td>
<td>425</td>
</tr>
<tr>
<td>NoteFactory</td>
<td>427</td>
</tr>
<tr>
<td>NoteYAMLParser</td>
<td>432</td>
</tr>
<tr>
<td>OverworldItemOptionSelection</td>
<td>434</td>
</tr>
<tr>
<td>PauseSystem</td>
<td>438</td>
</tr>
<tr>
<td>Player</td>
<td>450</td>
</tr>
<tr>
<td>PlayerStatManager</td>
<td>472</td>
</tr>
<tr>
<td>PlayerTools</td>
<td>475</td>
</tr>
<tr>
<td>PressureSystem</td>
<td>477</td>
</tr>
<tr>
<td>PressureSystems</td>
<td>479</td>
</tr>
<tr>
<td>DistrictConfiguration.ProceduralBuildingBasesSizeXDefinition</td>
<td>493</td>
</tr>
<tr>
<td>PropertyAttribute</td>
<td></td>
</tr>
<tr>
<td>BaseltemPopupAttribute</td>
<td>45</td>
</tr>
<tr>
<td>DistrictPopupAttribute</td>
<td>155</td>
</tr>
<tr>
<td>PropertyDrawer</td>
<td></td>
</tr>
<tr>
<td>BaseltemPopup</td>
<td>44</td>
</tr>
<tr>
<td>DistrictPopup</td>
<td>153</td>
</tr>
<tr>
<td>RandomUtility</td>
<td>527</td>
</tr>
<tr>
<td>Recipe</td>
<td>530</td>
</tr>
<tr>
<td>Regression</td>
<td>551</td>
</tr>
</tbody>
</table>
1.1 Class Hierarchy

Requirement ................................................................. 552
SamplingPointGenerator ............................................. 562
  RooftopPointGenerator ........................................... 559
  WaterPointGenerator ............................................... 638
SpriteManager ............................................................ 588
StatRate ................................................................. 590
  HealthRateManager .................................................. 256
  HungerRateManager ................................................ 261
  WarmthRateManager ............................................... 626
TeleportPlayer.TeleportLocation .................................... 593
Tuple<T1, T2> .......................................................... 613
Tuple<int, int> .......................................................... 613
VectorUtility .......................................................... 617
Voroni ................................................................. 620
WeatherSystem ........................................................ 646
WeightedPair .......................................................... 650
WorldItemFactory ...................................................... 659
Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

- AmbientSoundManager ........................................... 19
- AnnouncementFactory ........................................... 20
- AnnouncementYAMLParser ........................................ 23
- Attribute ............................................................. 24
- Barometer ............................................................ 26
- Baseitem ............................................................. 29
  
  Class attached to all items. Handles getting information from the various Item Categories.

- BaseItemPopup ..................................................... 44
- BaseItemPopupAttribute .......................................... 45
- BeaconInteractable ............................................... 46
- BezierLine ........................................................... 48
- Block ................................................................. 50
- BlockGenerator .................................................... 53
- BooleanOperator .................................................. 54
- Building .............................................................. 58
- City, BuildingEnumerable ......................................... 61
  
  Class for creating an iterator to traverse the tree-structure of buildings in the City.

- BuildingGenerator ................................................ 64
- buildingIndex ....................................................... 66
- DistrictConfiguration, BuildingTemplatePlacement
  
  Structure for accessing additional building placement configuration.

- CameraController .................................................. 68
- ChooseltemAmountPanelBehavior
  
  Item amount affect panel behavior.

- City ................................................................. 71
- CityBoundaries .................................................... 75
- CityChunk ........................................................... 78
- CityChunkManager .................................................. 81
  
  Manages loading and unloading a chunk of buildings

- CityController ...................................................... 84
- Clock ................................................................. 88
- ClothCategory ....................................................... 95
- CollectableItem
  
  Abstract class that is inherited by all BaseItem classes and Item Categories
DayNightSkybox
This script modifies the Blend variable in the custom shader, Skybox/Blended based on the Game.Insance's clock.

DebrisController
This class handles particle effects for miscellaneous debris, such as sand, leaves, dirt, and paper, in relation to them blowing around in the wind during the game. This is structured very similarly to the RainController class, except that this only handles one particle system at a time, without blending.

DistrictConfiguration
This class holds all the relevant info needed to define a district. It holds things like the building pieces the district contains, that districts materials, window types, etc.

DiurnalTemperatureVariance
A gamified version of: https://en.wikipedia.org/wiki/Diurnal_temperature_variation

DummyGenerator
Can be called to fade the screen in or out
2.1 Class List

Game.......................................................... 229
GameLoader
Tracks all loading tasks and determines overall loading percentage. .................... 236
GameLoaderTask
Tracks information for a loading task. ........................................ 238
GameSettings ............................................. 240
GameViewBehavior ....................................... 242
GenerationUtility ........................................ 245
GoogleDrive ............................................... 250
GridLayoutManager ....................................... 252
HasSelectionInterface ..................................... 254
HealthRateManager ....................................... 256
HUDBehavior .............................................. 259
HungerRateManager ....................................... 261
Idol ............................................................. 264
IdolCategory ............................................... 268
Ingredient .................................................. 271
IngredientButtonBehavior .................................. 273
InteractableObject ........................................ 276
InteractableRadioModel ................................... 279
InteractableSetUp ......................................... 283
Inventory
Defines the inventory of items the player can access. Currently just considers all the gameobjects
parented under item parent as part of the inventory. TODO: Enable this to be used for on hand
and raft inventories. Add inventory size limit and ability to adjust the size. Make the inventory
save strings of objects then use YAML file to define those items as needed. .................. 284
InventoryInteractable ...................................... 293
InventoryItemYAMLModel ................................ 295
InventoryUI .................................................. 297
InventoryYAMLModel ...................................... 301
InventoryYAMLParser ..................................... 303
ItemAction
Struct that defines an action being the name of the action and the UnityAction. The action may
also have an id that specifies which subcategory of action it falls under. .................... 306
ItemActionButtonUI ........................................ 310
ItemAttributeUI ............................................ 313
ItemCategory
Abstract class for the classes which contain suites of actions and attributes which can be used
to define a category of items. Examples include solid, liquid, and plant. .................... 316
ItemCondition ............................................... 321
ItemDiscarder .............................................. 325
ItemDistrictModel .......................................... 326
ItemFactory
Factory that creates base items given its name and complex crafted items given recipe and
ingredients. Currently just a placeholder class only able to craft a fishing rod. ................. 327
ItemGenerator ............................................... 333
ItemPlacementSamplePoint ............................... 337
ItemPoolInfo .............................................. 341
ItemPoolManager ......................................... 344
ItemRarity ................................................... 349
ItemSerializer ............................................. 351
ItemSpawner ............................................... 354
ItemStack .................................................... 358
ItemStackDetailPanelBehavior .......................... 361
ItemStackUI .................................................. 367
ItemType ................................................... 374
ItemYAMLMap .............................................. 379
JellyFishSpawner .............................................. 381
KeyCodeConfigMenuBehavior .................................... 383
LandMovement .................................................. 385
LightCategory .................................................. 391
LightItem ........................................................ 396
Lightning .......................................................... 400
LightningEffect .................................................. 401
LoadingScreen .................................................. 402
LoadingScreenFlavorText ....................................... 403
LoadingScreenFlavorTextManager ................................ 404
LoadingScreenFlavorTextYAMLParser 405
MainMenuBehavior ................................................ 407
MedicineCategory ................................................ 410
Movement .......................................................... 415
MysteryAnnouncement Mystery announcement base class 419
Note Base interactable note class for all notes. 422
NoteData Base class for all notes. 425
NoteFactory ...................................................... 427
NoteUIController Note user interface controller. TODO: Make this a child of game UI 429
NoteYAMLParser 432
NullParent Sometimes you want something to be in a prefab, but not have a parent in the heirarchy. This script helps with that. 433
OverworldItemOptionSelection .................................. 434
PauseSystem ..................................................... 438
PickUpItem ........................................................ 441
PlantCategory Contains attributes and actions that befit a plant category item. 444
Player .............................................................. 450
PlayerAnimationEvents This script has functions called by the player's animator 455
PlayerController .................................................. 457
PlayerFocus Controls the player's head IK setup to have them look at the closest interactable if they can. 465
PlayerInventory ................................................... 467
PlayerStatManager ................................................ 472
PlayerTools Used for keeping track of implementation of tools in the game scene. 475
PressureSystem .................................................... 477
PressureSystems .................................................. 479
PressureSystemVisualization ................................... 481
ProceduralBuilding A building created procedurally. 482
ProceduralBuildingAttachment Used to define important parts of building attachments that are used during building creation and item placement. 487
ProceduralBuildingBase Defines parts of the building base that are needed for procedural building generation. 490
DistrictConfiguration.ProceduralBuildingBasesSizeXDefinition Holds all of the bases and roofs for each size of building. 493
ProceduralBuildingCreator Defines all the buildings for each district and allows for the creation of buildings in those districts. 495
ProceduralBuildingInstance
This class holds all the info that comes with making a procedural building ........................................ 498

ProceduralBuildingRoof
Attached to the roof prefabs. Holds roof specific information needed when creating buildings. . 502

ProceduralBuildingWindow
Attached to the window prefabs. Holds window specific information needed when creating build-

ings. .............................................................................................................................................. 503

Radio .................................................................................................................................................. 505

RadioButton ........................................................................................................................................ 511

RaftCategory ...................................................................................................................................... 512

RaftInteractable ................................................................................................................................ 515

RaftMovement ..................................................................................................................................... 518

RaftWake ............................................................................................................................................... 524

RainController
This class handles the rain effect that is present throughout the entire game ........................................ 525

RandomUtility ..................................................................................................................................... 527

Recipe
Class that handles storing the contents of recipes and checking whether or not the requirements
are met. ................................................................................................................................................ 530

RecipeBookBehavior ............................................................................................................................. 534

RecipeButtonGUIBehavior ..................................................................................................................... 537

RecipePageBehavior ............................................................................................................................... 542

RecipeRequirementsUI .......................................................................................................................... 547

RecipeYamlSerializer ............................................................................................................................ 549

Regression .............................................................................................................................................. 551

Requirement
A requirement includes a type of item (an item with a certain tag) and the amount of it. ..................... 552

RestartBehavior ................................................................................................................................... 554

RooftopGeneration ................................................................................................................................. 556

RooftopPointGenerator .......................................................................................................................... 559

SamplingPointGenerator ....................................................................................................................... 562

SelectedIngredientButton ...................................................................................................................... 566

SetCreatureGUI ..................................................................................................................................... 570

SetEnvironmentGUI ............................................................................................................................... 571

SettingsMenuBehavior .......................................................................................................................... 572

ShelterCategory ...................................................................................................................................... 574

ShelterInteractable ................................................................................................................................. 577

SicknessNotification ............................................................................................................................... 580

SolidCategory
The solid item category. Contains attributes and actions that befit a solid category item. .................. 581

SplashScreenBehavior ........................................................................................................................... 587

SpriteManager ........................................................................................................................................ 588

StatRate .................................................................................................................................................. 590

TeleportPlayer.Tel
TeleportPlayer
A class that moves the player when they press a key. Used by fire escapes, certain doors, and
the window washers ............................................................................................................................... 595

TemplateBuilding
A building based on a prefab. ................................................................................................................ 597

TestGUI ................................................................................................................................................... 600

Thermometer ......................................................................................................................................... 604

Tool ........................................................................................................................................................ 607

TransformParent
Can Parent one transform to another while ignoring specific axis. ....................................................... 611

Tuple<T1, T2>
Literally here so I can have a way of having int pairs without converting back and forth from
Vector2. ............................................................................................................................................... 613

TutorialGraffiti ..................................................................................................................................... 614
<table>
<thead>
<tr>
<th>Class Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>uGUITools</td>
</tr>
<tr>
<td>VectorUtility</td>
</tr>
<tr>
<td>Voroni</td>
</tr>
<tr>
<td>WarmthIdolCategory</td>
</tr>
<tr>
<td>WarmthRateManager</td>
</tr>
<tr>
<td>WaterItemGeneration</td>
</tr>
<tr>
<td>WaterMovement</td>
</tr>
<tr>
<td>WaterPointGenerator</td>
</tr>
<tr>
<td>WeatherSoundSystem</td>
</tr>
<tr>
<td>WeatherSystem</td>
</tr>
<tr>
<td>WeightedPair</td>
</tr>
<tr>
<td>WhaleSpawner</td>
</tr>
<tr>
<td>WindMovement</td>
</tr>
<tr>
<td>WindowWasher</td>
</tr>
</tbody>
</table>

WindowWasher

Goes on the window washer prefab, handles the ropes on the washer, moving it up with the water, and teleporting the player.
Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Assets/Scripts/AmbientSoundManager.cs .................................................. 669
Assets/Scripts/Game.cs ............................................................................ 678
Assets/Scripts/Camera/CameraController.cs ................................................. 669
Assets/Scripts/City/Block.cs ................................................................. 669
Assets/Scripts/City/BlockGenerator.cs ......................................................... 669
Assets/Scripts/City/Building.cs .............................................................. 669
Assets/Scripts/City/BuildingGenerator.cs ..................................................... 670
Assets/Scripts/City/City.cs ..................................................................... 670
Assets/Scripts/City/CityBoundaries.cs ......................................................... 672
Assets/Scripts/City/CityController.cs ......................................................... 672
Assets/Scripts/City/DebugCity.cs .............................................................. 672
Assets/Scripts/City/District.cs ................................................................. 673
Assets/Scripts/City/DistrictDefinition.cs ..................................................... 673
Assets/Scripts/City/DistrictGenerator.cs ..................................................... 673
Assets/Scripts/City/DistrictPopupAttribute.cs .............................................. 673
Assets/Scripts/City/GenerationUtility.cs ..................................................... 673
Assets/Scripts/City/ProceduralBuilding.cs .................................................. 674
Assets/Scripts/City/TemplateBuilding.cs ..................................................... 674
Assets/Scripts/City/Voronoi.cs .................................................................. 674
Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingAttachment.cs ............................. 670
Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingBase.cs ........................................ 670
Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingCreator.cs .................................. 670
Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingInstance.cs ................................. 671
Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingRoof.cs ......................................... 672
Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingWindow.cs ................................ 672
Assets/Scripts/City/Management/CityChunk.cs ........................................... 673
Assets/Scripts/Creatures/CreatureManager.cs .............................................. 674
Assets/Scripts/Creatures/CreatureSpawner.cs .............................................. 674
Assets/Scripts/Creatures/CreatureTracker.cs .............................................. 675
Assets/Scripts/Creatures/FishSpawner.cs ..................................................... 675
Assets/Scripts/Creatures/JellyFishSpawner.cs .............................................. 675
Assets/Scripts/Creatures/WhaleSpawner.cs ................................................. 675
Assets/Scripts/Creatures/UI/SetCreatureGUI.cs ........................................ 675
Assets/Scripts/DeathScreen/DeathFlavorText.cs .......................................... 675
Assets/Scripts/Item Crafting/Backend/ItemCategory.cs ........................................ 684
Assets/Scripts/Item Crafting/Backend/ItemCondition.cs ........................................ 684
Assets/Scripts/Item Crafting/Backend/ItemFactory.cs ........................................... 685
Assets/Scripts/Item Crafting/Backend/ItemRarity.cs ............................................. 685
Assets/Scripts/Item Crafting/Backend/ItemStack.cs ............................................... 685
Assets/Scripts/Item Crafting/Backend/ItemTypes.cs .............................................. 685
Assets/Scripts/Item Crafting/Backend/LightCategory.cs ......................................... 685
Assets/Scripts/Item Crafting/Backend/MedicineCategory.cs ...................................... 685
Assets/Scripts/Item Crafting/Backend/PlantCategory.cs .......................................... 686
Assets/Scripts/Item Crafting/Backend/PlayerInventory.cs ........................................ 686
Assets/Scripts/Item Crafting/Backend/RaftCategory.cs ........................................... 686
Assets/Scripts/Item Crafting/Backend/Recipe.cs ..................................................... 686
Assets/Scripts/Item Crafting/Backend/Requirement.cs ............................................. 686
Assets/Scripts/Item Crafting/Backend/ShelterCategory.cs ....................................... 686
Assets/Scripts/Item Crafting/Backend/SolidCategory.cs .......................................... 686
Assets/Scripts/Item Crafting/Backend/WarmthIdolCategory.cs .................................. 687
Assets/Scripts/ItemProceduralGeneration/DistrictItemConfiguration.cs ...................... 687
Assets/Scripts/ItemProceduralGeneration/DistrictItemRarityConfiguration.cs ............. 687
Assets/Scripts/ItemProceduralGeneration/DummyGenerator.cs .................................. 687
Assets/Scripts/ItemProceduralGeneration/ItemDiscarder.cs .................................... 688
Assets/Scripts/ItemProceduralGeneration/ItemGenerator.cs .................................... 688
Assets/Scripts/ItemProceduralGeneration/ItemPlacementSamplePoint.cs .................... 688
Assets/Scripts/ItemProceduralGeneration/ItemPoolInfo.cs ..................................... 688
Assets/Scripts/ItemProceduralGeneration/ItemPoolManager.cs ................................ 688
Assets/Scripts/ItemProceduralGeneration/ItemSpawner.cs ...................................... 688
Assets/Scripts/ItemProceduralGeneration/RooftopGenerator.cs ................................ 688
Assets/Scripts/ItemProceduralGeneration/RooftopPointGenerator.cs ......................... 689
Assets/Scripts/ItemProceduralGeneration/SamplingPointGenerator.cs ....................... 689
Assets/Scripts/ItemProceduralGeneration/WaterItemGeneration.cs ............................ 689
Assets/Scripts/ItemProceduralGeneration/WaterPointGenerator.cs ............................. 689
Assets/Scripts/ItemProceduralGeneration/WeightedPair.cs .................................... 689
Assets/Scripts/ItemProceduralGeneration/WorldItemFactory.cs ................................ 689
Assets/Scripts/Items/Barometer.cs ................................................................. 690
Assets/Scripts/Items/FireInteractable.cs ............................................................. 690
Assets/Scripts/Items/FishingLure.cs ..................................................................... 690
Assets/Scripts/Items/FishingRod.cs ..................................................................... 690
Assets/Scripts/Items/FloatBehavior.cs .................................................................. 690
Assets/Scripts/Items/HasSelectionInterface.cs ...................................................... 690
Assets/Scripts/Items/Idol.cs .............................................................................. 690
Assets/Scripts/Items/InteractableObject.cs .......................................................... 690
Assets/Scripts/Items/InteractableSetUp.cs ........................................................... 691
Assets/Scripts/Items/InventoryInteractable.cs ....................................................... 691
Assets/Scripts/Items/LightItem.cs ........................................................................ 691
Assets/Scripts/Items/OverworldItemOptionSelection.cs .......................................... 691
Assets/Scripts/Items/PickUpItem.cs ..................................................................... 691
Assets/Scripts/Items/RaftInteractable.cs ............................................................... 691
Assets/Scripts/Items/ShelterInteractable.cs ............................................................ 692
Assets/Scripts/Items/Thermometer.cs .................................................................. 692
Assets/Scripts/Items/Tool.cs .............................................................................. 692
Assets/Scripts/Loading/GameLoader.cs .................................................................. 692
Assets/Scripts/Loading/GameLoaderTask.cs ............................................................ 692
Assets/Scripts/Loading/LoadingScreen.cs .............................................................. 692
Assets/Scripts/Loading/LoadingScreenFlavorText.cs .............................................. 693
Assets/Scripts/Loading/LoadingScreenFlavorTextManager.cs .................................. 693
Assets/Scripts/Movement/LandMovement.cs .......................................................... 693
Assets/Scripts/Movement/Movement.cs ................................................................. 693
Assets/Scripts/Movement/PlayerAnimationEvents.cs .............................................. 694
Assets/Scripts/Movement/PlayerFocus.cs ............................................................... 694
### 3.1 File List

<table>
<thead>
<tr>
<th>Directory</th>
<th>File Name</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assets/Scripts/YamlParsing/</td>
<td>InventoryItemYAMLModel.cs</td>
<td>706</td>
</tr>
<tr>
<td></td>
<td>InventoryYAMLModel.cs</td>
<td>706</td>
</tr>
<tr>
<td></td>
<td>InventoryYamlParser.cs</td>
<td>706</td>
</tr>
<tr>
<td></td>
<td>ItemDistrictModel.cs</td>
<td>706</td>
</tr>
<tr>
<td></td>
<td>ItemSerializer.cs</td>
<td>706</td>
</tr>
<tr>
<td></td>
<td>ItemYAMLMap.cs</td>
<td>706</td>
</tr>
<tr>
<td></td>
<td>LoadingScreenFlavorTextYAMLParser.cs</td>
<td>706</td>
</tr>
<tr>
<td></td>
<td>NoteYAMLParser.cs</td>
<td>707</td>
</tr>
<tr>
<td></td>
<td>RecipeYamlSerializer.cs</td>
<td>707</td>
</tr>
</tbody>
</table>
Chapter 4

Class Documentation

4.1 AmbientSoundManager Class Reference

Inheritance diagram for AmbientSoundManager:
Collaboration diagram for AmbientSoundManager:

```
MonoBehaviour

AmbientSoundManager
```

### 4.1.1 Detailed Description

Definition at line 6 of file AmbientSoundManager.cs.

The documentation for this class was generated from the following file:

```
- Assets/Scripts/AmbientSoundManager.cs
```

### 4.2 AnnouncementFactory Class Reference

Collaboration diagram for AnnouncementFactory:

```
AnnouncementFactory
+ AnnouncementDatabase
+ LocationDatabase
+ WeatherDatabase
+ TimeDatabase
+ PlayerDatabase
+ AnnouncementFactory()
```
4.2 AnnouncementFactory Class Reference

Public Member Functions

- **AnnouncementFactory ()**
  
  Initializes a new instance of the AnnouncementFactory class.

Public Attributes

- List<MysteryAnnouncement> AnnouncementDatabase
  
  The full announcement database.
- List<MysteryAnnouncement> LocationDatabase
  
  The database of announcements in the location category.
- List<MysteryAnnouncement> WeatherDatabase
  
  The database of announcements in the weather category.
- List<MysteryAnnouncement> TimeDatabase
  
  The database of announcements in the time category.
- List<MysteryAnnouncement> PlayerDatabase
  
  The database of announcements in the player category.

4.2.1 Detailed Description

Definition at line 7 of file AnnouncementFactory.cs.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 AnnouncementFactory()

AnnouncementFactory.AnnouncementFactory ()

Initializes a new instance of the AnnouncementFactory class.

Definition at line 81 of file AnnouncementFactory.cs.

4.2.3 Member Data Documentation

4.2.3.1 AnnouncementDatabase

List<MysteryAnnouncement> AnnouncementFactory.AnnouncementDatabase

The full announcement database.

Definition at line 12 of file AnnouncementFactory.cs.

Generated by Doxygen
4.2.3.2 LocationDatabase

`List<MysteryAnnouncement> AnnouncementFactory.LocationDatabase`

The database of announcements in the location category.

Definition at line 17 of file AnnouncementFactory.cs.

4.2.3.3 PlayerDatabase

`List<MysteryAnnouncement> AnnouncementFactory.PlayerDatabase`

The database of announcements in the player category.

Definition at line 32 of file AnnouncementFactory.cs.

4.2.3.4 TimeDatabase

`List<MysteryAnnouncement> AnnouncementFactory.TimeDatabase`

The database of announcements in the time category.

Definition at line 27 of file AnnouncementFactory.cs.

4.2.3.5 WeatherDatabase

`List<MysteryAnnouncement> AnnouncementFactory.WeatherDatabase`

The database of announcements in the weather category.

Definition at line 22 of file AnnouncementFactory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Radio/Announcements/AnnouncementFactory.cs
4.3 AnnouncementYAMLParser Class Reference

Collaboration diagram for AnnouncementYAMLParser:

```
AnnouncementYAMLParser
+ AnnouncementYAMLParser()
+ LoadAnnouncements()
```

Public Member Functions

- **AnnouncementYAMLParser (string ttsFile)**
  
  Initializes a new instance of the AnnouncementYAMLParser class.

- **List<MysteryAnnouncement> LoadAnnouncements()**
  
  Loads the announcements.

4.3.1 Detailed Description

Definition at line 11 of file AnnouncementYAMLParser.cs.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 AnnouncementYAMLParser()

```
AnnouncementYAMLParser.AnnouncementYAMLParser (string ttsFile)
```

Initializes a new instance of the AnnouncementYAMLParser class.

**Parameters**

- **ttsFile** | Tts file.

Definition at line 23 of file AnnouncementYAMLParser.cs.
4.3.3 Member Function Documentation

4.3.3.1 LoadAnnouncements()

\[ \text{List}\langle \text{MysteryAnnouncement} \rangle \ \text{AnnouncementYAMLParser.LoadAnnouncements()} \]

Loads the announcements.

Returns

The announcements.

Definition at line 32 of file AnnouncementYAMLParser.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/AnnouncementYAMLParser.cs

4.4 Attribute Class Reference

Collaboration diagram for Attribute:

```
Attribute
+ Name
+ Value
+ Attribute()
+ GetDuplicate()
```

Public Member Functions

- \text{Attribute} (string name, float value)
- \text{Attribute GetDuplicate} ()

\text{Gets the duplicate of the attribute}

Properties

- string \text{Name} [get, set]
- float \text{Value} [get, set]
4.4 Attribute Class Reference

4.4.1 Detailed Description

Definition at line 4 of file Attribute.cs.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 Attribute()

    Attribute ( string name, float value )

Definition at line 6 of file Attribute.cs.

4.4.3 Member Function Documentation

4.4.3.1 GetDuplicate()

    Attribute Attribute.GetDuplicate ( )

Gets the duplicate of the attribute

Returns
   The duplicate.

Definition at line 28 of file Attribute.cs.

4.4.4 Property Documentation

4.4.4.1 Name

    string Attribute.Name [get], [set]

Definition at line 13 of file Attribute.cs.
### 4.4.4.2 Value

```csharp
float Attribute.Value [get], [set]
```

Definition at line 19 of file Attribute.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/Attribute.cs

### 4.5 Barometer Class Reference

Inheritance diagram for Barometer:
Collaboration diagram for Barometer:

![Collaboration Diagram](image)

Public Member Functions

- override void **SetUpTool** *(BaseItem itemForTool)*
  
  Sets up the tool so that it is linked to the proper item in the inventory.

- override void **Use** ()
  
  Uses the barometer.

- override void **Equip** ()
  
  Equip the barometer.

- override void **Unequip** ()
  
  Unequip the barometer.

Properties

- bool **CheckBarometer**  
  
  Returns true if the barometer is in use.
Additional Inherited Members

4.5.1 Detailed Description

Definition at line 7 of file Barometer.cs.

4.5.2 Member Function Documentation

4.5.2.1 Equip()

```csharp
override void Barometer.Equip ( ) [virtual]
```

Equip the barometer.

Implements Tool.

Definition at line 68 of file Barometer.cs.

4.5.2.2 SetUpTool()

```csharp
override void Barometer.SetUpTool ( BaseItem itemForTool ) [virtual]
```

Sets up the tool so that it is linked to the proper item in the inventory.

Parameters

| itemForTool | Item for tool. |

Reimplemented from Tool.

Definition at line 34 of file Barometer.cs.

4.5.2.3 Unequip()

```csharp
override void Barometer.Unequip ( ) [virtual]
```

Unequip the barometer.

Implements Tool.

Definition at line 77 of file Barometer.cs.
4.5.2.4 Use()

```csharp
override void Barometer.Use () [virtual]
```

Uses the barometer.

Implements Tool.

Definition at line 59 of file Barometer.cs.

4.5.3 Property Documentation

4.5.3.1 CheckBarometer

```csharp
bool Barometer.CheckBarometer [get]
```

Returns true if the barometer is in use.

Definition at line 87 of file Barometer.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/Barometer.cs

4.6 BaseItem Class Reference

Class attached to all items. Handles getting information from the various Item Categories.
Inheritance diagram for BaseItem:
Collaboration diagram for BaseItem:

```
CollectableItem
   + GetPossibleActions()

BaseItem
   + DirtyFlag
   + UpdateExistingFlag
   + RemovalFlag
   + DiscardFlag
   + ItemName
   + Types
   + FlavorText
   + InventorySprite
   + WorldModel
   + ActionModifiedSprites
   + ModifyingActionNames
   + ActionModifiedModels
   + Rarity

   + UpdateItemEvent()
   + BaseItem()
   + BaseItem()
   + BaseItem()
   + InitializeBaseItem()
   + SetUpBaseItem()
   + AddItemCategory()
   + GetItemAttributes()
   + GetPossibleActions()
   + GetNumberOfActionsCompleted()
   and 13 more...
```

Public Member Functions

- delegate void `UpdateItemEvent(BaseItem item)`
  
  Delegate function that takes in a baseItem

- `BaseItem()`

  Initializer only used during Yaml Deserialization

- `BaseItem(string name)`

  Initializes a new instance of the BaseItem class.

- `BaseItem(BaseItem original)`

  Creates a deep copy of the BaseItem and all category items.

- `void InitializeBaseItem()`

  Initializes the lists that the base item needs.

- `void SetUpBaseItem()`

  Sets up the item by linking the item attributes together.
• void AddItemCategory (ItemCategory category)
  Adds an ItemCategory to the list of item categories.

• List<ItemCategory> GetItemAttributes ()
  Gets the attributes of an item. Returns all attributes in the BaseItem and all Item Categories.

• override List<ItemAction> GetPossibleActions ()
  Gets all the action that an item can perform.

• int GetNumberOfActionsCompleted ()
  Gets the number of actions completed by the item.

• void ChangeName (string name)
  Changes the name of the object.

• void FireItemNameChangedEvent ()
  Fires a text change event that fires any functions subscribed that the BaseItem has changed.

• void FireItemSpriteChangedEvent ()

• void RemoveCategoriesSpecified (List<string> names)
  Removes the categories specified in the list of names.

• void RemoveCategoriesExcluding (List<string> names)
  Removes the categories excluding the categories in the list of names.

• List<ItemCategory> GetItemCategories ()
  Gets the list of categories attached to the item.

• ItemCategory GetItemCategoryByParentClass (Type categoryType)
  Gets the first item found that is a subclass of the given class.

• ItemCategory GetItemCategoryByClass (Type categoryType)
  Gets the item category by class.

• BaseItem GetItemToModify ()
  Gets a separate copy of this item that will be modified during crafting. This function assumes that the duplicate items will be drawn from this item, thus the "amount" that goes into the duplicated item will be subtracted from this item. For example, if there are four river weeds, calling GetItemsToModify will result in the initial base item class for the river weeds having an amount of 2, while the duplicated base item class will have an amount of 2 as well.

• void RemoveAttributes (List<Attribute> toBeRemoved)
  Removes values from the attributes list.

• Attribute GetItemAttribute (string name)
  Gets the attribute from the list of all attributes in this item.

• void Discard ()
  Marks that this BaseItem should be discarded.

• void SetNewModel (int newModelIndex)
  Changes the model and sprite used to represent the item as specified by ActionModifiedModels and ActionModifiedSprites.

Public Attributes

• bool DirtyFlag = false
  If the item attribute has been changed, and a new item has resulted, DirtyFlag is true.

• bool UpdateExistingFlag = false
  If the item has been changed, but no new item has resulted, UpdateExistingFlag is true.

• bool RemovalFlag = false
  Flag checked by the InventoryItemBehavior to see if the BaseItem should actually be removed from the inventory due to being consumed or otherwise used when checking for modifications.

• bool DiscardFlag = false
  Flag checked by InventoryItemBehavior to see if the BaseItem should be discarded into the overworld.
Properties

- **string ItemName** [get, set]
  Gets or sets the name of the item.

- **List<string> Types** [get, set]
  Tags that specify what type of item this item is unfortunately enums are not serializable with yamldotnet as such, they must remain strings for now.

- **string FlavorsText** [get, set]
  Gets or sets the flavor text that describes the item.

- **string InventorySprite** [get, set]
  Gets or sets the sprite representation of the item in the inventory.

- **string WorldModel** [get, set]
  Gets or sets the model that represents the item in the world.

- **List<string> ActionModifiedSprites** [get, set]
  Gets or sets sprite that should be displayed when the item is woven into a basket.

- **List<string> ModifyingActionNames** [get, set]
  Gets or sets model that should be displayed when the item is woven into a rope.

- **List<string> ActionModifiedModels** [get, set]
  Gets or sets sprite that should be displayed when the item is woven into a rope.

- **string Rarity** [get, set]
  Gets or sets the rarity.

Events

- **UpdateItemEvent UpdateItemName**
  Event that can be subscribed to by functions of UpdateItemEvent format

- **UpdateItemEvent UpdateItemSprite**
  Occurs when item changes the sprite it should display.

### 4.6.1 Detailed Description

Class attached to all items. Handles getting information from the various Item Categories.

Definition at line 10 of file BaseItem.cs.

### 4.6.2 Constructor & Destructor Documentation

#### 4.6.2.1 BaseItem() [1/3]

BaseItem.BaseItem()

Initializer only used during Yaml Deserialization

Definition at line 153 of file BaseItem.cs.

#### 4.6.2.2 BaseItem() [2/3]

BaseItem.BaseItem(
  string name)

Initializes a new instance of the BaseItem class.
Parameters

| name | Name |

Definition at line 162 of file BaseItem.cs.

4.6.2.3 BaseItem()

BaseItem.BaseItem (BaseItem original)

Creates a deep copy of the BaseItem and all category items.

Parameters

| original | Original BaseItem to be copied |

Definition at line 173 of file BaseItem.cs.

4.6.3 Member Function Documentation

4.6.3.1 AddItemCategory()

void BaseItem.AddItemCategory (ItemCategory category)

Adds an ItemCategory to the list of item categories.

Parameters

| category | Category |

Definition at line 242 of file BaseItem.cs.

4.6.3.2 ChangeName()

void BaseItem.ChangeName (string name)

Changes the name of the object.
### Parameters

| name | Name. |

Definition at line 310 of file BaseItem.cs.

#### 4.6.3.3 Discard()

```csharp
void BaseItem.Discard();
```

Marks that this `BaseItem` should be discarded.

Definition at line 463 of file BaseItem.cs.

#### 4.6.3.4 FireItemNameChangedEvent()

```csharp
void BaseItem.FireItemNameChangedEvent();
```

Fires a text change event that fires any functions subscribed that the `BaseItem` has changed.

Definition at line 319 of file BaseItem.cs.

#### 4.6.3.5 FireItemSpriteChangedEvent()

```csharp
void BaseItem.FireItemSpriteChangedEvent();
```

Definition at line 327 of file BaseItem.cs.

#### 4.6.3.6 GetItemAttribute()

```csharp
Attribute BaseItem.GetItemAttribute(string name);
```

Gets the attribute from the list of all attributes in this item.

Returns

- The item attribute.
Parameters

\begin{tabular}{|c|c|}
\hline
name & Name. \\
\hline
\end{tabular}

Definition at line 447 of file BaseItem.cs.

4.6.3.7 GetItemAttributes()

\begin{verbatim}
List<Attribute> BaseItem.GetItemAttributes ()
\end{verbatim}

Gets the attributes of an item. Returns all attributes in the BaseItem and all Item Categories.

Returns

The attributes of an item in a Dictionary keyed by the attribute name. All attribute values are floats.

Definition at line 257 of file BaseItem.cs.

4.6.3.8 GetItemCategories()

\begin{verbatim}
List<ItemCategory> BaseItem.GetItemCategories ()
\end{verbatim}

Gets the list of categories attached to the item.

Returns

The item categories.

Definition at line 375 of file BaseItem.cs.

4.6.3.9 GetItemCategoryByClass()

\begin{verbatim}
ItemCategory BaseItem.GetItemCategoryByClass ( Type categoryType )
\end{verbatim}

Gets the item category by class.

Returns

The item category by class.
Parameters

| categoryType | Category type. |

Definition at line 403 of file BaseItem.cs.

### 4.6.3.10 GetItemCategoryByParentClass()

```csharp
ItemCategory BaseItem.GetItemCategoryByParentClass (Type categoryType )
```

Gets the first item found that is a subclass of the given class

Returns

The item category.

Parameters

| categoryType | Category type. |

Definition at line 385 of file BaseItem.cs.

### 4.6.3.11 GetItemToModify()

```csharp
BaseItem BaseItem.GetItemToModify ( )
```

Gets a separate copy of this item that will be modified during crafting. This function assumes that the duplicate items will be drawn from this item, thus the "amount" that goes into the duplicated item will be subtracted from this item. For example, if there are four river weeds, calling GetItemsToModify will result in the initial base item class for the river weeds having an amount of 2, while the duplicated base item class will have an amount of 2 as well.

Parameters

| amt | The amount of the item that should be affected during the crafting. |

Returns

The duplicate.

Definition at line 425 of file BaseItem.cs.
4.6.3.12 GetNumberOfActionsCompleted()

```csharp
int BaseItem.GetNumberOfActionsCompleted() { }
```

Gets the number of actions completed by the item.

**Returns**
- The all actions.

Definition at line 288 of file BaseItem.cs.

4.6.3.13 GetPossibleActions()

```csharp
override List<ItemAction> BaseItem.GetPossibleActions() { virtual }
```

Gets all the action that an item can perform.

**Returns**
- The possible actions of an item in a Dictionary keyed by the action name.

Reimplemented from CollectableItem.

Definition at line 270 of file BaseItem.cs.

4.6.3.14 InitializeBaseItem()

```csharp
void BaseItem.InitializeBaseItem() { }
```

Initializes the lists that the base item needs.

Definition at line 213 of file BaseItem.cs.

4.6.3.15 RemoveAttributes()

```csharp
void BaseItem.RemoveAttributes(
    List<Attribute> toBeRemoved)
```

Removes values from the attributes list.

**Parameters**
- toBeRemoved: Attributes to be removed.
4.6.3.16 RemoveCategoriesExcluding()

void BaseItem.RemoveCategoriesExcluding ( 
     List<string> names )

Removes the categories excluding the categories in the list of names.

Parameters

   names   | Names.

Definition at line 366 of file BaseItem.cs.

4.6.3.17 RemoveCategoriesSpecified()

void BaseItem.RemoveCategoriesSpecified ( 
     List<string> names )

Removes the categories specified in the list of names.

Parameters

   names   | Names.

Definition at line 357 of file BaseItem.cs.

4.6.3.18 SetNewModel()

void BaseItem.SetNewModel ( 
     int newModelIndex )

Changes the model and sprite used to represent the item as specified by ActionModifiedModels and ActionModifiedSprites.

Parameters

   newModelIndex   | Index number of the new model.

Definition at line 472 of file BaseItem.cs.
4.6.3.19  SetUpBaseItem()

void BaseItem.SetUpBaseItem ( )

Sets up the item by linking the item attributes together.
Definition at line 226 of file BaseItem.cs.

4.6.3.20  UpdateItemEvent()

delegate void BaseItem.UpdateItemEvent ( BaseItem item )

Delegate function that takes in a baseItem

4.6.4  Member Data Documentation

4.6.4.1  DirtyFlag

bool BaseItem.DirtyFlag = false

If the item attribute has been changed, and a new item has resulted, DirtyFlag is true.
Definition at line 132 of file BaseItem.cs.

4.6.4.2  DiscardFlag

bool BaseItem.DiscardFlag = false

Flag checked by InventoryItemBehavior to see if the BaseItem should be discarded into the overworld
Definition at line 148 of file BaseItem.cs.

4.6.4.3  RemovalFlag

bool BaseItem.RemovalFlag = false

Flag checked by the InventoryItemBehavior to see if the BaseItem should actually be removed from the inventory
due to being consumed or otherwise used when checking for modifications
Definition at line 143 of file BaseItem.cs.
4.6.4.4 UpdateExistingFlag

```csharp
bool BaseItem.UpdateExistingFlag = false
```

If the item has been changed, but no new item has resulted, UpdateExistingFlag is true.

Definition at line 137 of file BaseItem.cs.

### 4.6.5 Property Documentation

#### 4.6.5.1 ActionModifiedModels

```csharp
List<string> BaseItem.ActionModifiedModels [get], [set]
```

Gets or sets sprite that should be displayed when the item is woven into a rope.

The weave basket model.

Definition at line 89 of file BaseItem.cs.

#### 4.6.5.2 ActionModifiedSprites

```csharp
List<string> BaseItem.ActionModifiedSprites [get], [set]
```

Gets or sets sprite that should be displayed when the item is woven into a basket.

The weave basket model.

Definition at line 69 of file BaseItem.cs.

#### 4.6.5.3 FlavorText

```csharp
string BaseItem.FlavorText [get], [set]
```

Gets or sets the flavor text that describes the item.

The flavor text.

Definition at line 39 of file BaseItem.cs.
4.6.5.4  InventorySprite

string BaseItem.InventorySprite [get], [set]

Gets or sets the sprite representation of the item in the inventory.

The inventory sprite.

Definition at line 49 of file BaseItem.cs.

4.6.5.5  ItemName

string BaseItem.ItemName [get], [set]

Gets or sets the name of the item.

The name of the item.

Definition at line 17 of file BaseItem.cs.

4.6.5.6  ModifyingActionNames

List<string> BaseItem.ModifyingActionNames [get], [set]

Gets or sets model that should be displayed when the item is woven into a rope.

The weave basket model.

Definition at line 79 of file BaseItem.cs.

4.6.5.7  Rarity

string BaseItem.Rarity [get], [set]

Gets or sets the rarity.

The rarity.

Definition at line 99 of file BaseItem.cs.
4.6.5.8 Types

`List<string> BaseItem.Types [get], [set]`

tags that specify what type of item this item is unfortunately enums are not serializable with yamldotnet as such, they must remain strings for now

The types.

Definition at line 29 of file BaseItem.cs.

4.6.5.9 WorldModel

`string BaseItem.WorldModel [get], [set]`

Gets or sets the model that represents the item in the world.

The world model.

Definition at line 59 of file BaseItem.cs.

4.6.6 Event Documentation

4.6.6.1 UpdateItemName

`UpdateItemEvent BaseItem.UpdateItemName`

Event that can be subscribed to by functions of UpdateItemEvent format

Definition at line 112 of file BaseItem.cs.

4.6.6.2 UpdateItemSprite

`UpdateItemEvent BaseItem.UpdateItemSprite`

Occurs when item changes the sprite it should display.

Definition at line 117 of file BaseItem.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/BaseItem.cs
### 4.7 BaseItemPopup Class Reference

Inheritance diagram for BaseItemPopup:

![Inheritance Diagram](#)

Collaboration diagram for BaseItemPopup:

![Collaboration Diagram](#)

#### Public Member Functions

- `override void OnGUI(Rect position, SerializedProperty property, GUIContent label)`

  Overrides the OnGUI method and renders a dropdown to pick an available baseItem.

#### 4.7.1 Detailed Description

Definition at line 8 of file BaseItemPopup.cs.
4.7.2 Member Function Documentation

4.7.2.1 OnGUI()

```csharp
override void BaseItemPopup.OnGUI (Rect position, SerializedProperty property, GUIContent label)
```

Overrides the OnGUI method and renders a dropdown to pick an available baseItem.

**Parameters**

<table>
<thead>
<tr>
<th>position</th>
<th>Position of dropdown.</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>Property to set.</td>
</tr>
<tr>
<td>label</td>
<td>Property label.</td>
</tr>
</tbody>
</table>

Definition at line 19 of file BaseItemPopup.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Editor/BaseItemPopup.cs

4.8 BaseItemPopupAttribute Class Reference

Inheritance diagram for BaseItemPopupAttribute:
Collaboration diagram for BaseItemPopupAttribute:

4.8.1 Detailed Description

Definition at line 4 of file BaseItemPopupAttribute.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/BaseItemPopupAttribute.cs

4.9 BeaconInteractable Class Reference

Inheritance diagram for BeaconInteractable:
Collaboration diagram for BeaconInteractable:

```
MonoBehaviour

BeaconInteractable
+ TurnLightOn()
```

Public Member Functions

- void `TurnLightOn()`

  Called to turn the light on and point it up in the air.

4.9.1 Detailed Description

Definition at line 6 of file BeaconInteractable.cs.

4.9.2 Member Function Documentation

4.9.2.1 TurnLightOn()

```c
void BeaconInteractable.TurnLightOn ( )
```

Called to turn the light on and point it up in the air.

Definition at line 56 of file BeaconInteractable.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Ending/BeaconInteractable.cs

Generated by Doxygen
4.10 BezierLine Class Reference

Inheritance diagram for BezierLine:

Collaboration diagram for BezierLine:

Properties

- LineRenderer RendererComponent [get]
**4.10 BezierLine Class Reference**

*Gets the line renderer component.*

- **Vector3** `StartPoint`  [get, set]
  *Starting point of the bezier curve.*

- **Vector3** `EndPoint`  [get, set]
  *End point of the bezier curve.*

- **Vector3** `ControlPoint`  [get, set]
  *Control point which the curve is interpolated towards.*

**4.10.1 Detailed Description**

Definition at line 5 of file BezierLine.cs.

**4.10.2 Property Documentation**

**4.10.2.1 ControlPoint**

`Vector3 BezierLine.ControlPoint [get], [set]
Control point which the curve is interpolated towards.
Definition at line 75 of file BezierLine.cs.

**4.10.2.2 EndPoint**

`Vector3 BezierLine.EndPoint [get], [set]
End point of the bezier curve.
Definition at line 59 of file BezierLine.cs.

**4.10.2.3 RendererComponent**

`LineRenderer BezierLine.RendererComponent [get]
Gets the line renderer component.
Definition at line 20 of file BezierLine.cs.`
4.10.2.4 StartPoint

Vector3 BezierLine.StartPoint [get], [set]

Starting point of the bezier curve.

Definition at line 43 of file BezierLine.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/BezierLine.cs

4.11 Block Class Reference

Collaboration diagram for Block:

```
+ Center
+ Verticies
+ Buildings
+ BoundingBox
+ Block()
+ ContainsPoint()
```

Public Member Functions

- **Block** (Vector3 controlPoint, Vector3[] vertices)
  
  Constructor for a city block

- **bool** ContainsPoint (Vector2 point)
  
  Checks whether the point is within the bounds of the block.

Properties

- **Vector3** Center [get]
  
  Center of the district.

- **Vector3**[] Verticies [get]
  
  The vertices that define the edges of the blocks.

- **List< Building >** Buildings [get]
  
  The list of buildings in this block.

- **Bounds** BoundingBox [get]
  
  The bounds that contain all of the block vertices.
4.11 Block Class Reference

4.11.1 Detailed Description

Definition at line 5 of file Block.cs.

4.11.2 Constructor & Destructor Documentation

4.11.2.1 Block()

Block.Block (  
    Vector3 controlPoint,  
    Vector3 [][] vertices )

Constructor for a city block

Parameters

<table>
<thead>
<tr>
<th>controlPoint</th>
<th>Control point used to construct the district.</th>
</tr>
</thead>
<tbody>
<tr>
<td>vertices</td>
<td>A list of four vertices to define the edges of the blocks.</td>
</tr>
</tbody>
</table>

Definition at line 14 of file Block.cs.

4.11.3 Member Function Documentation

4.11.3.1 ContainsPoint()

bool Block.ContainsPoint (  
    Vector2 point )

Checks whether the point is within the bounds of the block.

Returns

true, if point is within block, false otherwise.

Parameters

| point | Point to be checked. |

Definition at line 69 of file Block.cs.
4.11.4  Property Documentation

4.11.4.1  BoundingBox

Bounds Block.BoundingBox [get]

The bounds that contain all of the block vertices.

Definition at line 52 of file Block.cs.

4.11.4.2  Buildings

List<Building> Block.Buildings [get]

The list of buildings in this block.

Definition at line 43 of file Block.cs.

4.11.4.3  Center

Vector3 Block.Center [get]

Center of the district.

Definition at line 25 of file Block.cs.

4.11.4.4  Verticies

Vector3 [] Block.Verticies [get]

The vertices that define the edges of the blocks.

Definition at line 34 of file Block.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/Block.cs
Public Member Functions

- `Block[] Generate(int seed, District district)`

  Generate the blocks within the specified district based on the values specified in the block generator.

4.12.1 Detailed Description

Definition at line 7 of file BlockGenerator.cs.
4.12.2 Member Function Documentation

4.12.2.1 Generate()

```csharp
Block[] BlockGenerator.Generate(
    int seed,
    District district)
```

Generate the blocks within the specified district based on the values specified in the block generator.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>seed</code></td>
<td>The generation seed.</td>
</tr>
<tr>
<td><code>district</code></td>
<td>The district to generate blocks within.</td>
</tr>
</tbody>
</table>

Returns

An array of the generated blocks.

Definition at line 26 of file BlockGenerator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/BlockGenerator.cs

4.13 BooleanOperator Class Reference

Collaboration diagram for BooleanOperator:

```
+ BooleanOperatorDelegate()
+ Less()
+ LessOrEqual()
+ Greater()
+ GreaterOrEqual()
+ NotEqual()
+ Equal()
```
Public Member Functions

- delegate bool BooleanOperatorDelegate (float a, float b)
  Boolean operator delegate.

Static Public Member Functions

- static bool Less (float a, float b)
  Returns true if a is less than b.
- static bool LessOrEqual (float a, float b)
  Returns true if a is less than or equal to b.
- static bool Greater (float a, float b)
  Returns true if a is greater than b.
- static bool GreaterOrEqual (float a, float b)
  Returns true if a is greater than or equal to b.
- static bool NotEqual (float a, float b)
  Returns true if a is not equal to b.
- static bool Equal (float a, float b)
  Returns true if a is equal to b.

4.13.1 Detailed Description

Definition at line 1 of file BooleanOperator.cs.

4.13.2 Member Function Documentation

4.13.2.1 BooleanOperatorDelegate()

delagate bool BooleanOperator.BooleanOperatorDelegate {
    float a,
    float b )

Boolean operator delegate.

4.13.2.2 Equal()

static bool BooleanOperator.Equal {
    float a,
    float b ) [static]

Returns true if a is equal to b.
Parameters

<table>
<thead>
<tr>
<th>a</th>
<th>The alpha component.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>The blue component.</td>
</tr>
</tbody>
</table>

Definition at line 64 of file BooleanOperator.cs.

4.13.2.3 Greater()

```csharp
static bool BooleanOperator.Greater (  
    float a,  
    float b ) [static]
```

Returns true if a is greater than b.

Parameters

<table>
<thead>
<tr>
<th>a</th>
<th>The alpha component.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>The blue component.</td>
</tr>
</tbody>
</table>

Definition at line 34 of file BooleanOperator.cs.

4.13.2.4 GreaterOrEqual()

```csharp
static bool BooleanOperator.GreaterOrEqual (  
    float a,  
    float b ) [static]
```

Returns true if a is greater than or equal to b.

Parameters

<table>
<thead>
<tr>
<th>a</th>
<th>The alpha component.</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>The blue component.</td>
</tr>
</tbody>
</table>

Definition at line 44 of file BooleanOperator.cs.

4.13.2.5 Less()

```csharp
static bool BooleanOperator.Less (  
    float a,  
    float b ) [static]
```

Returns true if a is less than b.
Parameters

<table>
<thead>
<tr>
<th></th>
<th>The alpha component.</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td></td>
</tr>
</tbody>
</table>

Definition at line 13 of file BooleanOperator.cs.

### 4.13.2.6 LessOrEqual()

```csharp
static bool BooleanOperator.LessOrEqual ( float a, float b ) [static]
```

Returns true if a is less than or equal to b.

Returns
true, if or equal was lessed, false otherwise.

Parameters

<table>
<thead>
<tr>
<th></th>
<th>The alpha component.</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td></td>
</tr>
</tbody>
</table>

Definition at line 24 of file BooleanOperator.cs.

### 4.13.2.7 NotEqual()

```csharp
static bool BooleanOperator.NotEqual ( float a, float b ) [static]
```

Returns true if a is not equal to b.

Parameters

<table>
<thead>
<tr>
<th></th>
<th>The alpha component.</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td></td>
</tr>
</tbody>
</table>

Definition at line 54 of file BooleanOperator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/BooleanOperator.cs
4.14 Building Class Reference

Inheritance diagram for Building:

Collaboration diagram for Building:
Public Member Functions

- abstract void **Load** ()
  
  *Loads the instance of the building into the scene.*

- void **Unload** ()
  
  *Unloads the instance of the building from the scene.*

Properties

- bool **IsLoaded** [get, protected set]
  
  *Returns true if the building is loaded in the world.*

- Vector3 **Position** [get, protected set]
  
  *Position of the root of the building.*

- GameObject **Instance** [get, protected set]
  
  *The gameObject representing the building in the scene.*

- Transform **Parent** [get, protected set]
  
  *The parent we are parenting the buildings to.*

- ItemPlacementSamplePoint [ ] **AttachmentInformation** [get, set]
  
  *Gets or sets the door and shelter attachment information.*

- GameObject [ ] **Attachments** [get, set]
  
  *Gets or sets the attachments. These include doors and shelters.*

- Bounds **BoundingBox** [get]
  
  *The bounds defining the size of the building*

4.14.1 Detailed Description

Definition at line 4 of file Building.cs.

4.14.2 Member Function Documentation

4.14.2.1 **Load**()

abstract void Building.Load () [pure virtual]

*Loads the instance of the building into the scene.*

Implemented in **ProceduralBuilding**, and **TemplateBuilding**.

4.14.2.2 **Unload**()

void Building.Unload ()

*Unloads the instance of the building from the scene.*

Definition at line 125 of file Building.cs.
4.14.3 Property Documentation

4.14.3.1 AttachmentInformation

ItemPlacementSamplePoint [] Building.AttachmentInformation [get], [set]

Gets or sets the door and shelter attachment information.

The attachment information.

Definition at line 49 of file Building.cs.

4.14.3.2 Attachments

GameObject [] Building.Attachments [get], [set]

Gets or sets the attachments. These include doors and shelters.

The attachments.

Definition at line 59 of file Building.cs.

4.14.3.3 BoundingBox

Bounds Building.BoundingBox [get]

The bounds defining the size of the building

Definition at line 68 of file Building.cs.

4.14.3.4 Instance

GameObject Building.Instance [get], [protected set]

The gameObject representing the building in the scene.

Definition at line 30 of file Building.cs.
4.14.3.5  IsLoaded

bool Building.IsLoaded [get], [protected set]
Returns true if the building is loaded in the world.
Definition at line 12 of file Building.cs.

4.14.3.6  Parent

Transform Building.Parent [get], [protected set]
The parent we are parenting the buildings to.
Definition at line 39 of file Building.cs.

4.14.3.7  Position

Vector3 Building.Position [get], [protected set]
Position of the root of the building.
Definition at line 21 of file Building.cs.
The documentation for this class was generated from the following file:

  • Assets/Scripts/City/Building.cs

4.15  City.BuildingEnumerator Class Reference

Class for creating an iterator to traverse the tree-structure of buildings in the City.
Inheritance diagram for City.BuildingEnumerator:

```
IEnumerator< Building >

City.BuildingEnumerator
+ Current
+ Current
+ MoveNext()  
+ Dispose()  
+ Reset()
~ BuildingEnumerator()
```
Collaboration diagram for City.BuildingEnumerator:

Public Member Functions

- bool **MoveNext** ()
  
  Moves to the next building in iteration.

- void **Dispose** ()
  
  Disposes of the iterator.

- void **Reset** ()
  
  Traverses the tree and adds all buildings to the iterator.

Properties

- **Building Current** [get]
  
  Gets current building in iteration.

4.15.1 Detailed Description

Class for creating an iterator to traverse the tree-structure of buildings in the **City**.

Definition at line 63 of file City.cs.

4.15.2 Member Function Documentation
4.15.2.1 Dispose()

```csharp
void City.BuildingEnumerator.Dispose();
```

Disposes of the iterator.

Definition at line 113 of file City.cs.

4.15.2.2 MoveNext()

```csharp
bool City.BuildingEnumerator.MoveNext();
```

Moves to the next building in iteration.

Returns

True if index is in range.

Definition at line 105 of file City.cs.

4.15.2.3 Reset()

```csharp
void City.BuildingEnumerator.Reset();
```

Traverses the tree and adds all buildings to the iterator.

Definition at line 118 of file City.cs.

4.15.3 Property Documentation

4.15.3.1 Current

```csharp
```

Gets current building in iteration.

Definition at line 83 of file City.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/City.cs
4.16 BuildingGenerator Class Reference

Inheritance diagram for BuildingGenerator:

```
MonoBehaviour

BuildingGenerator
    + CityCenterBuilding
    + Generate()
```

Collaboration diagram for BuildingGenerator:

```
MonoBehaviour

BuildingGenerator
    + CityCenterBuilding
    + Generate()
```

Public Member Functions

- **Building [ ] Generate** (int seed, Block block, DistrictConfiguration configuration, Bounds cityBounds, Vector3 cityCenter, bool generateWeenie)

  Generates a building.
Properties

- GameObject CityCenterBuilding [get]
  
  Template for the building to place at the center of the city.

4.16.1 Detailed Description

Definition at line 5 of file BuildingGenerator.cs.

4.16.2 Member Function Documentation

4.16.2.1 Generate()

Building [] BuildingGenerator.Generate {
    int seed,
    Block block,
    DistrictConfiguration configuration,
    Bounds cityBounds,
    Vector3 cityCenter,
    bool generateWeenie
}

Generates a building.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seed</td>
<td>The procedural generation seed</td>
</tr>
<tr>
<td>block</td>
<td>The block in which the building is to be generated.</td>
</tr>
<tr>
<td>configuration</td>
<td>The building configuration for this district.</td>
</tr>
<tr>
<td>cityBounds</td>
<td>The bounds of the city.</td>
</tr>
<tr>
<td>cityCenter</td>
<td>The center of the city.</td>
</tr>
<tr>
<td>generateWeenie</td>
<td>If true, this block will generate the district Weenie building.</td>
</tr>
</tbody>
</table>

Returns

Definition at line 64 of file BuildingGenerator.cs.

4.16.3 Property Documentation
4.16.3.1 CityCenterBuilding

GameObject BuildingGenerator.CityCenterBuilding [get]

Template for the building to place at the center of the city.

Definition at line 47 of file BuildingGenerator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/BuildingGenerator.cs

4.17 buildingIndex Struct Reference

Collaboration diagram for buildingIndex:

```
buildingIndex
+ BuildingSize
+ PartNumber
```

Public Attributes

- int BuildingSize
- int PartNumber

4.17.1 Detailed Description

Definition at line 25 of file ProceduralBuildingCreator.cs.

4.17.2 Member Data Documentation

4.17.2.1 BuildingSize

int buildingIndex.BuildingSize

Definition at line 27 of file ProceduralBuildingCreator.cs.
4.18 DistrictConfiguration.BuildingTemplatePlacement Class Reference

Structure for accessing additional building placement configuration.

Collaboration diagram for DistrictConfiguration.BuildingTemplatePlacement:

```
DistrictConfiguration.BuildingTemplatePlacement
  + Building
  + PlacementFrequency
```

**Public Attributes**

- GameObject **Building**
  
  *GameObject template for building to be placed.*

- float **PlacementFrequency** = 0.5f
  
  *Frequency of building placement on a scale of 0 to 1.*

4.18.1 Detailed Description

Structure for accessing additional building placement configuration.

Definition at line 17 of file DistrictDefinition.cs.

4.18.2 Member Data Documentation
4.18.2.1 Building

GameObject DistrictConfiguration.BuildingTemplatePlacement.Building

GameObject template for building to be placed.
Definition at line 23 of file DistrictDefinition.cs.

4.18.2.2 PlacementFrequency

float DistrictConfiguration.BuildingTemplatePlacement.PlacementFrequency = 0.5f

Frequency of building placement on a scale of 0 to 1.
Definition at line 30 of file DistrictDefinition.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/DistrictDefinition.cs

4.19 CameraController Class Reference

Inheritance diagram for CameraController:

```
MonoBehaviour

CameraController
  + CameraPositions
  + EndingTriggered
  + CurrentView
  + ZoomLevel

  + RotateRight()
  + RotateLeft()
  + Zoom()
```

Generated by Doxygen
Collaboration diagram for CameraController:

Public Member Functions

- void RotateRight ()
  Rotates the camera to the next right position.
- void RotateLeft ()
  Rotates the camera to the next left position.
- void Zoom (float amount)
  Zoom in the camera by the amount.

Public Attributes

- Transform [] CameraPositions
- bool EndingTriggered = false

Properties

- Transform CurrentView [get]
  Gets the transform of the camera's current target view.
- float ZoomLevel [get]
  Gets the current level of zoom.

4.19.1 Detailed Description

Definition at line 4 of file CameraController.cs.
4.19.2 Member Function Documentation

4.19.2.1 RotateLeft()

```csharp
void CameraController.RotateLeft();
```

Rotates the camera to the next left position.

Definition at line 113 of file CameraController.cs.

4.19.2.2 RotateRight()

```csharp
void CameraController.RotateRight();
```

Rotates the camera to the next right position.

Definition at line 104 of file CameraController.cs.

4.19.2.3 Zoom()

```csharp
void CameraController.Zoom(float amount);
```

Zoom in the camera by the amount.

**Parameters**

- `amount` Amount to zoom.

Definition at line 126 of file CameraController.cs.

4.19.3 Member Data Documentation

4.19.3.1 CameraPositions

```csharp
Transform[] CameraController.CameraPositions
```

Definition at line 6 of file CameraController.cs.
4.19.3.2 EndingTriggered

```csharp
bool CameraController.EndingTriggered = false
```

Definition at line 36 of file CameraController.cs.

4.19.4 Property Documentation

4.19.4.1 CurrentView

```csharp
Transform CameraController.CurrentView [get]
```

Gets the transform of the camera's current target view.

Definition at line 77 of file CameraController.cs.

4.19.4.2 ZoomLevel

```csharp
float CameraController.ZoomLevel [get]
```

Gets the current level of zoom.

Definition at line 96 of file CameraController.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Camera/CameraController.cs

4.20 ChooseltemAmountPanelBehavior Class Reference

Item amount affect panel behavior.
Inheritance diagram for ChooseItemAmountPanelBehavior:

```
MonoBehaviour

ChooseItemAmountPanelBehavior
  + ItemNameDisplay
  + SelectedItem
  + Change()
  + OpenItemAmountPanel()
  + CancelChosenAmount()
  + CloseEntirePanel()
  + OpenItemDetailPanel()
  + ChooseNumOfItemsToAffect()
  + FinalizeAction()
```

Collaboration diagram for ChooseItemAmountPanelBehavior:

```
MonoBehaviour

Text

ChooseItemAmountPanelBehavior
  + SelectedItem
  + Change()
  + OpenItemAmountPanel()
  + CancelChosenAmount()
  + CloseEntirePanel()
  + OpenItemDetailPanel()
  + ChooseNumOfItemsToAffect()
  + FinalizeAction()```
Public Member Functions

- **void** Change (int amt)
  
  Change the number of units to affect. Updates the number displayed and the number saved in the backend.

- **void** OpenItemAmountPanel ()
  
  Opens the panel that allows users to select the desired number of items to affect with the action. Clears the information left behind from the last time it was open. Sets the max amount to be the number of units available from the selected item.

- **void** CancelChosenAmount ()
  
  Close the panel.

- **void** CloseEntirePanel ()
  
  Closes the entire item info panel.

- **void** OpenItemDetailPanel (GameObject selected)
  
  Opens the item detail panel.

- **void** ChooseNumOfItemsToAffect (ItemActionButtonUI actionButton)
  
  Chooses the number of items to affect.

- **void** FinalizeAction ()
  
  Executes the action.

Public Attributes

- Text **ItemNameDisplay**

Properties

- GameObject **SelectedItem** [get, set]

4.20.1 Detailed Description

Item amount affect panel behavior.

Definition at line 10 of file ChooseItemAmountPanelBehavior.cs.

4.20.2 Member Function Documentation

4.20.2.1 CancelChosenAmount()

**void** ChooseItemAmountPanelBehavior.CancelChosenAmount ( )

Close the panel.

Definition at line 147 of file ChooseItemAmountPanelBehavior.cs.

4.20.2.2 Change()

**void** ChooseItemAmountPanelBehavior.Change ( int amt )

Change the number of units to affect. Updates the number displayed and the number saved in the backend.
Parameters

| amt  | The amount to change. Typically either 1 or -1. |

Definition at line 71 of file ChooseItemAmountPanelBehavior.cs.

4.20.2.3 ChooseNumOfItemsToAffect()

```csharp
void ChooseItemAmountPanelBehavior.ChooseNumOfItemsToAffect (ItemActionButtonUI actionButton)
```

Chooses the number of items to affect.
Definition at line 198 of file ChooseItemAmountPanelBehavior.cs.

4.20.2.4 CloseEntirePanel()

```csharp
void ChooseItemAmountPanelBehavior.CloseEntirePanel ()
```

Closes the entire item info panel.
Definition at line 157 of file ChooseItemAmountPanelBehavior.cs.

4.20.2.5 FinalizeAction()

```csharp
void ChooseItemAmountPanelBehavior.FinalizeAction ()
```

Executes the action.
Definition at line 207 of file ChooseItemAmountPanelBehavior.cs.

4.20.2.6 OpenItemAmountPanel()

```csharp
void ChooseItemAmountPanelBehavior.OpenItemAmountPanel ()
```

Opens the panel that allows users to select the desired number of items to affect with the action. Clears the information left behind from the last time it was open. Sets the max amount to be the number of units available from the selected item.
Definition at line 112 of file ChooseItemAmountPanelBehavior.cs.

4.20.2.7 OpenItemDetailPanel()

```csharp
void ChooseItemAmountPanelBehavior.OpenItemDetailPanel (GameObject selected)
```

Opens the item detail panel.
4.21 City Class Reference

Parameters

| selected | Selected |

Definition at line 166 of file ChooseItemAmountPanelBehavior.cs.

4.20.3 Member Data Documentation

4.20.3.1 ItemNameDisplay

`Text ChooseItemAmountPanelBehavior.ItemNameDisplay`

Definition at line 34 of file ChooseItemAmountPanelBehavior.cs.

4.20.4 Property Documentation

4.20.4.1 SelectedItem

`GameObject ChooseItemAmountPanelBehavior.SelectedItem [get], [set]`

Definition at line 52 of file ChooseItemAmountPanelBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/InventoryGUI/ChooseItemAmountPanelBehavior.cs

4.21 City Class Reference

Collaboration diagram for City:

<table>
<thead>
<tr>
<th>City</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Districts</td>
</tr>
<tr>
<td>+ BoundingBox</td>
</tr>
<tr>
<td>+ Center</td>
</tr>
<tr>
<td>+ TallestBuilding</td>
</tr>
<tr>
<td>+ City()</td>
</tr>
<tr>
<td>+ GetEnumerator()</td>
</tr>
</tbody>
</table>
Classes

- class BuildingEnumerator
  
  Class for creating an iterator to traverse the tree-structure of buildings in the City.

Public Member Functions

- City (District[] districts, Bounds boundingBox, Vector3 cityCenter, Building tallestBuilding)
  
  Creates a city.

- IEnumerator<Building> GetEnumerator ()
  
  Get an iterator that traverses tree-structure of buildings in the city.

Properties

- District[] Districts [get]
  
  The list of districts contained in the city.

- Bounds BoundingBox [get]
  
  The bounding box that defines the size of the city.

- Vector3 Center [get]
  
  The position of the center of the city.

- Building TallestBuilding [get]
  
  The tallest building located at the city center.

4.21.1 Detailed Description

Definition at line 5 of file City.cs.

4.21.2 Constructor & Destructor Documentation

4.21.2.1 City()

City.City (District[] districts, Bounds boundingBox, Vector3 cityCenter, Building tallestBuilding)

Creates a city.

Parameters

<table>
<thead>
<tr>
<th>districts</th>
<th>The list of districts contained in the city.</th>
</tr>
</thead>
<tbody>
<tr>
<td>boundingBox</td>
<td>The bounding box that defines the size of the city.</td>
</tr>
<tr>
<td>cityCenter</td>
<td>The location of the city center.</td>
</tr>
<tr>
<td>tallestBuilding</td>
<td>The instance of the tallest building.</td>
</tr>
</tbody>
</table>
4.21 City Class Reference

Definition at line 16 of file City.cs.

4.21.3 Member Function Documentation

4.21.3.1 GetEnumerator()

IEnumerator<Building> City.GetEnumerator ()

Get an iterator that traverses tree-structure of buildings in the city.

Returns

Building iterator.

Definition at line 145 of file City.cs.

4.21.4 Property Documentation

4.21.4.1 BoundingBox

Bounds City.BoundingBox [get]

The bounding box that defines the size of the city.

Definition at line 37 of file City.cs.

4.21.4.2 Center

Vector3 City.Center [get]

The position of the center of the city.

Definition at line 46 of file City.cs.

4.21.4.3 Districts

District [] City.Districts [get]

The list of districts contained in the city.

Definition at line 28 of file City.cs.
4.21.4.4 TallestBuilding

**Building City.TallestBuilding [get]**

The tallest building located at the city center.

Definition at line 55 of file City.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/City.cs

4.22 CityBoundaries Class Reference

Collaboration diagram for CityBoundaries:

```
CityBoundaries
+ CityBounds
+ RandomVector2d
+ RandomVector3d
+ CityBoundaries()
+ BoundVector2d()
+ BoundVector3d()
```

Public Member Functions

- **CityBoundaries ()**

  Initializes a new instance of the CityBoundaries class.

- Vector2 **BoundVector2d** (Vector2 vector)

  Takes a vector2 and clamps its values so it cannot extend outside of the city's boundaries.

- Vector3 **BoundVector3d** (Vector3 vector)

  Takes a vector3 and clamps its values so it cannot extend outside of the city's boundaries.

Public Attributes

- **Bounds CityBounds**

  The city bounds.
Properties

- Vector2 RandomVector2d [get]
  
  Gets a random Vector2 within the city’s boundaries.

- Vector3 RandomVector3d [get]
  
  Gets a random Vector3 within the city’s boundaries.

4.22.1 Detailed Description

Definition at line 3 of file CityBoundaries.cs.

4.22.2 Constructor & Destructor Documentation

4.22.2.1 CityBoundaries()

CityBoundaries.CityBoundaries ( )

Initializes a new instance of the CityBoundaries class.

Definition at line 13 of file CityBoundaries.cs.

4.22.3 Member Function Documentation

4.22.3.1 BoundVector2d()

Vector2 CityBoundaries.BoundVector2d ( Vector2 vector )

Takes a vector2 and clamps its values so it cannot extend outside of the city's boundaries.

Returns

The vector.

Parameters

vector Vector.

Definition at line 55 of file CityBoundaries.cs.
4.22.3.2 BoundVector3d()

```csharp
Vector3 CityBoundaries.BoundVector3d (Vector3 vector)
```

Takes a vector3 and clamps its values so it cannot extend outside of the city's boundaries.

Returns
The vector3d.

Parameters
- `vector` Vector.

Definition at line 67 of file CityBoundaries.cs.

4.22.4 Member Data Documentation

4.22.4.1 CityBounds

```csharp
Bounds CityBoundaries.CityBounds
```

The city bounds.

Definition at line 8 of file CityBoundaries.cs.

4.22.5 Property Documentation

4.22.5.1 RandomVector2d

```csharp
Vector2 CityBoundaries.RandomVector2d [get]
```

Gets a random Vector2 within the city's boundaries.

The random vector.

Definition at line 27 of file CityBoundaries.cs.
4.23 CityChunk Class Reference

Manages loading and unloading a chunk of buildings

Collaboration diagram for CityChunk:

<table>
<thead>
<tr>
<th>CityChunk</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Location</td>
</tr>
<tr>
<td>+ BoundingBox</td>
</tr>
<tr>
<td>+ Buildings</td>
</tr>
<tr>
<td>+ IsLoaded</td>
</tr>
<tr>
<td>+ CurrentCoroutine</td>
</tr>
<tr>
<td>+ CityChunk()</td>
</tr>
<tr>
<td>+ Load()</td>
</tr>
<tr>
<td>+ Unload()</td>
</tr>
</tbody>
</table>

Public Member Functions

- **CityChunk** (int x, int y, Bounds bounds)
  
  Creates in instance of CityChunk.

- **IEnumerator Load()**

  Asynchronously loads buildings.

- **IEnumerator Unload()**

  Asynchronously unloads buildings.
Properties

- **Tuple<int, int> Location [get]**
  Location of this chunk in the chunk grid.
- **Bounds BoundingBox [get]**
  Bounds defining chunk coverage
- **List<Building> Buildings [get]**
  List of buildings managed by this chunk.
- **bool IsLoaded [get]**
  True if the chunk has been set to be loaded.
- **IEnumerator CurrentCoroutine [get, set]**
  Current loading or unloading co-routine. Returns null if there is no current co-routine.

4.23.1 Detailed Description

Manages loading and unloading a chunk of buildings

Definition at line 8 of file CityChunk.cs.

4.23.2 Constructor & Destructor Documentation

4.23.2.1 CityChunk()

CityChunk.CityChunk (  
  int x,  
  int y,  
  Bounds bounds )

Creates an instance of CityChunk.

Parameters

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>x</strong></td>
<td>x-axis coordinate of grid.</td>
</tr>
<tr>
<td><strong>y</strong></td>
<td>z-axis coordinate of grid.</td>
</tr>
<tr>
<td><strong>bounds</strong></td>
<td>Area covered by chunk.</td>
</tr>
</tbody>
</table>

Definition at line 16 of file CityChunk.cs.

4.23.3 Member Function Documentation
4.23.3.1 Load()

IEnumerator CityChunk.Load()

Asyncronously loads buildings.
Definition at line 72 of file CityChunk.cs.

4.23.3.2 Unload()

IEnumerator CityChunk.Unload()

Asyncronously unloads buildings.
Definition at line 91 of file CityChunk.cs.

4.23.4 Property Documentation

4.23.4.1 BoundingBox

Bounds CityChunk.BoundingBox [get]

Bounds defining chunk coverage
Definition at line 37 of file CityChunk.cs.

4.23.4.2 Buildings

List<Building> CityChunk.Buildings [get]

List of buildings managed by this chunk.
Definition at line 46 of file CityChunk.cs.

4.23.4.3 CurrentCoroutine

IEnumerator CityChunk.CurrentCoroutine [get], [set]

Current loading or unloading co-routine. Returns null if there is no current co-routine.
Definition at line 64 of file CityChunk.cs.
4.23.4.4 IsLoaded

bool CityChunk.IsLoaded [get]

True of the chunk has been set to be loaded.
Definition at line 55 of file CityChunk.cs.

4.23.4.5 Location

Tuple<int, int> CityChunk.Location [get]

Location of this chunk in the chunk grid.
Definition at line 28 of file CityChunk.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/Management/CityChunk.cs

4.24 CityChunkManager Class Reference

Used to manage loading and unloading city building chunks according to player location in world.

Inheritance diagram for CityChunkManager:

![Inheritance Diagram](image-url)
Collaboration diagram for CityChunkManager:

Public Member Functions

- void `Init(City city)`
  
  **Divides city building into chunks.**

Properties

- float `ChunkSize` [get]
  
  **Size of chunks across.**

- int `ChunksAwayToLoad` [get]
  
  **Number of chunks away to load.**

- int `ChunksAcross` [get]
  
  **Number of chunks to cover x-axis extent of city.**

- int `ChunksDown` [get]
  
  **Number of chunks to cover y-axis of city.**

- `CityChunk[,] Chunks` [get]
  
  **The grid of city chunks.**

- bool `FreezeLoading` [get, set]
  
  **If true, freezes city loading and unloading.**

- `Tuple<int, int> CurrentChunk` [get, set]
  
  **Coordinate location of the chunk the player is currently in.**

### 4.24.1 Detailed Description

Used to manage loading and unloading city building chunks according to player location in world.

Definition at line 8 of file CityChunkManager.cs.
4.24.2 Member Function Documentation

4.24.2.1 Init()

```csharp
void CityChunkManager.Init ( City city )
```

Divides city building into chunks.

**Parameters**

| city | Initialized city. |

Definition at line 34 of file CityChunkManager.cs.

4.24.3 Property Documentation

4.24.3.1 Chunks

```csharp
CityChunk[,] CityChunkManager.Chunks [get]
```

The grid of city chunks.

Definition at line 125 of file CityChunkManager.cs.

4.24.3.2 ChunksAcross

```csharp
int CityChunkManager.ChunksAcross [get]
```

Number of chunks to cover x-axis extent of city.

Definition at line 107 of file CityChunkManager.cs.

4.24.3.3 ChunksAwayToLoad

```csharp
int CityChunkManager.ChunksAwayToLoad [get]
```

Number of chunks away to load.

Definition at line 98 of file CityChunkManager.cs.
4.24.3.4 ChunksDown

```csharp
int CityChunkManager.ChunksDown [get];
```

Number of chunks to cover y-axis of city.

Definition at line 116 of file CityChunkManager.cs.

4.24.3.5 ChunkSize

```csharp
float CityChunkManager.ChunkSize [get];
```

Size of chunks across.

Definition at line 89 of file CityChunkManager.cs.

4.24.3.6 CurrentChunk

```csharp
Tuple<int, int> CityChunkManager.CurrentChunk [get], [set];
```

Coordinate location of the chunk the player is currently in.

Definition at line 143 of file CityChunkManager.cs.

4.24.3.7 FreezeLoading

```csharp
bool CityChunkManager.FreezeLoading [get], [set];
```

If true, freezes city loading and unloading.

Definition at line 134 of file CityChunkManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/Management/CityChunkManager.cs
4.25 CityController Class Reference

Inheritance diagram for CityController:

```
Class Diagram

MonoBehaviour

CityController
```

Collaboration diagram for CityController:

```
Class Diagram

MonoBehaviour

CityController
```

4.25.1 Detailed Description

Definition at line 5 of file CityController.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/CityController.cs
Inheritance diagram for Clock:

```
MonoBehaviour

Clock
+ Tick
+ TwentyFourHours
+ TwelveHours
+ Hour
+ HalfHour
+ AnglePerSecond
+ CurrentTime
+ CurrentGameTimeInHours
+ FormattedTime
+ IsDay

+ SecondDelegateUpdate()
+ HalfHourDelegateUpdate()
+ HourDelegateUpdate()
+ GetTwelveHourPercentage()
```
Collaboration diagram for Clock:

```
MonoBehaviour
 |
 |
 |
 |
Clock
+ Tick
+ TwentyFourHours
+ TwelveHours
+ Hour
+ HalfHour
+ AnglePerSecond
+ CurrentTime
+ CurrentGameTimeInHours
+ FormattedTime
+ IsDay
+ SecondDelegateUpdate()
+ HalfHourDelegateUpdate()
+ HourDelegateUpdate()
+ GetTwelveHourPercentage()
```

Public Member Functions

- delegate void SecondDelegateUpdate ()
- delegate void HalfHourDelegateUpdate ()
- delegate void HourDelegateUpdate ()
- float GetTwelveHourPercentage ()

  Gets the percentage that of twelve hours that the current time is representative of. This is useful for lerping and other related functions.

Properties

- float Tick  [get]
- float TwentyFourHours  [get]

  The twenty four hours.

- float TwelveHours  [get]

  The twelve hours.

- float Hour  [get]

  Gets the hour.

- float HalfHour  [get]

  Half an hour of game time in real world seconds

- float AnglePerSecond  [get]
4.26 Clock Class Reference

Gets the angle per second based on time scale.

- float `CurrentTime` [get]
  
  Gets the current time.

- float `CurrentGameTimeInHours` [get]
  
  Gets the current game time in hours.

- string `FormattedTime` [get]
  
  Gets the formatted time of the game clock.

- bool `IsDay` [get]
  
  Determines whether it is day night.

Events

- `SecondDelegateUpdate SecondUpdate`
- `HalfHourDelegateUpdate HalfHourUpdate`
- `HourDelegateUpdate HourUpdate`

4.26.1 Detailed Description

Definition at line 5 of file Clock.cs.

4.26.2 Member Function Documentation

4.26.2.1 GetTwelveHourPercentage()

```csharp
float Clock.GetTwelveHourPercentage() {
    // Implementation
}
```

Gets the percentage that of twelve hours that the current time is representative of. This is useful for lerping and other related functions.

Returns

The twelve hour percentage.

Definition at line 216 of file Clock.cs.

4.26.2.2 HalfHourDelegateUpdate()

```csharp
delegate void Clock.HalfHourDelegateUpdate();
```

Generated by Doxygen
4.26.2.3 HourDelegateUpdate()

delegate void Clock.HourDelegateUpdate();

4.26.2.4 SecondDelegateUpdate()

delegate void Clock.SecondDelegateUpdate();

4.26.3 Property Documentation

4.26.3.1 AnglePerSecond

float Clock.AnglePerSecond [get]

Gets the angle per second based on time scale.
The angle per second.
Definition at line 66 of file Clock.cs.

4.26.3.2 CurrentGameTimeInHours

float Clock.CurrentGameTimeInHours [get]

Gets the current game time in hours.
The current game time in hours.
Definition at line 86 of file Clock.cs.

4.26.3.3 CurrentTime

float Clock.CurrentTime [get]

Gets the current time.
The current time.
Definition at line 76 of file Clock.cs.
4.26 Clock Class Reference

4.26.3.4 FormattedTime

string Clock.FormattedTime [get]

Gets the formatted time of the game clock.

The formatted time.

Definition at line 134 of file Clock.cs.

4.26.3.5 HalfHour

float Clock.HalfHour [get]

Half an hour of game time in real world seconds

The half hour.

Definition at line 56 of file Clock.cs.

4.26.3.6 Hour

float Clock.Hour [get]

Gets the hour.

The hour.

Definition at line 46 of file Clock.cs.

4.26.3.7 IsDay

bool Clock.IsDay [get]

Determines whether this it is or day night.

Returns

ture if this instance is day; otherwise, false.

Definition at line 203 of file Clock.cs.
4.26.3.8  Tick

float Clock.Tick  [get]

Definition at line 18 of file Clock.cs.

4.26.3.9  TwelveHours

float Clock.TwelveHours  [get]

The twelve hours.

Definition at line 36 of file Clock.cs.

4.26.3.10  TwentyFourHours

float Clock.TwentyFourHours  [get]

The twenty four hours.

Definition at line 27 of file Clock.cs.

4.26.4  Event Documentation

4.26.4.1  HalfHourUpdate

HalfHourDelegateUpdate Clock.HalfHourUpdate

Definition at line 98 of file Clock.cs.

4.26.4.2  HourUpdate

HourDelegateUpdate Clock.HourUpdate

Definition at line 101 of file Clock.cs.
4.26.4.3 SecondUpdate

_definition at line 95 of file Clock.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/Clock.cs

4.27 ClothCategory Class Reference

Inheritance diagram for ClothCategory:
Collaboration diagram for ClothCategory:

Public Member Functions

- override ItemCategory GetDuplicate()  
  Gets a copy of the ItemCategory.
- override void ReadyCategory()  
  Preps the category for use by loading attributes and actions into lists.
- void PutOn()
Puts on the cloth item. TODO: Make this similar to how equipping works.

- void TakeOff ()
  Takes the cloth item off. TODO: Make this similar to how unequipping works.

Protected Attributes

- const string onPlayerAttributeName = "onPlayer"

Properties

- float ThreadDensity [get, set]
  Gets or sets the thread density.

- float FabricThickness [get, set]
  Gets or sets the fabric thickness.

- float Impermiability [get, set]
  Gets or sets the impermiability.

- float OnPlayer [get, set]
  Gets or sets the on player.

Additional Inherited Members

4.27.1 Detailed Description

Definition at line 7 of file ClothCategory.cs.

4.27.2 Member Function Documentation

4.27.2.1 GetDuplicate()

override ItemCategory ClothCategory.GetDuplicate ( ) [virtual]

Gets a copy of the ItemCategory.

Returns

The duplicate.

Reimplemented from ItemCategory.

Definition at line 69 of file ClothCategory.cs.
4.27.2.2 PutOn()

```csharp
void ClothCategory.PutOn()
```

Puts on the cloth item. TODO: Make this similar to how equipping works.

Definition at line 121 of file ClothCategory.cs.

4.27.2.3 ReadyCategory()

```csharp
override void ClothCategory.ReadyCategory() [virtual]
```

Preps the category for use by loading attributes and actions into lists.

Reimplemented from ItemCategory.

Definition at line 94 of file ClothCategory.cs.

4.27.2.4 TakeOff()

```csharp
void ClothCategory.TakeOff()
```

Takes the cloth item off. TODO: Make this similar to how unequipping works.

Definition at line 136 of file ClothCategory.cs.

4.27.3 Member Data Documentation

4.27.3.1 onPlayerAttributeName

```csharp
const string ClothCategory.onPlayerAttributeName = "onPlayer" [protected]
```

Definition at line 63 of file ClothCategory.cs.

4.27.4 Property Documentation
4.27 ClothCategory Class Reference

4.27.4.1 FabricThickness

float ClothCategory.FabricThickness [get], [set]

Gets or sets the fabric thickness.

The fabric thickness.

Definition at line 24 of file ClothCategory.cs.

4.27.4.2 Impermiability

float ClothCategory.Impermiability [get], [set]

Gets or sets the impermiability.

The impermiability.

Definition at line 34 of file ClothCategory.cs.

4.27.4.3 OnPlayer

float ClothCategory.OnPlayer [get], [set]

Gets or sets the on player.

The on player.

Definition at line 44 of file ClothCategory.cs.

4.27.4.4 ThreadDensity

float ClothCategory.ThreadDensity [get], [set]

Gets or sets the thread density.

The thread density.

Definition at line 14 of file ClothCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ClothCategory.cs
4.28 CollectableItem Class Reference

Abstract class that is inherited by all BaseItem classes and Item Categories

Inheritance diagram for CollectableItem:

Collaboration diagram for CollectableItem:

Public Member Functions

- virtual List<ItemAction> GetPossibleActions ()
  
  Gets all the action that an item can perform.

4.28.1 Detailed Description

Abstract class that is inherited by all BaseItem classes and Item Categories

Definition at line 8 of file CollectableItem.cs.
4.28.2 Member Function Documentation

4.28.2.1 GetPossibleActions()

virtual List<ItemAction> CollectableItem.GetPossibleActions() [virtual]

Gets all the action that an item can performe.

Returns

The possible actions of an item in a Dictionary keyed by the action name.

Reimplemented in BaseItem, and ItemCategory.

Definition at line 15 of file CollectableItem.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/CollectableItem.cs

4.29 ConsoleCommandRouter Class Reference

Inheritance diagram for ConsoleCommandRouter:
Public Member Functions

- string **Test** (params string[] args)
  
  Testing console with very important command

- string **DebugMode** (params string[] args)
  
  Calls debug movement from the console

- string **DebugWeatherGUI** (params string[] args)
  
  Opens debug weather gui

- string **SetTimeScale** (params string[] args)
  
  Set time scale for game

- string **ActivatePressureSystem** (params string[] args)
  
  Activates the pressure system.

- string **DebugCreatureCountGUI** (params string[] args)
  
  Opens debug creature count gui

- string **PrintWeather** (params string[] args)
  
  Prints the weather.

- string **TeleportToEnding** (params string[] args)
  
  Teleports the player to a gameobject.

- string **SetHealth** (params string[] args)
  
  Sets the player's health to a number

4.29.1 Detailed Description

Definition at line 7 of file ConsoleCommandRouter.cs.
4.29.2 Member Function Documentation

4.29.2.1 ActivatePressureSystem()

```csharp
string ConsoleCommandRouter.ActivatePressureSystem (params string[] args)
```

Activates the pressure system.

**Returns**

The pressure system.

**Parameters**

- **args** Arguments.

Definition at line 122 of file ConsoleCommandRouter.cs.

4.29.2.2 DebugCreatureCountGUI()

```csharp
string ConsoleCommandRouter.DebugCreatureCountGUI (params string[] args)
```

Opens debug creature count gui

**Returns**

The weather GUI.

**Parameters**

- **args** Arguments.

Definition at line 142 of file ConsoleCommandRouter.cs.

4.29.2.3 DebugMode()

```csharp
string ConsoleCommandRouter.DebugMode (params string[] args)
```

Calls debug movement from the console
Returns

The movement.

Parameters

| args | Arguments |

Definition at line 67 of file ConsoleCommandRouter.cs.

4.29.2.4 DebugWeatherGUI()

```csharp
string ConsoleCommandRouter.DebugWeatherGUI (  
    params string [][] args )
```

Opens debug weather gui

Returns

The weather GUI.

Parameters

| args | Arguments |

Definition at line 86 of file ConsoleCommandRouter.cs.

4.29.2.5 PrintWeather()

```csharp
string ConsoleCommandRouter.PrintWeather (  
    params string [] args )
```

Prints the weather.

Returns

The weather.

Parameters

| args | Arguments |

Definition at line 173 of file ConsoleCommandRouter.cs.
4.29.2.6 SetHealth()

string ConsoleCommandRouter.SetHealth (  
    params string [] args )

Sets the player's health to a number

Parameters

<table>
<thead>
<tr>
<th>args</th>
<th>Arguments</th>
</tr>
</thead>
</table>

Definition at line 194 of file ConsoleCommandRouter.cs.

4.29.2.7 SetTimeScale()

string ConsoleCommandRouter.SetTimeScale (  
    params string [] args )

Set time scale for game

Returns

Validation time scale was set

Parameters

<table>
<thead>
<tr>
<th>args</th>
<th>Arguments</th>
</tr>
</thead>
</table>

Definition at line 109 of file ConsoleCommandRouter.cs.

4.29.2.8 TeleportToEnding()

string ConsoleCommandRouter.TeleportToEnding (  
    params string [] args )

Teleports the player to a gameobject.

Parameters

<table>
<thead>
<tr>
<th>args</th>
<th>Arguments</th>
</tr>
</thead>
</table>

Definition at line 183 of file ConsoleCommandRouter.cs.
4.29.2.9 Test()

```csharp
string ConsoleCommandRouter.Test (  
    params string [] args)
```

Testing console with very important command

Parameters

| args | Arguments. |

Definition at line 57 of file `ConsoleCommandRouter.cs`.
The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/ConsoleCommandRouter.cs

4.30 ContainerCategory Class Reference

Inheritance diagram for ContainerCategory:
Collaboration diagram for ContainerCategory:

Public Member Functions

- **override ItemCategory GetDuplicate()**
  
  Gets a copy of the ItemCategory.

- **override void ReadyCategory()**
  
  Preps the category for use by loading attributes and actions into lists.

- **void SetDown()**
  
  Sets down the container in the world. Drops it where the player stands.
Properties

- int **Size** [get, set]

Additional Inherited Members

4.30.1 Detailed Description

Definition at line 7 of file ContainerCategory.cs.

4.30.2 Member Function Documentation

4.30.2.1 GetDuplicate()

```csharp
override ItemCategory ContainerCategory.GetDuplicate () [virtual]
```

Gets a copy of the **ItemCategory**.

Returns

The duplicate.

Reimplemented from **ItemCategory**.

Definition at line 22 of file ContainerCategory.cs.

4.30.2.2 ReadyCategory()

```csharp
override void ContainerCategory.ReadyCategory () [virtual]
```

Preps the category for use by loading attributes and actions into lists.

Reimplemented from **ItemCategory**.

Definition at line 41 of file ContainerCategory.cs.

4.30.2.3 SetDown()

```csharp
void ContainerCategory.SetDown ()
```

Sets down the container in the world. Drops it where the player stands.

Definition at line 55 of file ContainerCategory.cs.
4.30.3 Property Documentation

4.30.3.1 Size

```csharp
int ContainerCategory.Size [get], [set]
```

Definition at line 10 of file ContainerCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ContainerCategory.cs

4.31 ControlScheme Class Reference

Collaboration diagram for ControlScheme:

![Collaboration Diagram](image)

Public Attributes

- KeyCode Forward
- KeyCode Back
- KeyCode Right
- KeyCode Left
- KeyCode ForwardSecondary
- KeyCode BackSecondary
- KeyCode RightSecondary
- KeyCode LeftSecondary
- KeyCode Sprint
- KeyCode Jump
and 11 more...

Generated by Doxygen
4.31.1 Detailed Description

Definition at line 5 of file ControlScheme.cs.

4.31.2 Member Data Documentation

4.31.2.1 Action

KeyCode ControlScheme.Action

Definition at line 19 of file ControlScheme.cs.

4.31.2.2 Back

KeyCode ControlScheme.Back

Definition at line 8 of file ControlScheme.cs.

4.31.2.3 BackSecondary

KeyCode ControlScheme.BackSecondary

Definition at line 13 of file ControlScheme.cs.
4.31.2.4 CameraLeft

KeyCode ControlScheme.CameraLeft

Definition at line 23 of file ControlScheme.cs.

4.31.2.5 CameraRight

KeyCode ControlScheme.CameraRight

Definition at line 22 of file ControlScheme.cs.

4.31.2.6 CameraZoomAxis

string ControlScheme.CameraZoomAxis

Definition at line 24 of file ControlScheme.cs.

4.31.2.7 CameraZoomInKey

KeyCode ControlScheme.CameraZoomInKey

Definition at line 25 of file ControlScheme.cs.

4.31.2.8 CameraZoomOutKey

KeyCode ControlScheme.CameraZoomOutKey

Definition at line 26 of file ControlScheme.cs.

4.31.2.9 Crafting

KeyCode ControlScheme.Crafting

Definition at line 31 of file ControlScheme.cs.
4.31.2.10 Forward

KeyCode ControlScheme.Forward

Definition at line 7 of file ControlScheme.cs.

4.31.2.11 ForwardSecondary

KeyCode ControlScheme.ForwardSecondary

Definition at line 12 of file ControlScheme.cs.

4.31.2.12 Inventory

KeyCode ControlScheme.Inventory

Definition at line 29 of file ControlScheme.cs.

4.31.2.13 Jump

KeyCode ControlScheme.Jump

Definition at line 18 of file ControlScheme.cs.

4.31.2.14 Left

KeyCode ControlScheme.Left

Definition at line 10 of file ControlScheme.cs.

4.31.2.15 LeftSecondary

KeyCode ControlScheme.LeftSecondary

Definition at line 15 of file ControlScheme.cs.
4.31.2.16  Pause

KeyCode ControlScheme.Pause

Definition at line 28 of file ControlScheme.cs.

4.31.2.17  Radio

KeyCode ControlScheme.Radio

Definition at line 30 of file ControlScheme.cs.

4.31.2.18  Right

KeyCode ControlScheme.Right

Definition at line 9 of file ControlScheme.cs.

4.31.2.19  RightSecondary

KeyCode ControlScheme.RightSecondary

Definition at line 14 of file ControlScheme.cs.

4.31.2.20  Sprint

KeyCode ControlScheme.Sprint

Definition at line 17 of file ControlScheme.cs.

4.31.2.21  UseTool

KeyCode ControlScheme.UseTool

Definition at line 20 of file ControlScheme.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Player/ControlScheme.cs
4.32 CraftingStat Class Reference

Collaboration diagram for CraftingStat:

Properties

- string StatName [get, set]
  the name of the stat to be considered during crafting
- List<string> StatAffectingItems [get, set]
  the items (by tag) that will be considered when finding the result of the stat. For example, for a fishing rod, the "Elasticity" stat only takes into account the rod and rope items, not hook, so the AffectItems for that would be ["rod", "rope"]
- List<float> QualityThreshold [get, set]
  the values that the item must reach in order to be considered good or excellent. Failing to reach either of these values will result in a "poor" item

4.32.1 Detailed Description

Definition at line 4 of file CraftingStat.cs.

4.32.2 Property Documentation

4.32.2.1 QualityThreshold

List<float> CraftingStat.QualityThreshold [get], [set]
the values that the item must reach in order to be considered good or excellent. Failing to reach either of these values will result in a "poor" item

The threshold.
Definition at line 34 of file CraftingStat.cs.
4.32.2 StatAffectingItems

\texttt{List\lang{string} CraftingStat.StatAffectingItems [get], [set]}

the items (by tag) that will be considered when finding the result of the stat. For example, for a fishing rod, the "Elasticity" stat only takes into account the rod and rope items, not hook, so the AffectItems for that would be \texttt{["rod", "rope"]}

The affecting items.

Definition at line 23 of file CraftingStat.cs.

4.32.3 StatName

\texttt{string CraftingStat.StatName [get], [set]}

the name of the stat to be considered during crafting

The name of the stat.

Definition at line 11 of file CraftingStat.cs.

The documentation for this class was generated from the following file:

- Assets\Scripts\Item Crafting\Backend\CraftingStat.cs

4.33 CraftingSystemSerializer Class Reference

Inheritance diagram for CraftingSystemSerializer:
Collaboration diagram for CraftingSystemSerializer:

```plaintext
CraftingSystemSerializer
+ Filename
# categoryNames
# categoryTypes
# uriPrefix
+ SetUpCategoryInformation()
```

Public Member Functions

- `void SetUpCategoryInformation()`
  
  Fills out the categoryNames and categoryTypes lists with the necessary information

Public Attributes

- `string Filename`

Protected Attributes

- `List<string> categoryNames`
  
  the possible categories that can be added to a baseItem categoryNames stores name of the category
- `List<Type> categoryTypes`
- `string uriPrefix = "tag:yaml.org,2002:"
  
  tag:yaml.org,2002 is shorthanded as "!" in the yaml file, but when registering the tag, it is necessary to use the full Uri

4.33.1 Detailed Description

Definition at line 10 of file CraftingSystemSerializer.cs.

4.33.2 Member Function Documentation

4.33.2.1 SetUpCategoryInformation()

`void CraftingSystemSerializer.SetUpCategoryInformation()`

Fills out the categoryNames and categoryTypes lists with the necessary information

Definition at line 49 of file CraftingSystemSerializer.cs.
4.33 CraftingSystemSerializer Class Reference

4.33.3 Member Data Documentation

4.33.3.1 categoryNames

List<string> CraftingSystemSerializer.categoryNames [protected]

the possible categories that can be added to a baseItem categoryNames stores name of the category
Types stores the type

Definition at line 19 of file CraftingSystemSerializer.cs.

4.33.3.2 categoryTypes

List<Type> CraftingSystemSerializer.categoryTypes [protected]

Definition at line 20 of file CraftingSystemSerializer.cs.

4.33.3.3 Filename

string CraftingSystemSerializer.Filename

Definition at line 12 of file CraftingSystemSerializer.cs.

4.33.3.4 uriPrefix

string CraftingSystemSerializer.uriPrefix = "tag:yaml.org,2002:" [protected]

tag:yaml.org,2002 is shorthanded as "!" in the yaml file, but when registering the tag, it is necessary to use the full
Uri

Definition at line 44 of file CraftingSystemSerializer.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/CraftingSystemSerializer.cs

Generated by Doxygen
4.34 CreatureManager Class Reference

Collaboration diagram for CreatureManager:

```
<table>
<thead>
<tr>
<th>CreatureManager</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ IdealCreatureCount</td>
</tr>
<tr>
<td>+ CreatureCount</td>
</tr>
<tr>
<td>+ Spawn()</td>
</tr>
<tr>
<td>+ PutCreatureInPool()</td>
</tr>
<tr>
<td>+ UpdateCreatureInfo()</td>
</tr>
<tr>
<td>+ UpdateCreatureCount()</td>
</tr>
<tr>
<td>+ CreatureManager()</td>
</tr>
</tbody>
</table>
```

Public Member Functions

- **void** `Spawn(Vector3 position, GameObject creature)`  
  *Spawn the specified position and creature.*
- **void** `PutCreatureInPool(int creatureIndex)`  
  *Puts creature into the pool.*
- **void** `UpdateCreatureInfo(float radius, float waterLevelOffset)`  
  *Checks the creature position and kills them if they aren't in the radius*
- **void** `UpdateCreatureCount(Vector3 position, GameObject creature)`  
  *Updates the creature count by spawning more creatures if we are below the ideal creature count. If we are above time will eventually remedy this.*
- **CreatureManager**(int maxCount, int idealCount)  
  *Initializes a new instance of the CreatureManager class.*

Properties

- **int** `IdealCreatureCount [get, set]`  
  *The ideal amount of creatures in the scene*
- **int** `CreatureCount [get]`  
  *Count of how many creatures are in the scene*

4.34.1 Detailed Description

Definition at line 5 of file CreatureManager.cs.

4.34.2 Constructor & Destructor Documentation

Generated by Doxygen
4.34.2.1 CreatureManager()

CreatureManager.CreatureManager (  
    int maxCount,  
    int idealCount )

Initializes a new instance of the CreatureManager class.

Parameters

<table>
<thead>
<tr>
<th>maxCount</th>
<th>Max count.</th>
</tr>
</thead>
<tbody>
<tr>
<td>idealCount</td>
<td>Ideal count.</td>
</tr>
</tbody>
</table>

Definition at line 158 of file CreatureManager.cs.

4.34.3 Member Function Documentation

4.34.3.1 PutCreatureInPool()

void CreatureManager.PutCreatureInPool (  
    int creatureIndex )

Puts creature into the pool.

Parameters

| creature | Creature. |

Definition at line 88 of file CreatureManager.cs.

4.34.3.2 Spawn()

void CreatureManager.Spawn (  
    Vector3 position,  
    GameObject creature )

Spawn the specified position and creature.

Parameters

<table>
<thead>
<tr>
<th>position</th>
<th>Position.</th>
</tr>
</thead>
<tbody>
<tr>
<td>creature</td>
<td>Creature.</td>
</tr>
</tbody>
</table>

Definition at line 44 of file CreatureManager.cs.
4.34.3.3 UpdateCreatureCount()

```csharp
void CreatureManager.UpdateCreatureCount(
    Vector3 position,
    GameObject creature)
```

Updates the creature count by spawning more creatures if we are below the ideal creature count. If we are above time will eventually remedy this.

**Parameters**

<table>
<thead>
<tr>
<th>position</th>
<th>Position.</th>
</tr>
</thead>
<tbody>
<tr>
<td>creature</td>
<td>Creature.</td>
</tr>
</tbody>
</table>

Definition at line 145 of file CreatureManager.cs.

4.34.3.4 UpdateCreatureInfo()

```csharp
void CreatureManager.UpdateCreatureInfo(
    float radius,
    float waterLevelOffset)
```

Checks the creature position and kills them if they aren't in the radius

**Parameters**

<table>
<thead>
<tr>
<th>radius</th>
<th>Radius.</th>
</tr>
</thead>
<tbody>
<tr>
<td>waterLevelOffset</td>
<td>Water level offset.</td>
</tr>
</tbody>
</table>

Definition at line 106 of file CreatureManager.cs.

4.34.4 Property Documentation

4.34.4.1 CreatureCount

```csharp
int CreatureManager.CreatureCount [get]
```

Count of how many creatures are in the scene

The creature count.

Definition at line 34 of file CreatureManager.cs.
4.34.4.2 IdealCreatureCount

```csharp
int CreatureManager.IdealCreatureCount [get], [set]
```

The ideal amount of creatures in the scene

Definition at line 24 of file CreatureManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Creatures/CreatureManager.cs

### 4.35 CreatureSpawner Class Reference

Inheritance diagram for CreatureSpawner:

![Inheritance Diagram for CreatureSpawner](image-url)
Collaboration diagram for CreatureSpawner:

Public Member Functions

- void CreatureKilled (CreatureTracker creature)
  
  Event where creature has been killed. Place creature into pool and spawn another.

Protected Member Functions

- virtual Vector3 findSpawnLocation ()
  
  Find a valid spawn location

- virtual void Init ()
  
  Initialize anything class specific

Protected Attributes

- int minCreatureCount
- int meanCreatureCount
- int maxCreatureCount
- CreatureManager creatureManager
Properties

- int Count [get]

  Get the count of the number of creatures alive

### 4.35.1 Detailed Description

Definition at line 5 of file CreatureSpawner.cs.

### 4.35.2 Member Function Documentation

#### 4.35.2.1 CreatureKilled()

```csharp
void CreatureSpawner.CreatureKilled (CreatureTracker creature)
```

Event where creature has been killed. Place creature into pool and spawn another.

**Parameters**

- `creature` : Creature.

Definition at line 85 of file CreatureSpawner.cs.

#### 4.35.2.2 findSpawnLocation()

```csharp
virtual Vector3 CreatureSpawner.findSpawnLocation () [protected], [virtual]
```

Find a valid spawn location

**Returns**

- The spawn location.

Reimplemented in WhaleSpawner.

Definition at line 73 of file CreatureSpawner.cs.
4.35.2.3 Init()

virtual void CreatureSpawner.Init() [protected], [virtual]

Initialize anything class specific

Reimplemented in FishSpawner.

Definition at line 106 of file CreatureSpawner.cs.

4.35.3 Member Data Documentation

4.35.3.1 creatureManager

CreatureManager CreatureSpawner.creatureManager [protected]

Definition at line 27 of file CreatureSpawner.cs.

4.35.3.2 maxCreatureCount

int CreatureSpawner.maxCreatureCount [protected]

Definition at line 16 of file CreatureSpawner.cs.

4.35.3.3 meanCreatureCount

int CreatureSpawner.meanCreatureCount [protected]

Definition at line 13 of file CreatureSpawner.cs.

4.35.3.4 minCreatureCount

int CreatureSpawner.minCreatureCount [protected]

Definition at line 11 of file CreatureSpawner.cs.

4.35.4 Property Documentation
4.35.4.1 Count

```csharp
int CreatureSpawner.Count [get]
```

Get the count of the number of creatures alive

The count.

Definition at line 40 of file CreatureSpawner.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Creatures/CreatureSpawner.cs

4.36 CreatureTracker Class Reference

Inheritance diagram for CreatureTracker:

![Inheritance Diagram](image_url)
Collaboration diagram for CreatureTracker:

```
MonoBehaviour

CreatureTracker
+ IsDead
+ Index

Public Attributes

• bool IsDead = false
  If the creature is dead or not
• int Index = 0
  The index in the creature manager's array

4.36.1 Detailed Description

Definition at line 4 of file CreatureTracker.cs.

4.36.2 Member Data Documentation

4.36.2.1 Index

int CreatureTracker.Index = 0

The index in the creature manager's array

Definition at line 14 of file CreatureTracker.cs.
4.36.2  IsDead

bool CreatureTracker.IsDead = false

If the creature is dead or not

Definition at line 9 of file CreatureTracker.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Creatures/CreatureTracker.cs

4.37  DayNight Class Reference

Inheritance diagram for DayNight:

Collaboration diagram for DayNight:
4.37.1 Detailed Description

Definition at line 4 of file DayNight.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/DayNight.cs

4.38 DayNightSkybox Class Reference

This script modifies the _Blend variable in the custom shader, Skybox/Blended based on the Game.Insance's clock.

Inheritance diagram for DayNightSkybox:

![Inheritance Diagram](image1)

Collaboration diagram for DayNightSkybox:

![Collaboration Diagram](image2)
4.38.1 Detailed Description

This script modifies the _Blend variable in the custom shader, Skybox/Blended based on the Game.Insance's clock. Definition at line 7 of file DayNightSkybox.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Effects/DayNightSkybox.cs

4.39 DeathFlavorText Class Reference

Inheritance diagram for DeathFlavorText:

![Inheritance Diagram](image)

Collaboration diagram for DeathFlavorText:

![Collaboration Diagram](image)
Public Attributes

- Text `flavorText`

4.39.1 Detailed Description

Definition at line 5 of file DeathFlavorText.cs.

4.39.2 Member Data Documentation

4.39.2.1 `flavorText`

Text DeathFlavorText.flavorText

Definition at line 9 of file DeathFlavorText.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/DeathScreen/DeathFlavorText.cs

4.40 DeathManager Class Reference

Collaboration diagram for DeathManager:

```
DeathManager
+
Death()
```

Public Member Functions

- `void Death()`
  
  Loads death screen.

4.40.1 Detailed Description

Definition at line 5 of file DeathManager.cs.
4.40.2 Member Function Documentation

4.40.2.1 Death()

```csharp
void DeathManager.Death();
```

Loads death screen.

Definition at line 12 of file DeathManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/DeathScreen/DeathManager.cs

4.41 DebrisController Class Reference

This class handles particle effects for miscellaneous debris, such as sand, leaves, dirt, and paper, in relation to them blowing around in the wind during the game. This is structured very similarly to the RainController class, except that this only handles one particle system at a time, without blending.

Inheritance diagram for DebrisController:
Collaboration diagram for DebrisController:

```
MonoBehaviour

DebrisController
+ UseCustomValues
+ Debris
+ DebrisStartThreshold
+ WindVectorXZ
+ DebrisLevel
```

Public Attributes

- `bool UseCustomValues`
- `ParticleSystem Debris`
- `float DebrisStartThreshold = 40f`
  
  *At what point should we start kicking up debris?*

Properties

- `Vector2 WindVectorXZ [get, set]`
  
  *The wind vector in XZ (considering Y is Up, but we’re only concerned with 2D wind)*
- `float DebrisLevel [get, set]`
  
  *Gets or sets the amount of debris we should be kicking up, in a value between 0 and 100.*

### 4.41.1 Detailed Description

This class handles particle effects for miscellaneous debris, such as sand, leaves, dirt, and paper, in relation to them blowing around in the wind during the game. This is structured very similarly to the `RainController` class, except that this only handles one particle system at a time, without blending.

Definition at line 9 of file DebrisController.cs.

### 4.41.2 Member Data Documentation
4.41 DebrisController Class Reference

4.41.2.1 Debris

ParticleSystem DebrisController.Debris

Definition at line 16 of file DebrisController.cs.

4.41.2.2 DebrisStartThreshold

float DebrisController.DebrisStartThreshold = 40f

At what point should we start kicking up debris?

Definition at line 36 of file DebrisController.cs.

4.41.2.3 UseCustomValues

bool DebrisController.UseCustomValues

Definition at line 13 of file DebrisController.cs.

4.41.3 Property Documentation

4.41.3.1 DebrisLevel

float DebrisController.DebrisLevel [get], [set]

Gets or sets the amount of debris we should be kicking up, in a value between 0 and 100.

Definition at line 67 of file DebrisController.cs.

4.41.3.2 WindVectorXZ

Vector2 DebrisController.WindVectorXZ [get], [set]

The wind vector in XZ (considering Y is Up, but we're only concerned with 2D wind)

Definition at line 51 of file DebrisController.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Effects/DebrisController.cs

Generated by Doxygen
### 4.42 DebugCity Class Reference

Inheritance diagram for DebugCity:

![Inheritance Diagram](image)

Collaboration diagram for DebugCity:

![Collaboration Diagram](image)

#### 4.42.1 Detailed Description

Definition at line 4 of file DebugCity.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/DebugCity.cs
4.43 Dial Class Reference

Inheritance diagram for Dial:

```
MonoBehaviour
      ↓
      Dial
         + knobDegree
```

Collaboration diagram for Dial:

```
MonoBehaviour
      ↓
      Dial
         + knobDegree
```

Public Attributes

- float knobDegree

4.43.1 Detailed Description

Definition at line 7 of file Dial.cs.
4.43.2 Member Data Documentation

4.43.2.1 knobDegree

float Dial.knobDegree

Definition at line 22 of file Dial.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Radio/Dial.cs

4.44 District Class Reference

Collaboration diagram for District:

```
+ SeedPoint
+ EdgeVerticies
+ Name
+ Blocks
+ BoundingBox
+ Configuration
```

Public Member Functions

- **District** (Vector3 seedPoint, Vector3[] verticies, DistrictConfiguration configuration)
  
  Creates a new district

- **bool ContainsPoint** (Vector2 point)
  
  Checks whether the point is within the bounds of the district.
Properties

- Vector3 **SeedPoint** [get]
  The seed point used to generate the district.
- Vector3 [] **EdgeVertices** [get]
  The list of edge vertices.
- string **Name** [get]
  Gets or sets the name.
- List< Block > **Blocks** [get]
  The list of blocks contained in this district.
- Bounds **BoundingBox** [get]
  The bounds that contain all of the district vertices.
- DistrictConfiguration **Configuration** [get]
  The configuration for constructing buildings in this district.

4.44.1 Detailed Description

Definition at line 6 of file District.cs.

4.44.2 Constructor & Destructor Documentation

4.44.2.1 District()

District.District (  
  Vector3 seedPoint,  
  Vector3 [] verticies,  
  DistrictConfiguration configuration )

Creates a new district

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seedPoint</td>
<td>The seed point that created the district.</td>
</tr>
<tr>
<td>verticies</td>
<td>The edge vertices defining the district.</td>
</tr>
<tr>
<td>configuration</td>
<td>The configuration object defining district characteristics.</td>
</tr>
</tbody>
</table>

Definition at line 16 of file District.cs.

4.44.3 Member Function Documentation

Generated by Doxygen
### ContainsPoint()

```csharp
bool District.ContainsPoint(
    Vector2 point
)
```

Checks whether the point is within the bounds of the district.

**Returns**

true, if point is within district, false otherwise.

**Parameters**

- **point** | Point to be checked.

Definition at line 93 of file District.cs.

### Property Documentation

#### Blocks

```csharp
List<Block> District.Blocks [get]
```

The list of blocks contained in this district.

Definition at line 58 of file District.cs.

#### BoundingBox

```csharp
Bounds District.BoundingBox [get]
```

The bounds that contain all of the district vertices.

Definition at line 67 of file District.cs.

#### Configuration

```csharp
DistrictConfiguration District.Configuration [get]
```

The configuration for constructing buildings in this district.

Definition at line 83 of file District.cs.
4.44.4.4 EdgeVerticies

Vector3 [] District.EdgeVerticies [get]

The list of edge vertices.

Definition at line 37 of file District.cs.

4.44.5 Name

string District.Name [get]

Gets or sets the name.

The name.

Definition at line 47 of file District.cs.

4.44.6 SeedPoint

Vector3 District.SeedPoint [get]

The seed point used to generate the district.

Definition at line 28 of file District.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/District.cs

4.45 DistrictConfiguration Class Reference

This class holds all the relevant info needed to define a district. It holds things like the building pieces the district contains, that districts materials, window types, etc.
Collaboration diagram for DistrictConfiguration:

```
DistrictConfiguration
+ WeenieBuildingTemplate
+ Doors
+ Shelters
+ MaxFloors
+ MinFloors
+ districtProceduralMaterials
+ districtProceduralWindow
Materials
+ Name
+ MinAttachmentChance
+ MaxAttachmentChance
+ SizeXDefinitions
+ DistrictAttachments
+ DistrictWindows
+ DistrictMaxWindowSpacings
+ DistrictMinWindowSpacings
+ DistrictMaterials
+ DistrictWindowMaterials
and 6 more...
```

Classes

- class BuildingTemplatePlacement
  
  Structure for accessing additional building placement configuration.

- struct ProceduralBuildingBasesSizeXDefinition
  
  Holds all of the bases and roofs for each size of building.

Public Attributes

- GameObject WeenieBuildingTemplate
  
  The Weenie building template.

- List<GameObject> Doors
  
  The doors in the district.

- List<GameObject> Shelters
  
  The shelters in the district.

- int MaxFloors
  
  The maximum number of floors for buildings in this district.

- int MinFloors
  
  The minimum number of floors for buildings in this district.

- List<Material> districtProceduralMaterials = new List<Material>()

- List<Material> districtProceduralWindowMaterials = new List<Material>()
Properties

- **string** Name  [get]
  The name of the district.
- **float** MinAttachmentChance  [get]
- **float** MaxAttachmentChance  [get]
- **ProceduralBuildingBasesSizeXDefinition [] SizeXDefinitions**  [get]
  An array of the different size buildings the district can have.
- **ProceduralBuildingAttachment [] DistrictAttachments**  [get]
  The attachments that this district takes.
- **ProceduralBuildingWindow [] DistrictWindows**  [get]
  The windows that this district can have
- **float** DistrictMaxWindowSpacings  [get]
  The max distance windows can be apart from each other in this district.
- **float** DistrictMinWindowSpacings  [get]
  The min distance windows can be apart from each other in this district.
- **Material []** DistrictMaterials  [get]
  The materials buildings can have in this district.
- **Material []** DistrictWindowMaterials  [get]
  The materials windows can have in this district.
- **Gradient** MaterialAlbedoColors  [get]
  Additional colors building's albedo can take on to increase building variability.
- **float** WindowWasherChance  [get]
  On any given building in this district it has this percent of a chance of having a window washer on it (if the building is compatible)
- **WindowWasher []** DistrictWindowWashers  [get]
  Window washers (if any) that can be spawned in this District
- **AnimationCurve** WindowWasherMaxLengthCurve  [get]
  A curve between 0 and 1 that defines the max length of window washers in the district.
- **float** WindowWasherStartUpChance  [get]
  The percent chance that a window washer in this district will start in the up position.
- **GameObject []** DistrictPosters  [get]
  Posters that appear in this district

4.45.1 Detailed Description

This class holds all the relevant info needed to define a district. It holds things like the building pieces the district contains, that districts materials, window types, etc.

Definition at line 11 of file DistrictDefinition.cs.

4.45.2 Member Data Documentation

4.45.2.1 districtProceduralMaterials

List&lt;Material&gt; DistrictConfiguration.districtProceduralMaterials = new List&lt;Material&gt;();

Definition at line 286 of file DistrictDefinition.cs.
4.45.2.2 districtProceduralWindowMaterials

List&lt;Material&gt; DistrictConfiguration.districtProceduralWindowMaterials = new List&lt;Material&gt;();

Definition at line 287 of file DistrictDefinition.cs.

4.45.2.3 Doors

List&lt;GameObject&gt; DistrictConfiguration.Doors

The doors in the district.

Definition at line 62 of file DistrictDefinition.cs.

4.45.2.4 MaxFloors

int DistrictConfiguration.MaxFloors

The maximum number of floors for buildings in this district.

Definition at line 76 of file DistrictDefinition.cs.

4.45.2.5 MinFloors

int DistrictConfiguration.MinFloors

The minimum number of floors for buildings in this district.

Definition at line 82 of file DistrictDefinition.cs.

4.45.2.6 Shelters

List&lt;GameObject&gt; DistrictConfiguration.Shelters

The shelters in the district.

Definition at line 68 of file DistrictDefinition.cs.
4.45.2  WeenieBuildingTemplate

GameObject DistrictConfiguration.WeenieBuildingTemplate

The Weenie building template.
Definition at line 56 of file DistrictDefinition.cs.

4.45.3  Property Documentation

4.45.3.1  DistrictAttachments

ProceduralBuildingAttachment [ ] DistrictConfiguration.DistrictAttachments [get]

The attachments that this district takes.
Definition at line 200 of file DistrictDefinition.cs.

4.45.3.2  DistrictMaterials

Material [ ] DistrictConfiguration.DistrictMaterials [get]

The materials buildings can have in this district.
Definition at line 252 of file DistrictDefinition.cs.

4.45.3.3  DistrictMaxWindowSpacings

float DistrictConfiguration.DistrictMaxWindowSpacings [get]

The max distance windows can be apart from eachother in this district.
Definition at line 226 of file DistrictDefinition.cs.

4.45.3.4  DistrictMinWindowSpacings

float DistrictConfiguration.DistrictMinWindowSpacings [get]

The min distance windows can be apart from eachother in this district.
Definition at line 239 of file DistrictDefinition.cs.
4.45.3.5 DistrictPosters

GameObject [] DistrictConfiguration.DistrictPosters [get]

Posters that appear in this district

Definition at line 349 of file DistrictDefinition.cs.

4.45.3.6 DistrictWindowMaterials

Material [] DistrictConfiguration.DistrictWindowMaterials [get]

The materials windows can have in this district.

Definition at line 265 of file DistrictDefinition.cs.

4.45.3.7 DistrictWindows

ProceduralBuildingWindow [] DistrictConfiguration.DistrictWindows [get]

The windows that this district can have

Definition at line 213 of file DistrictDefinition.cs.

4.45.3.8 DistrictWindowWashers

WindowWasher [] DistrictConfiguration.DistrictWindowWashers [get]

Window washers (if any) that can be spawned in this District

Definition at line 308 of file DistrictDefinition.cs.

4.45.3.9 MaterialAlbedoColors

Gradient DistrictConfiguration.MaterialAlbedoColors [get]

Additional colors building's albedo can take on to increase building variability.

Definition at line 279 of file DistrictDefinition.cs.
4.45.3.10 MaxAttachmentChance

float DistrictConfiguration.MaxAttachmentChance [get]

Definition at line 99 of file DistrictDefinition.cs.

4.45.3.11 MinAttachmentChance

float DistrictConfiguration.MinAttachmentChance [get]

Definition at line 88 of file DistrictDefinition.cs.

4.45.3.12 Name

string DistrictConfiguration.Name [get]

The name of the district.

Definition at line 42 of file DistrictDefinition.cs.

4.45.3.13 SizeXDefinitions

ProceduralBuildingBasesSizeXDefinition[ ] DistrictConfiguration.SizeXDefinitions [get]

An array of the different size buildings the district can have.

Definition at line 187 of file DistrictDefinition.cs.

4.45.3.14 WindowWasherChance

float DistrictConfiguration.WindowWasherChance [get]

On any given building in this district it has this percent of a chance of having a window washer on it (if the building is compatible)

Definition at line 295 of file DistrictDefinition.cs.
### 4.45.3.15 WindowWasherMaxLengthCurve

`AnimationCurve DistrictConfiguration.WindowWasherMaxLengthCurve [get]`

A curve between 0 and 1 that defines the max length of window washers in the district.

Definition at line 322 of file `DistrictDefinition.cs`.

### 4.45.3.16 WindowWasherStartUpChance

`float DistrictConfiguration.WindowWasherStartUpChance [get]`

The percent chance that a window washer in this district will start in the up position.

Definition at line 336 of file `DistrictDefinition.cs`.

The documentation for this class was generated from the following file:

- `Assets/Scripts/City/DistrictDefinition.cs`

### 4.46 DistrictGenerator Class Reference

Inheritance diagram for `DistrictGenerator`:

```
MonoBehaviour

DistrictGenerator

+ Generate()
```
Public Member Functions

- **District [] Generate (int seed, Bounds cityBounds)**
  
  Generate the districts.

4.46.1 Detailed Description

Definition at line 7 of file DistrictGenerator.cs.

4.46.2 Member Function Documentation

4.46.2.1 Generate()

```csharp
District [] DistrictGenerator.Generate ( int seed,
                                        Bounds cityBounds )
```

Generate the districts.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>seed</td>
<td>The city generation seed.</td>
</tr>
<tr>
<td>cityBounds</td>
<td>The bounds defining the city's size.</td>
</tr>
</tbody>
</table>
Returns

Definition at line 24 of file DistrictGenerator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/DistrictGenerator.cs

### 4.47 DistrictItemConfiguration Class Reference

Collaboration diagram for DistrictItemConfiguration:

![Collaboration Diagram](image)

#### Public Member Functions

- **DistrictItemConfiguration ()**
  
  *Initializes a new instance of the DistrictItemInfo class.*

#### Properties

- **List<string> ItemNames [get, set]**
  
  *Gets or sets the item templates that will be duplicated to create objects.*

- **List<float> ItemExtents [get, set]**
  
  *Gets or sets the item extents.*

- **List<GameObject> ItemTemplates [get, set]**
  
  *Gets or sets the items that can be generated.*

- **List<float> DoorExtents [get, set]**
  
  *Gets or sets the door extents.*

- **List<GameObject> DoorTemplates [get, set]**
  
  *Gets or sets the door templates.*

- **List<float> ShelterExtents [get, set]**
  
  *Gets or sets the shelter extents.*

- **List<GameObject> ShelterTemplates [get, set]**
  
  *Gets or sets the shelter templates.*
4.47 DistrictItemConfiguration Class Reference

4.47.1 Detailed Description

Definition at line 5 of file DistrictItemConfiguration.cs.

4.47.2 Constructor & Destructor Documentation

4.47.2.1 DistrictItemConfiguration()

DistrictItemConfiguration.DistrictItemConfiguration ( )

Initializes a new instance of the DistrictItemInfo class.

Definition at line 10 of file DistrictItemConfiguration.cs.

4.47.3 Property Documentation

4.47.3.1 DoorExtents

List<float> DistrictItemConfiguration.DoorExtents [get], [set]

Gets or sets the door extents.

The door extents.

Definition at line 51 of file DistrictItemConfiguration.cs.

4.47.3.2 DoorTemplates

List<GameObject> DistrictItemConfiguration.DoorTemplates [get], [set]

Gets or sets the door templates.

The door templates.

Definition at line 61 of file DistrictItemConfiguration.cs.
4.47.3.3  ItemExtents

List<float> DistrictItemConfiguration.ItemExtents [get], [set]

Gets or sets the item extents.

The item extents.

Definition at line 31 of file DistrictItemConfiguration.cs.

4.47.3.4  ItemNames

List<string> DistrictItemConfiguration.ItemNames [get], [set]

Gets or sets the item templates that will be duplicated to create objects.

The item templates.

Definition at line 21 of file DistrictItemConfiguration.cs.

4.47.3.5  ItemTemplates

List<GameObject> DistrictItemConfiguration.ItemTemplates [get], [set]

Gets or sets the Items that can be generated.

The item templates.

Definition at line 41 of file DistrictItemConfiguration.cs.

4.47.3.6  ShelterExtents

List<float> DistrictItemConfiguration.ShelterExtents [get], [set]

Gets or sets the shelter extents.

The shelter extents.

Definition at line 71 of file DistrictItemConfiguration.cs.
4.47.3 ShelterTemplates

List<GameObject> DistrictItemConfiguration.ShelterTemplates [get], [set]

Gets or sets the shelter templates.

The shelter templates.

Definition at line 81 of file DistrictItemConfiguration.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/DistrictItemConfiguration.cs

4.48 DistrictItemRarityConfiguration Class Reference

Collaboration diagram for DistrictItemRarityConfiguration:

```
<table>
<thead>
<tr>
<th>DistrictItemRarityConfiguration</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ DistrictItemRarityConfiguration()</td>
</tr>
<tr>
<td>+ GetWeightedRandomItemIndex()</td>
</tr>
<tr>
<td>+ SetUpVoseAlias()</td>
</tr>
</tbody>
</table>
```

Public Member Functions

- DistrictItemRarityConfiguration ()
- int GetWeightedRandomItemIndex ()
  
  Gets a random index corresponding to an item that is weighted according to the rarity of the item.

- void SetUpVoseAlias (List<float> rarityWeights)
  
  Sets up the information needed to use vose alias for weighted random generation. Random generation using this method involves generating random numbers at unequal rates depending on the weights assigned to the number. In this case, the index number associated with an item, and the weights are dependent on the rarity level of the item. To achieve this in constant time, weighted pairs are created and put into a list. Then a random weighted pair is picked from that list, and one of the parts of the weighted pair is returned. For a more thorough explanation of this method please go to this link: http://www.keithschwarz.com/darts-dice-coins/

4.48.1 Detailed Description

Definition at line 4 of file DistrictItemRarityConfiguration.cs.
4.48.2 Constructor & Destructor Documentation

4.48.2.1 DistrictItemRarityConfiguration()

DistrictItemRarityConfiguration.DistrictItemRarityConfiguration ( )

Definition at line 9 of file DistrictItemRarityConfiguration.cs.

4.48.3 Member Function Documentation

4.48.3.1 GetWeightedRandomItemIndex()

int DistrictItemRarityConfiguration.GetWeightedRandomItemIndex ( )

Gets a random index corresponding to an item that is weighted according to the rarity of the item.

Returns

A weighted random index.

Definition at line 18 of file DistrictItemRarityConfiguration.cs.

4.48.3.2 SetUpVoseAlias()

void DistrictItemRarityConfiguration.SetUpVoseAlias ( List< float > rarityWeights )

Sets up the information needed to use vose alias for weighted random generation. Random generation using this method involves generating random numbers at unequal rates depending on the weights assigned to the number. In this case, the index number associated with an item, and the weights are dependent on the rarity level of the item. To achieve this in constant time, weighted pairs are created and put into a list. Then a random weighted pair is picked from that list, and one of the parts of the weighted pair is returned. For a more thorough explanation of this method please go to this link: http://www.keithschwarz.com/darts-dice-coins/

Parameters

rarityWeights The values assigned to rarities. The smaller the rarer.

Definition at line 44 of file DistrictItemRarityConfiguration.cs.

The documentation for this class was generated from the following file:
• Assets/Scripts/ItemProceduralGeneration/DistrictItemRarityConfiguration.cs

4.49 DistrictPopup Class Reference

Inheritance diagram for DistrictPopup:

```
PropertyDrawer

DistrictPopup
+ OnGUI()
```

Collaboration diagram for DistrictPopup:

```
PropertyDrawer

DistrictPopup
+ OnGUI()
```

Public Member Functions

• override void OnGUI (Rect position, SerializedProperty property, GUIContent label)

Overrides the OnGUI method and renders a dropdown to pick an available baseItem.
4.49.1 Detailed Description

Definition at line 8 of file DistrictPopup.cs.

4.49.2 Member Function Documentation

4.49.2.1 OnGUI()

override void DistrictPopup.OnGUI ( Rect position, SerializedProperty property, GUIContent label )

Overrides the OnGUI method and renders a dropdown to pick an available baseItem.

Parameters

<table>
<thead>
<tr>
<th>position</th>
<th>Position of dropdown.</th>
</tr>
</thead>
<tbody>
<tr>
<td>property</td>
<td>Property to set.</td>
</tr>
<tr>
<td>label</td>
<td>Property label.</td>
</tr>
</tbody>
</table>

Definition at line 19 of file DistrictPopup.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Editor/DistrictPopup.cs
Inheritance diagram for DistrictPopupAttribute:

```
PropertyAttribute

DistrictPopupAttribute
```

Collaboration diagram for DistrictPopupAttribute:

```
PropertyAttribute

DistrictPopupAttribute
```

4.50.1 Detailed Description

Definition at line 3 of file DistrictPopupAttribute.cs.

The documentation for this class was generated from the following file:

```
• Assets/Scripts/City/DistrictPopupAttribute.cs
```
4.51 DiurnalTemperatureVariance Class Reference

A gamified version of: https://en.wikipedia.org/wiki/Diurnal_temperature_variation

Collaboration diagram for DiurnalTemperatureVariance:

```
+ TemperatureVariance
|     |
|     |
```

Properties

- static float TemperatureVariance [get]
  
  Gets a float representing the temperature change at that particular time of day. This is Diurnal some it will go up and down following the same pattern as the tide.

4.51.1 Detailed Description

A gamified version of: https://en.wikipedia.org/wiki/Diurnal_temperature_variation

Definition at line 5 of file DiurnalTemperatureVariance.cs.

4.51.2 Property Documentation

4.51.2.1 TemperatureVariance

float DiurnalTemperatureVariance.TemperatureVariance [static], [get]

Gets a float representing the temperature change at that particular time of day. This is Diurnal some it will go up and down following the same pattern as the tide.

The temperature variance.

Definition at line 17 of file DiurnalTemperatureVariance.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/DiurnalTemperatureVariance.cs
4.52 DummyGenerator Class Reference

Inheritance diagram for DummyGenerator:

```
MonoBehaviour

DummyGenerator
+ buildings
+ doors
+ shelters
```

Collaboration diagram for DummyGenerator:

```
MonoBehaviour

DummyGenerator
+ buildings
+ doors
+ shelters
```

Public Attributes

- `List<Renderer> buildings`
  
  *This is a dummy generator to be deleted before merging. Used for testing in the ItemSpawner scene!*
- `List<GameObject> doors`
- `List<GameObject> shelters`
4.52.1 Detailed Description

Definition at line 5 of file DummyGenerator.cs.

4.52.2 Member Data Documentation

4.52.2.1 buildings

List<Renderer> DummyGenerator.buildings

This is a dummy generator to be deleted before merging. Used for testing in the ItemSpawner scene!

Definition at line 12 of file DummyGenerator.cs.

4.52.2.2 doors

List<GameObject> DummyGenerator.doors

Definition at line 14 of file DummyGenerator.cs.

4.52.2.3 shelters

List<GameObject> DummyGenerator.shelters

Definition at line 16 of file DummyGenerator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/DummyGenerator.cs
4.53 DummyNoteGenerator Class Reference

Inheritance diagram for DummyNoteGenerator:

```
MonoBehaviour

DummyNoteGenerator
```

Collaboration diagram for DummyNoteGenerator:

```
MonoBehaviour

DummyNoteGenerator
```

4.53.1 Detailed Description

Definition at line 9 of file DummyNoteGenerator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Notes/DummyNoteGenerator.cs
4.54 EndCreditsBehavior Class Reference

Inheritance diagram for EndCreditsBehavior:

Collaboration diagram for EndCreditsBehavior:

4.54.1 Detailed Description

Definition at line 5 of file EndCreditsBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/MainMenuScene/EndCreditsBehavior.cs
Public Member Functions

- void EndingTrigger (Collider other, bool deathTrigger)
  
  Called to trigger the ending

- void BeaconTurnedOn ()
  
  Called by the beacon script to signify that the ending has begun, and the player needs to wait for rescue.
Public Attributes

- GameObject [] hideThese
- MonoBehaviour [] disableThese

4.55.1 Detailed Description

Definition at line 5 of file EndingController.cs.

4.55.2 Member Function Documentation

4.55.2.1 BeaconTurnedOn()

```cpp
void EndingController.BeaconTurnedOn ()
```

Called by the beacon script to signify that the ending has begun, and the player needs to wait for rescue.

Definition at line 111 of file EndingController.cs.

4.55.2.2 EndingTrigger()

```cpp
void EndingController.EndingTrigger ( Collider other, bool deathTrigger )
```

Called to trigger the ending

Parameters

| other | The collider that entered one of the ending triggers, should be the player |

///

Parameters

| deathtrigger | Did the player enter the death trigger? |

Definition at line 79 of file EndingController.cs.

4.55.3 Member Data Documentation
4.55.3.1 disableThese

MonoBehaviour [] EndingController.disableThese

Definition at line 21 of file EndingController.cs.

4.55.3.2 hideThese

GameObject [] EndingController.hideThese

Definition at line 17 of file EndingController.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Ending/EndingController.cs

4.56 EndingTrigger Class Reference

Inheritance diagram for EndingTrigger:
Collaboration diagram for EndingTrigger:

![Collaboration Diagram](image)

### 4.56.1 Detailed Description

Definition at line 4 of file EndingTrigger.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Ending/EndingTrigger.cs
4.57 EquipableCategory Class Reference

Inheritance diagram for EquipableCategory:
Collaboration diagram for EquipableCategory:

```
Public Member Functions

• override ItemCategory GetDuplicate() 
  Gets a copy of the ItemCategory.

• override void ReadyCategory() 
  Preps the category for use by loading attributes and actions into lists.

• void Equip() 
```

Generated by Doxygen
Equip this item.

- void **UnEquip** ()
  Unequip this item.

**Protected Attributes**

- const string **equipActionName** = "Equip"
- const string **unequipActionName** = "Unequip"
- const string **equipedAttributeName** = "equiped"

**Properties**

- float **Equiped** [get, set]
  Gets or sets the equiped state. 1 is equiped, 0 is unequiped. TODO: When there are multiple equipables, when a new item is equiped, the pre-equiped item must have this set to 0f

**Additional Inherited Members**

4.57.1 Detailed Description

Definition at line 5 of file EquipableCategory.cs.

4.57.2 Member Function Documentation

4.57.2.1 Equip()

void EquipableCategory.Equip ()

Equip this item.

Definition at line 68 of file EquipableCategory.cs.

4.57.2.2 GetDuplicate()

override ItemCategory EquipableCategory.GetDuplicate () [virtual]

Gets a copy of the ItemCategory.

Returns
The duplicate.

Reimplemented from ItemCategory.

Reimplemented in LightCategory, WarmthIdolCategory, and FishingRodCategory.

Definition at line 26 of file EquipableCategory.cs.
4.57.2.3 ReadyCategory()

```csharp
override void EquipableCategory.ReadyCategory() [virtual]
```
Preps the category for use by loading attributes and actions into lists.
Reimplemented from `ItemCategory`.
Reimplemented in `LightCategory`, `FishingRodCategory`, `WarmthIdolCategory`, and `IdolCategory`.
Definition at line 47 of file `EquipableCategory.cs`.

4.57.2.4 UnEquip()

```csharp
void EquipableCategory.UnEquip()
```
Unequip this item.
Definition at line 79 of file `EquipableCategory.cs`.

4.57.3 Member Data Documentation

4.57.3.1 equipActionName

```csharp
const string EquipableCategory.equipActionName = "Equip" [protected]
```
Definition at line 18 of file `EquipableCategory.cs`.

4.57.3.2 equipedAttributeName

```csharp
const string EquipableCategory.equipedAttributeName = "equiped" [protected]
```
Definition at line 20 of file `EquipableCategory.cs`.

4.57.3.3 unequipActionName

```csharp
const string EquipableCategory.unequipActionName = "Unequip" [protected]
```
Definition at line 19 of file `EquipableCategory.cs`.

Generated by Doxygen
4.57.4 Property Documentation

4.57.4.1 Equiped

float EquipableCategory.Equiped [get], [set]

Gets or sets the equipped state. 1 is equipped, 0 is unequipped. TODO: When there are multiple equipables, when
a new item is equipped, the pre-equipped item must have this set to 0f

The equipped.

Definition at line 13 of file EquipableCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/EquipableCategory.cs

4.58 EquippedItemDropdown Class Reference

Inheritance diagram for EquippedItemDropdown:

```
MonoBehaviour

EquippedItemDropdown
  + contentPanel
  + equippedItemHolder
    + ToggleDisplay()
    + DisplayOptions()
    + HideOptions()
    + SetEquipped()
    + GetSelected()
    + Unequipped()
```
Collaboration diagram for EquippedItemDropdown:

Public Member Functions

- void **ToggleDisplay** ()
  
  Toggles the display.

- void **DisplayOptions** ()
  
  Displays the options.

- void **HideOptions** ()
  
  Hides the options.

- void **SetEquipped** (string image, List<ItemAction> actions)
  
  Sets the information for equipped item.

- string **GetSelected** ()
  
  Gets the selected item option.

- void **Unequipped** ()
  
  Removes information for unequipped

Public Attributes

- GameObject **contentPanel**
- Image **equippedItemHolder**

4.58.1 Detailed Description

Definition at line 6 of file EquippedItemDropdown.cs.
4.58.2 Member Function Documentation

4.58.2.1 DisplayOptions()

```csharp
void EquippedItemDropdown.DisplayOptions();
```

Displays the options.

Definition at line 55 of file EquippedItemDropdown.cs.

4.58.2.2 GetSelected()

```csharp
string EquippedItemDropdown.GetSelected();
```

Gets the selected item option.

Returns

The selected.

Definition at line 95 of file EquippedItemDropdown.cs.

4.58.2.3 HideOptions()

```csharp
void EquippedItemDropdown.HideOptions();
```

Hides the options.

Definition at line 67 of file EquippedItemDropdown.cs.

4.58.2.4 SetEquipped()

```csharp
void EquippedItemDropdown.SetEquipped(
    string image,
    List<ItemAction> actions)
```

Sets the information for equipped item.

Parameters

<table>
<thead>
<tr>
<th>image</th>
<th>Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>actions</td>
<td>Actions</td>
</tr>
</tbody>
</table>

Generated by Doxygen
4.58.2.5  **ToggleDisplay()**

```csharp
void EquippedItemDropdown.ToggleDisplay()
```

Toggles the display.

Definition at line 40 of file EquippedItemDropdown.cs.

4.58.2.6  **Unequipped()**

```csharp
void EquippedItemDropdown.Unequipped()
```

Removes information for unequipped

Definition at line 103 of file EquippedItemDropdown.cs.

4.58.3  **Member Data Documentation**

4.58.3.1  **contentPanel**

```csharp
GameObject EquippedItemDropdown.contentPanel
```

Definition at line 14 of file EquippedItemDropdown.cs.

4.58.3.2  **equippedItemHolder**

```csharp
Image EquippedItemDropdown.equippedItemHolder
```

Definition at line 18 of file EquippedItemDropdown.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/WorldInteraction/EquippedItemDropdown.cs
4.59 EventManager Class Reference

Collaboration diagram for EventManager:

```
EventManager
+ StormStartedDelegate()
+ StormStoppedDelegate()
+ PlayerBoardRaftDelegate()
+ WeatherUpdatedDelegate()
+ RadioMusicDelegate()
+ StormStart()
+ StormStop()
+ RaftBoarded()
+ WeatherUpdated()
+ RadioMusicTurnedOn()
+ RadioMusicTurnedOff()
```

Public Member Functions

- delegate void StormStartedDelegate ()
- delegate void StormStoppedDelegate ()
- delegate void PlayerBoardRaftDelegate ()
- delegate void WeatherUpdatedDelegate (float precipitation)
- delegate void RadioMusicDelegate (bool turnedOn)
- void StormStart ()
  
  Storm start function called by WeatherSystem. Notifies StormStartedSubscription subscribers.
- void StormStop ()
  
  Storm stop function called by WeatherSystem. Notifies StormStoppedSubscription subscribers.
- void RaftBoarded ()
  
  Raft boarded function called by PlayerController. Notifies PlayerBoardRaftSubscription subscribers.
- void WeatherUpdated (float precipitation)
  
  Weather updated function called by WeatherSystem. Notifies WeatherUpdatedSubscription subscribers.
- void RadioMusicTurnedOn ()
  
- void RadioMusicTurnedOff ()
  

Events

- StormStartedDelegate StormStartedSubscription
  
  Storm started event -- can be subscribed to by calling EventManager.Instance.StormStartedSubscription += functionToCall
- StormStoppedDelegate StormStoppedSubscription
  
  Storm stopped event
• **PlayerBoardRaftDelegate** PlayerBoardRaftSubscription
  
  *Raft boarded event*

• **WeatherUpdatedDelegate** WeatherUpdatedSubscription

  *Weather updated event*

• **RadioMusicDelegate** RadioMusicSubscription

  *Radio music channel event*

### 4.59.1 Detailed Description

Definition at line 5 of file EventManager.cs.

### 4.59.2 Member Function Documentation

#### 4.59.2.1 PlayerBoardRaftDelegate()

delegate void EventManager.PlayerBoardRaftDelegate ( )

#### 4.59.2.2 RadioMusicDelegate()

delegate void EventManager.RadioMusicDelegate ( 
  bool turnedOn )

#### 4.59.2.3 RadioMusicTurnedOff()

void EventManager.RadioMusicTurnedOff ( )


Definition at line 96 of file EventManager.cs.

#### 4.59.2.4 RadioMusicTurnedOn()

void EventManager.RadioMusicTurnedOn ( )


Definition at line 85 of file EventManager.cs.
4.59.2.5 RaftBoarded()

```csharp
void EventManager.RaftBoarded();
```

Raft boarded function called by PlayerController. Notifies PlayerBoardRaftSubscription subscribers.

Definition at line 62 of file EventManager.cs.

4.59.2.6 StormStart()

```csharp
void EventManager.StormStart();
```

Storm start function called by WeatherSystem. Notifies StormStartedSubscription subscribers.

Definition at line 40 of file EventManager.cs.

4.59.2.7 StormStartedDelegate()

```csharp
delegate void EventManager.StormStartedDelegate();
```

4.59.2.8 StormStop()

```csharp
void EventManager.StormStop();
```

Storm stop function called by WeatherSystem. Notifies StormStoppedSubscription subscribers.

Definition at line 51 of file EventManager.cs.

4.59.2.9 StormStoppedDelegate()

```csharp
delegate void EventManager.StormStoppedDelegate();
```

4.59.2.10 WeatherUpdated()

```csharp
void EventManager.WeatherUpdated(float precipitation);
```

Weather updated function called by WeatherSystem. Notifies WeatherUpdatedSubscription subscribers.

Definition at line 73 of file EventManager.cs.
4.59.2.11 WeatherUpdatedDelegate()

delegate void EventManager.WeatherUpdatedDelegate(float precipitation)

4.59.3 Event Documentation

4.59.3.1 PlayerBoardRaftSubscription

PlayerBoardRaftDelegate EventManager.PlayerBoardRaftSubscription
Raft boarded event
Definition at line 22 of file EventManager.cs.

4.59.3.2 RadioMusicSubscription

RadioMusicDelegate EventManager.RadioMusicSubscription
Radio music channel event
Definition at line 34 of file EventManager.cs.

4.59.3.3 StormStartedSubscription

StormStartedDelegate EventManager.StormStartedSubscription
Storm started event – can be subscribed to by calling EventManager.Instance.StormStartedSubscription += functionToCall
Definition at line 10 of file EventManager.cs.

4.59.3.4 StormStoppedSubscription

StormStoppedDelegate EventManager.StormStoppedSubscription
Storm stopped event
Definition at line 16 of file EventManager.cs.
4.59.3.5 WeatherUpdatedSubscription

WeatherUpdatedDelegate EventManager.WeatherUpdatedSubscription

Weather updated event

Definition at line 28 of file EventManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Events/EventManager.cs

4.60 FaderManager Class Reference

Can be called to fade the screen in or out.

Inheritance diagram for FaderManager:
Collaboration diagram for FaderManager:

![Collaboration Diagram]

**Public Member Functions**

- void **EndGameFade** (float fadeTime)
  
  *Fade in the ending stuff*

- void **ShowMainMenuButton** ()
  
  *Enable the main menu button*

- void **GoToMain** ()
  
  *Take me to the main menu*

**4.60.1 Detailed Description**

Can be called to fade the screen in or out.

TODO: Let it do what I said above. Right now it's just for fading the game out at the end!

Definition at line 12 of file FaderManager.cs.

**4.60.2 Member Function Documentation**

**4.60.2.1 EndGameFade()**

```csharp
void FaderManager.EndGameFade ( 
    float fadeTime )
```

*Fade in the ending stuff*
Parameters

| fadeTime | Time the fade takes |

Definition at line 32 of file FaderManager.cs.

4.60.2.2 GoToMain()

```csharp
void FaderManager.GoToMain();
```

Take me to the main menu

Definition at line 50 of file FaderManager.cs.

4.60.2.3 ShowMainMenuButton()

```csharp
void FaderManager.ShowMainMenuButton();
```

Enable the main menu button

Definition at line 42 of file FaderManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/Utility/FaderManager.cs

4.61 FileManager Class Reference

Collaboration diagram for FileManager:

```
+ FileManager

+ BuildPath()
+ SaveFile()
+ GetDocument()
+ SaveAllDocuments()
```
Static Public Member Functions

- static string BuildPath (string fileName)
  Builds the path.
- static void SaveFile (string fileName, string content)
  Saves content to the file
- static string GetDocument (string fileName)
  Gets the document using google drive, if possible when in editor mode.
- static void SaveAllDocuments ()
  Saves all documents from google drive.

4.61.1 Detailed Description

Definition at line 4 of file FileManager.cs.

4.61.2 Member Function Documentation

4.61.2.1 BuildPath()

static string FileManager.BuildPath ( string fileName ) [static]

Builds the path.

Returns

  The path.

Parameters

| fileName | File name |

Definition at line 13 of file FileManager.cs.

4.61.2.2 GetDocument()

static string FileManager.GetDocument ( string fileName ) [static]

Gets the document using google drive, if possible when in editor mode.

Returns

  The document.
Parameters

| fileName | File name. |

Definition at line 61 of file FileManager.cs.

4.61.2.3  SaveAllDocuments()

static void FileManager.SaveAllDocuments ( ) [static]

Saves all documents from google drive.

Definition at line 71 of file FileManager.cs.

4.61.2.4  SaveFile()

static void FileManager.SaveFile ( string fileName, string content ) [static]

Saves content to the file

Returns

The file.

Parameters

| fileName | File name. |
| content  | Content.  |

Definition at line 29 of file FileManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/File/FileManager.cs
Inheritance diagram for FireBaseCategory:
Collaboration diagram for FirebaseCategory:

Public Member Functions

- `override ItemCategory GetDuplicate()`
  Creates a copy of this fuel category.
- `override void ReadyCategory()`
  Readies the item category by adding the attributes and actions it can complete.
- `float CalculateRemainingFuel()`
Calculates remaining fuel in the fire. This is not an action that can be done in the inventory! This is only used in the world.

- **void AddFuel** (float burnTime)
  Adds fuel to the fire. Not an action that can be done in the inventory! This is only used in the world.

- **void SetDown ()**
  Sets down the fire pit in the world. Drops it where the player stands.

### Properties

- **float BurnRateMultiplier** [get, set]
  Gets and sets the life the fire gains.

- **float StartingFuel** [get, set]
  Gets or sets the starting fuel.

- **float FuelRemaining** [get, set]
  Gets or sets the fuel remaining.

### Additional Inherited Members

#### 4.62.1 Detailed Description

Definition at line 5 of file FirebaseCategory.cs.

#### 4.62.2 Member Function Documentation

##### 4.62.2.1 AddFuel()

```csharp
void FirebaseCategory.AddFuel (float burnTime )
```

Adds fuel to the fire. Not an action that can be done in the inventory! This is only used in the world.

**Parameters**

| burnTime | Burn time. |

Definition at line 99 of file FirebaseCategory.cs.

##### 4.62.2.2 CalculateRemainingFuel()

```csharp
float FirebaseCategory.CalculateRemainingFuel ( )
```

Calculates remaining fuel in the fire. This is not an action that can be done in the inventory! This is only used in the world.
Returns

The fire.

Definition at line 88 of file FireBaseCategory.cs.

4.62.2.3 GetDuplicate()

override ItemCategory FireBaseCategory.GetDuplicate () [virtual]

Creates a copy of this fuel category.

Returns

The duplicate.

Reimplemented from ItemCategory.

Definition at line 48 of file FireBaseCategory.cs.

4.62.2.4 ReadyCategory()

override void FireBaseCategory.ReadyCategory () [virtual]

Readies the item category by adding the attributes and actions it can complete.

Reimplemented from ItemCategory.

Definition at line 71 of file FireBaseCategory.cs.

4.62.2.5 SetDown()

void FireBaseCategory.SetDown ()

Sets down the fire pit in the world. Drops it where the player stands.

Definition at line 107 of file FireBaseCategory.cs.

4.62.3 Property Documentation
4.62.3.1 BurnRateMultiplier

float FirebaseCategory.BurnRateMultiplier [get], [set]

Gets and sets the life the fire gains.

Definition at line 11 of file FirebaseCategory.cs.

4.62.3.2 FuelRemaining

float FirebaseCategory.FuelRemaining [get], [set]

Gets or sets the fuel remaining.

The fuel remaining.

Definition at line 31 of file FirebaseCategory.cs.

4.62.3.3 StartingFuel

float FirebaseCategory.StartingFuel [get], [set]

Gets or sets the starting fuel.

The starting fuel.

Definition at line 21 of file FirebaseCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/FirebaseCategory.cs
Inheritance diagram for FireInteractable:
Public Member Functions

- **override void SetUp ()**
  
  *Sets openInventory as an action that should fire off when PerformAction is called.*

- **void UseFirePit ()**
  
  *Interact with the fire pit.*

- **IEnumerator UpdateFire ()**
  
  *Updates the fire.*

- **void BurnItem ()**
  
  *Add fuel to the fire.*

- **void IgniteFire ()**
  
  *Ignites the fire.*

- **void OnTriggerEnter (Collider other)**
  
  *When next to fire set IsByFire bool to true.*
Properties

- **FireBaseCategory FireBase** [get, set]
  
  Gets or sets the fire base category object.

Additional Inherited Members

4.63.1 Detailed Description

Definition at line 7 of file FireInteractable.cs.

4.63.2 Member Function Documentation

4.63.2.1 BurnItem()

```cpp
void FireInteractable.BurnItem ( )
```

Add fuel to the fire.

Parameters

- **fuelName** Fuel name.

Definition at line 121 of file FireInteractable.cs.

4.63.2.2 IgniteFire()

```cpp
void FireInteractable.IgniteFire ( )
```

Ignites the fire.

Definition at line 158 of file FireInteractable.cs.

4.63.2.3 OnTriggerEnter()

```cpp
void FireInteractable.OnTriggerEnter ( Collider other )
```

When next to fire set IsByFire bool to true.

Definition at line 174 of file FireInteractable.cs.
4.63.2.4 SetUp()

```csharp
override void FireInteractable.SetUp() [virtual]
```

Sets openInventory as an action that should fire off when PerformAction is called.

Reimplemented from `InteractableObject`.

Definition at line 48 of file `FireInteractable.cs`.

4.63.2.5 UpdateFire()

```csharp
IEnumerator FireInteractable.UpdateFire() {

Updates the fire.

Returns

The fire.

Definition at line 93 of file `FireInteractable.cs`.

4.63.2.6 UseFirePit()

```csharp
void FireInteractable.UseFirePit()
```

Interact with the fire pit.

Definition at line 77 of file `FireInteractable.cs`.

4.63.3 Property Documentation

4.63.3.1 FireBase

```csharp
FireBaseCategory FireInteractable.FireBase [get], [set]
```

Gets or sets the fire base category object.

The fire base.

Definition at line 14 of file `FireInteractable.cs`.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/FireInteractable.cs
Inheritance diagram for FishAgent:
Collaboration diagram for FishAgent:

```
MonoBehaviour

FishAgentConfig
+ CohesionRadius
+ SeparationRadius
+ AlignmentRadius
+ WanderRadius
+ AvoidRadius
+ CohesionWeight
+ SeparationWeight
+ AlignmentWeight
+ WanderWeight
+ AvoidWeight
and 6 more...
+ MaxRadius
+ MaxWeight
+ RandomizeSelf()

#config

FishAgent
+ Velocity
  # predatorLayer
  # agentLayer
+ Cohesion()
+ Separation()
+ Alignment()
+ Wander()
+ Combine()
+ InFieldOfView()
# setLayers()
```

**Public Member Functions**

- Vector2 **Cohesion ()**
  
  *Go to center of neighbors*

- Vector2 **Separation ()**
  
  *Move away from neighbors*

- Vector2 **Alignment ()**
  
  *Rotate in direction of movement*
4.64 FishAgent Class Reference

- Vector2 **Wander** ()
  
  *Smooth out movement*

- virtual Vector2 **Combine** ()
  
  *Use alignment, Cohesion, and Separation to define behavior with different proportions based on importance*

- bool **InFieldOfView** (Vector3 agent)
  
  *Check if agent is in field of view for this agent*

### Protected Member Functions

- virtual void **setLayers** ()

### Protected Attributes

- LayerMask **predatorLayer**
- LayerMask **agentLayer**
- FishAgentConfig **config**

### Properties

- Vector3 **Velocity** [get]
  
  *The velocity of the agent*

#### 4.64.1 Detailed Description

Definition at line 8 of file FishAgent.cs.

#### 4.64.2 Member Function Documentation

##### 4.64.2.1 Alignment()

**Vector2 FishAgent.Alignment ()**

*Rotate in direction of movement*

*Returns*

Definition at line 187 of file FishAgent.cs.
4.64.2.2 Cohesion()

Vector2 FishAgent.Cohesion()

Go to center of neighbors

Returns
    Center Point

Definition at line 115 of file FishAgent.cs.

4.64.2.3 Combine()

virtual Vector2 FishAgent.Combine() [virtual]

Use alignment, Cohesion, and Separation to define behavior with different proportions based on importance

Returns
    Vector with correct behavior

Definition at line 231 of file FishAgent.cs.

4.64.2.4 InFieldOfView()

bool FishAgent.InFieldOfView(
    Vector3 agent)

Check if agent is in field of view for this agent

Parameters

agent

Returns

Definition at line 244 of file FishAgent.cs.
4.64 FishAgent Class Reference

4.64.2.5 Separation()

`Vector2 FishAgent.Separation ()`

Move away from neighbors

Returns

Definition at line 156 of file FishAgent.cs.

4.64.2.6 setLayers()

`virtual void FishAgent.setLayers () [protected], [virtual]`

Definition at line 33 of file FishAgent.cs.

4.64.2.7 Wander()

`Vector2 FishAgent.Wander ()`

Smooth out movement

Returns

Definition at line 215 of file FishAgent.cs.

4.64.3 Member Data Documentation

4.64.3.1 agentLayer

`LayerMask FishAgent.agentLayer [protected]`

Definition at line 18 of file FishAgent.cs.
4.64.3.2 config

`FishAgentConfig` FishAgent.config [protected]

Definition at line 19 of file FishAgent.cs.

4.64.3.3 predatorLayer

`LayerMask` FishAgent.predatorLayer [protected]

Definition at line 17 of file FishAgent.cs.

4.64.4 Property Documentation

4.64.4.1 Velocity

`Vector3` FishAgent.Velocity [get]

The velocity of the agent

Definition at line 28 of file FishAgent.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/Creatures/FishAgent.cs
Inheritance diagram for FishAgentConfig:
Collaboration diagram for FishAgentConfig:

```
MonoBehaviour

FishAgentConfig
+ CohesionRadius
+ SeparationRadius
+ AlignmentRadius
+ WanderRadius
+ AvoidRadius
+ CohesionWeight
+ SeparationWeight
+ AlignmentWeight
+ WanderWeight
+ AvoidWeight
+ MaxRadius
+ MaxWeight
+ RandomizeSelf()
```

Public Member Functions

- void RandomizeSelf()

  Randomizes the parameters

Public Attributes

- float CohesionRadius = 20f
- float SeparationRadius = 10f
- float AlignmentRadius = 3f
- float WanderRadius = 2f
- float AvoidRadius = 0f
- float CohesionWeight = 80f
- float SeparationWeight = 70f
- float AlignmentWeight = 56f
- float WanderWeight = 30f
- float AvoidWeight = 0f
- float MaxAcceleration = 2f
- float MaxVelocity = 2f
- float Jitter = 2f
- float WanderDistanceRadius = 2f
- float MaxFieldOfViewAngle = 180f
- float waterLevelOffset = -1
Static Public Attributes

- static float MaxRadius = 20f
- static float MaxWeight = 300f

4.65.1 Detailed Description

Definition at line 4 of file FishAgentConfig.cs.

4.65.2 Member Function Documentation

4.65.2.1 RandomizeSelf()

void FishAgentConfig.RandomizeSelf ()

Randomizes the parameters

Definition at line 43 of file FishAgentConfig.cs.

4.65.3 Member Data Documentation

4.65.3.1 AlignmentRadius

float FishAgentConfig.AlignmentRadius = 3f

Definition at line 13 of file FishAgentConfig.cs.

4.65.3.2 AlignmentWeight

float FishAgentConfig.AlignmentWeight = 56f

Definition at line 20 of file FishAgentConfig.cs.

4.65.3.3 AvoidRadius

float FishAgentConfig.AvoidRadius = 0f

Definition at line 15 of file FishAgentConfig.cs.
4.65.3.4 AvoidWeight

```csharp
float FishAgentConfig.AvoidWeight = 0f
```
Definition at line 22 of file FishAgentConfig.cs.

4.65.3.5 CohesionRadius

```csharp
float FishAgentConfig.CohesionRadius = 20f
```
Definition at line 11 of file FishAgentConfig.cs.

4.65.3.6 CohesionWeight

```csharp
float FishAgentConfig.CohesionWeight = 80f
```
Definition at line 18 of file FishAgentConfig.cs.

4.65.3.7 Jitter

```csharp
float FishAgentConfig.Jitter = 2f
```
Definition at line 29 of file FishAgentConfig.cs.

4.65.3.8 MaxAcceleration

```csharp
float FishAgentConfig.MaxAcceleration = 2f
```
Definition at line 25 of file FishAgentConfig.cs.

4.65.3.9 MaxFieldOfViewAngle

```csharp
float FishAgentConfig.MaxFieldOfViewAngle = 180f
```
Definition at line 34 of file FishAgentConfig.cs.
4.65.3.10 MaxRadius

float FishAgentConfig.MaxRadius = 20f [static]

Definition at line 6 of file FishAgentConfig.cs.

4.65.3.11 MaxVelocity

float FishAgentConfig.MaxVelocity = 2f

Definition at line 26 of file FishAgentConfig.cs.

4.65.3.12 MaxWeight

float FishAgentConfig.MaxWeight = 300f [static]

Definition at line 7 of file FishAgentConfig.cs.

4.65.3.13 SeparationRadius

float FishAgentConfig.SeparationRadius = 10f

Definition at line 12 of file FishAgentConfig.cs.

4.65.3.14 SeparationWeight

float FishAgentConfig.SeparationWeight = 70f

Definition at line 19 of file FishAgentConfig.cs.

4.65.3.15 WanderDistanceRadius

float FishAgentConfig.WanderDistanceRadius = 2f

Definition at line 30 of file FishAgentConfig.cs.
4.65.3.16 WanderRadius

```csharp
float FishAgentConfig.WanderRadius = 2f
```
Definition at line 14 of file FishAgentConfig.cs.

4.65.3.17 WanderWeight

```csharp
float FishAgentConfig.WanderWeight = 30f
```
Definition at line 21 of file FishAgentConfig.cs.

4.65.3.18 waterLevelOffset

```csharp
float FishAgentConfig.waterLevelOffset = -1
```
Definition at line 38 of file FishAgentConfig.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/Creatures/FishAgentConfig.cs

4.66 FishingLure Class Reference

Inheritance diagram for FishingLure:
Collaboration diagram for FishingLure:

```
MonoBehaviour

FishingLure
+ IsReeling
+ ReelingSpeed
+ Position
+ Cast()
+ Reel()
```

### Public Member Functions

- **void** `Cast(Vector3 force)`  
  *Cast the lure at the specified force.*
- **void** `Reel(Vector3 targetLocation)`  
  *Reel in the lure.*

### Properties

- **bool** `IsReeling [get]`  
  *Returns true if the lure is currently being reeling in.*
- **float** `ReelingSpeed [get, set]`  
  *Gets and sets the speed to reel in the lure.*
- **Vector3** `Position [get]`  
  *The current position of the lure.*

### 4.66.1 Detailed Description

Definition at line 4 of file FishingLure.cs.

### 4.66.2 Member Function Documentation

#### 4.66.2.1 Cast()

```
void FishingLure.Cast (  
    Vector3 force )
```

*Cast the lure at the specified force.*
Class Documentation

Parameters

| force | The force to cast the lure with. |

Definition at line 108 of file FishingLure.cs.

4.66.2.2 Reel()

```csharp
void FishingLure.Reel ( Vector3 targetLocation )
```

Reel in the lure.

Parameters

| targetLocation | The point from which to reel. |

Definition at line 117 of file FishingLure.cs.

4.66.3 Property Documentation

4.66.3.1 IsReeling

```csharp
bool FishingLure.IsReeling [get]
```

Returns true if the lure is currently being reeling in.

Definition at line 75 of file FishingLure.cs.

4.66.3.2 Position

```csharp
Vector3 FishingLure.Position [get]
```

The current position of the lure.

Definition at line 93 of file FishingLure.cs.
4.66.3 ReelingSpeed

float FishingLure.ReelingSpeed [get], [set]

Gets and sets the speed to reel in the lure.

Definition at line 84 of file FishingLure.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/FishingLure.cs

4.67 FishingRod Class Reference

Inheritance diagram for FishingRod:
Collaboration diagram for FishingRod:

```
MonoBehaviour

Tool
+ ToolName
+ InUse
  # attachJoint
  # toolName
  # unequipActName
+ Use()
+ Equip()
+ Unequip()
+ SetUpTool()

FishingRod
+ WasCast
+ CanCast
+ SetUpTool()
+ Use()
+ Equip()
+ Unequip()
```

Public Member Functions

- `override void SetUpTool(BaseItem itemForTool)`
  
  Sets up the tool so that it is linked to the proper item in the inventory.

- `override void Use()`

  Uses the rod. Casts or reels in the rod depended on its current state.

- `override void Equip()`

  Equip the rod.

- `override void Unequip()`

  Unequip the rod.

Properties

- `bool WasCast [get]`

  Returns true if the fishing rod has been cast.

- `bool CanCast [get]`

  Returns true if the fishing rod can cast from the player's current orientation.
Additional Inherited Members

4.67.1 Detailed Description

Definition at line 7 of file FishingRod.cs.

4.67.2 Member Function Documentation

4.67.2.1 Equip()

override void FishingRod.Equip ( ) [virtual]
Equip the rod.
Implements Tool.
Definition at line 106 of file FishingRod.cs.

4.67.2.2 SetUpTool()

override void FishingRod.SetUpTool ( 
        BaseItem itemForTool ) [virtual]
Sets up the tool so that it is linked to the proper item in the inventory.
Parameters

\begin{verbatim}
itemForTool | Item for tool.
\end{verbatim}
Reimplemented from Tool.
Definition at line 38 of file FishingRod.cs.

4.67.2.3 Unequip()

override void FishingRod.Unequip ( ) [virtual]
Unequip the rod.
Implements Tool.
Definition at line 116 of file FishingRod.cs.
4.67.2.4 Use()

override void FishingRod.Use ( ) [virtual]

Uses the rod. Casts or reels in the rod depended on its current state.

Implements Tool.

Definition at line 87 of file FishingRod.cs.

4.67.3 Property Documentation

4.67.3.1 CanCast

bool FishingRod.CanCast [get]

Returns true if the fishing rod can cast from the player's current orientation.

Definition at line 77 of file FishingRod.cs.

4.67.3.2 WasCast

bool FishingRod.WasCast [get]

Returns true if the fishing rod has been cast.

Definition at line 61 of file FishingRod.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/FishingRod.cs
Inheritance diagram for FishingRodCategory:
Collaboration diagram for FishingRodCategory:

Public Member Functions

- **override ItemCategory GetDuplicate()**
  
  *Gets a copy of the ItemCategory.*

- **override void ReadyCategory()**
  
  *Preps the category for use by loading attributes and actions into lists.*
Additional Inherited Members

4.68.1 Detailed Description

Definition at line 5 of file FishingRodCategory.cs.

4.68.2 Member Function Documentation

4.68.2.1 GetDuplicate()

override ItemCategory FishingRodCategory.GetDuplicate() [virtual]

Gets a copy of the ItemCategory.

Returns

The duplicate.

Reimplemented from EquipableCategory.

Definition at line 11 of file FishingRodCategory.cs.

4.68.2.2 ReadyCategory()

override void FishingRodCategory.ReadyCategory() [virtual]

Preps the category for use by loading attributes and actions into lists.

Reimplemented from EquipableCategory.

Definition at line 32 of file FishingRodCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/FishingRodCategory.cs

Generated by Doxygen
4.69 FishSpawner Class Reference

Inheritance diagram for FishSpawner:
Collaboration diagram for FishSpawner:

![Collaboration diagram](image)

**Protected Member Functions**

- override void `Init()`

  *Override the init function to subscribe to storms, so amount of fish on screen can be controlled*

**Additional Inherited Members**

### 4.69.1 Detailed Description

Definition at line 4 of file FishSpawner.cs.

### 4.69.2 Member Function Documentation
4.69.2.1 Init()

override void FishSpawner.Init () [protected], [virtual]

Override the init function to subscribe to storms, so amount of fish on screen can be controlled

Reimplemented from CreatureSpawner.

Definition at line 58 of file FishSpawner.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Creatures/FishSpawner.cs

### 4.70 FleshCategory Class Reference

Inheritance diagram for FleshCategory:

![Inheritance Diagram](image-url)
Collaboration diagram for FleshCategory:

Public Member Functions

- **override** ItemCategory GetDuplicate() 
  
  Creates a copy of this flesh category.

- **override** void ReadyCategory() 
  
  Readies the item category by adding the attributes and actions it can complete.

- void Cook()
Cooks the item. Decreases health effect.

- void **Eat** ()
  
  *Player* eats item. If there is a health effect, the player will get food poisoning.

Properties

- float **HealthEffect**  **[get, set]**
  
  *Gets or sets the health effect.*

- float **HungerGain**  **[get, set]**
  
  *Gets or sets the hunger gain.*

Additional Inherited Members

4.70.1 Detailed Description

Definition at line 5 of file FleshCategory.cs.

4.70.2 Member Function Documentation

4.70.2.1 **Cook()**

```c
void FleshCategory.Cook ()
```

Cooks the item. Decreases health effect.

Definition at line 85 of file FleshCategory.cs.

4.70.2.2 **Eat()**

```c
void FleshCategory.Eat ()
```

*Player* eats item. If there is a health effect, the player will get food poisoning.

Definition at line 120 of file FleshCategory.cs.
4.70.2.3  GetDuplicate()

```csharp
override ItemCategory FleshCategory.GetDuplicate() [virtual]
```

Creates a copy of this flesh category.

Returns

The duplicate.

Reimplemented from ItemCategory.

Definition at line 48 of file FleshCategory.cs.

4.70.2.4  ReadyCategory()

```csharp
override void FleshCategory.ReadyCategory() [virtual]
```

Readies the item category by adding the attributes and actions it can complete.

Reimplemented from ItemCategory.

Definition at line 69 of file FleshCategory.cs.

4.70.3  Property Documentation

4.70.3.1  HealthEffect

```csharp
float FleshCategory.HealthEffect [get], [set]
```

Gets or sets the health effect.

The health effect.

Definition at line 12 of file FleshCategory.cs.

4.70.3.2  HungerGain

```csharp
float FleshCategory.HungerGain [get], [set]
```

Gets or sets the hunger gain.

The hunger gain.

Definition at line 22 of file FleshCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/FleshCategory.cs
4.71 FloatBehavior Class Reference

Inheritance diagram for FloatBehavior:

```
   MonoBehaviour
   |
   v
FloatBehavior
  + SetFloatHeight()
```

Collaboration diagram for FloatBehavior:

```
   MonoBehaviour
   |
   v
FloatBehavior
  + SetFloatHeight()
```

Public Member Functions

- void SetFloatHeight (float height)

  *Sets the maximum height the object can go until it is completely out of the water*

4.71.1 Detailed Description

Definition at line 4 of file FloatBehavior.cs.
4.71.2 Member Function Documentation

4.71.2.1 SetFloatHeight()

```csharp
void FloatBehavior.SetFloatHeight ( float height )
```

Sets the maximum height the object can go until it is completely out of the water

**Parameters**

| height | Height. |

Definition at line 33 of file FloatBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/FloatBehavior.cs

### 4.72 FloodWater Class Reference

Inheritance diagram for FloodWater:

![Inheritance Diagram](diagram.png)
Collaboration diagram for FloodWater:

```
MonoBehaviour

FloodWater
+ WaterLevelHeight
```

Properties

- `float WaterLevelHeight [get]`
  
  *Gets the height of the water level.*

4.72.1 Detailed Description

Definition at line 4 of file FloodWater.cs.

4.72.2 Property Documentation

4.72.2.1 WaterLevelHeight

`float FloodWater.WaterLevelHeight [get]`

*Gets the height of the water level.*

The height of the water level.

Definition at line 18 of file FloodWater.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/FloodWater.cs
Inheritance diagram for FuelCategory:
Collaboration diagram for FuelCategory:

```
CollectableItem
+ GetPossibleActions()

BaseItem
+ DirtyFlag
+ UpdateExistingFlag
+ RemovalFlag
+ DiscardFlag
+ ItemName
+ Types
+ FlavorText
+ InventorySprite
+ WorldModel
+ ActionModifiedSprites
+ ModifyingActionNames
+ ActionModifiedModels
+ Rarity
+ UpdateItemEvent()
+ BaseItem()
+ BaseItem()
+ BaseItem()
+ InitializeBaseItem()
+ SetUpBaseItem()
+ AddItemCategory()
+ GetItemAttributes()
+ GetPossibleActions()
+ GetNumberOfActionsCompleted()
#baseItem

ItemCategory
+ Attributes
+ Actions
+ SetBaseItem()
+ GetPossibleActions()
+ ReadyCategory()
+ GetDuplicate()
+ SetActionComplete()
+ GetAttribute()
# finishDuplication()

FuelCategory
+ BurnTime
+ GetDuplicate()
+ ReadyCategory()
```

Public Member Functions

- **override** `ItemCategory GetDuplicate()`  
  
  *Creates a copy of this fuel category.*

- **override void** `ReadyCategory()`  
  
  *Readies the item category by adding the attributes and actions it can complete.*
Properties

- float **BurnTime**  [get, set]
  
  *Gets or sets the burn time.*

Additional Inherited Members

4.73.1 Detailed Description

Definition at line 5 of file FuelCategory.cs.

4.73.2 Member Function Documentation

4.73.2.1 GetDuplicate()

```csharp
override ItemCategory FuelCategory.GetDuplicate () [virtual]
```

Creates a copy of this fuel category.

Returns

- The duplicate.

Reimplemented from ItemCategory.

Definition at line 23 of file FuelCategory.cs.

4.73.2.2 ReadyCategory()

```csharp
override void FuelCategory.ReadyCategory () [virtual]
```

Readies the item category by adding the attributes and actions it can complete.

Reimplemented from ItemCategory.

Definition at line 40 of file FuelCategory.cs.

4.73.3 Property Documentation
4.73.3.1 BurnTime

float FuelCategory::BurnTime [get], [set]

Gets or sets the burn time.

The burn time.

Definition at line 12 of file FuelCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/FuelCategory.cs

4.74 FXSplash Class Reference

Inheritance diagram for FXSplash:
Collaboration diagram for FXSplash:

![Collaboration Diagram](image.png)

Public Member Functions

- void setAnimationSpeed (float speed)
  
  Set the animation speed of the splash

Public Attributes

- Animator SplashAnimator

4.74.1 Detailed Description

Definition at line 4 of file FXSplash.cs.

4.74.2 Member Function Documentation

4.74.2.1 setAnimationSpeed()

void FXSplash.setAnimationSpeed (float speed)

Set the animation speed of the splash
Parameters

| speed | How fast the animation should play, default is 1 |

Definition at line 32 of file FXSplash.cs.

4.74.3 Member Data Documentation

4.74.3.1 SplashAnimator

Animator FXSplash.SplashAnimator

Definition at line 6 of file FXSplash.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Effects/FXSplash.cs

4.75 FXSplashManager Class Reference

Inheritance diagram for FXSplashManager:
Collaboration diagram for FXSplashManager:

```
MonoBehaviour

FXSplashManager
+ SplashPoolSize
+ FillPool()
+ CreateSplash()
+ CreateSplash()
+ CreateSplash()
```

Public Member Functions

- void `FillPool()`
  
  Fill the pool with splashes. *(That sounds like a fun summer day to me)*

- void `CreateSplash(Vector3 position, Vector3 scale, float speed=1f, bool allowAddingToPool=false)`
  
  Place and play a water splash effect.

- void `CreateSplash(Vector3 position, float scale, float speed=1f, bool allowAddingToPool=false)`
  
  Place and play a water splash effect. *Uniform Scale.*

- void `CreateSplash(Vector3 position, float speed=1f, bool allowAddingToPool=false)`
  
  Place and play a water splash effect. *Scale defaults to 1.*

Properties

- int `SplashPoolSize` [get, set]

4.75.1 Detailed Description

Definition at line 5 of file FXSplashManager.cs.

4.75.2 Member Function Documentation
4.75.2.1  CreateSplash() [1/3]

void FXSplashManager.CreateSplash (  
    Vector3 position,  
    Vector3 scale,  
    float speed = 1f,  
    bool allowAddingToPool = false)

Place and play a water splash effect.

Parameters

<table>
<thead>
<tr>
<th>parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>The position of the new splash</td>
</tr>
<tr>
<td>scale</td>
<td>The scale of the new splash</td>
</tr>
<tr>
<td>allowAddingToPool</td>
<td>Can a new splash be added to the pool if there are no available splashes?</td>
</tr>
</tbody>
</table>

Definition at line 85 of file FXSplashManager.cs.

4.75.2.2  CreateSplash() [2/3]

void FXSplashManager.CreateSplash (  
    Vector3 position,  
    float scale,  
    float speed = 1f,  
    bool allowAddingToPool = false)

Place and play a water splash effect. Uniform Scale.

Parameters

<table>
<thead>
<tr>
<th>parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>The position of the new splash</td>
</tr>
<tr>
<td>scale</td>
<td>The scale of the new splash</td>
</tr>
<tr>
<td>allowAddingToPool</td>
<td>Can a new splash be added to the pool if there are no available splashes?</td>
</tr>
</tbody>
</table>

Definition at line 96 of file FXSplashManager.cs.

4.75.2.3  CreateSplash() [3/3]

void FXSplashManager.CreateSplash (  
    Vector3 position,  
    float speed = 1f,  
    bool allowAddingToPool = false)

Place and play a water splash effect. Scale defaults to 1.
Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>position</td>
<td>The position of the new splash</td>
</tr>
<tr>
<td>scale</td>
<td>The scale of the new splash</td>
</tr>
<tr>
<td>allowAddingToPool</td>
<td>Can a new splash be added to the pool if there are no available splashes?</td>
</tr>
</tbody>
</table>

Definition at line 107 of file FXSplashManager.cs.

4.75.2.4 FillPool()

```csharp
void FXSplashManager.FillPool();
```

Fill the pool with splashes. (That sounds like a fun summer day to me)

Definition at line 43 of file FXSplashManager.cs.

4.75.3 Property Documentation

4.75.3.1 SplashPoolSize

```csharp
int FXSplashManager.SplashPoolSize {get}, {set}
```

Definition at line 13 of file FXSplashManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Effects/FXSplashManager.cs

4.76 Game Class Reference

Collaboration diagram for Game:

```
Game
+ Instance
+ Player
+ PlayerInstance
+ CityBounds
+ CityInstance
+ DebugMode
+ WeatherInstance
+ EventManager
+ IsRadioActive
+ RadioInstance
and 12 more...
+ Reset()
+ DebugModeDelegate()
```
Public Member Functions

- void **Reset** ()
  
  Resets the game from the start.
- delegate void **DebugModeDelegate** ()

Properties

- static **Game** Instance [get]
  
  The current game instance. If there is no instance, one is created.
- static **Player** Player [get]
  
  Shortcut to calling Instance.PlayerInstance
- **Player** PlayerInstance [get]
  
  The current player instance.
- CityBoundaries CityBounds [get]
  
  Gets the city bounds.
- **City** CityInstance [get, set]
  
  The current city. Returns null if the city has not yet been populated.
- bool DebugMode [get, set]
  
  Turns on debug mode.
- WeatherSystem WeatherInstance [get]
  
  Gets the weather instance.
- **EventManager** EventManager [get]
  
  Gets the event manager instance.
- bool IsRadioActive [get]
- **Radio** RadioInstance [get, set]
- **Clock** ClockInstance [get]
  
  Gets the clock instance.
- float WaterLevelHeight [get]
  
  Gets the height of the water level.
- **PauseSystem** PauseInstance [get]
  
  Gets or private sets the pause instance.
- ItemFactory ItemFactoryInstance [get]
  
  Gets the item factory instance which stores all data about items.
- GameSettings GameSettingsInstance [get]
- WorldItemFactory WorldItemFactoryInstance [get]
  
  Gets the item factory instance which creates items in the world.
- ControlScheme Scheme [get, set]
  
  Gets the control scheme configured for the game.
- GameViewBehavior GameViewInstance [get, set]
  
  Gets or sets the game view instance.
- **GameLoader** Loader [get]
  
  Tracks and reports game loading.
- DeathManager DeathManagerInstance [get]
  
  Gets the death manager instance.
- ItemPoolManager ItemPoolInstance [get, set]
  
  Gets or sets the item pool instance.

Events

- DebugModeDelegate DebugModeSubscription
4.76 Game Class Reference

4.76.1 Detailed Description

Definition at line 1 of file Game.cs.

4.76.2 Member Function Documentation

4.76.2.1 DebugModeDelegate()

delegate void Game.DebugModeDelegate();

4.76.2.2 Reset()

void Game.Reset();

Resets the game from the start.
Definition at line 25 of file Game.cs.

4.76.3 Property Documentation

4.76.3.1 CityBounds

CityBoundaries Game.CityBounds [get]

Gets the city bounds.
The city bounds.
Definition at line 73 of file Game.cs.

4.76.3.2 CityInstance

City Game.CityInstance [get], [set]

The current city. Returns null if the city has not yet been populated.
Definition at line 82 of file Game.cs.
4.76.3.3 ClockInstance

Clock Game.ClockInstance [get]

Gets the clock instance.

The clock instance.

Definition at line 153 of file Game.cs.

4.76.3.4 DeathManagerInstance

DeathManager Game.DeathManagerInstance [get]

Gets the death manager instance.

The death manager instance.

Definition at line 278 of file Game.cs.

4.76.3.5 DebugMode

bool Game.DebugMode [get], [set]

Turns on debug mode.

Definition at line 91 of file Game.cs.

4.76.3.6 EventManager

EventManager Game.EventManager [get]

Gets the event manager instance.

The event manager instance.

Definition at line 124 of file Game.cs.
4.76.3.7  GameSettingsInstance

**GameSettings** Game.GameSettingsInstance  [get]

Gets the game settings instance.

The game settings instance.

Definition at line 229 of file Game.cs.

4.76.3.8  GameViewInstance

**GameViewBehavior** Game.GameViewInstance  [get], [set]

Gets or sets the game view instance.

The game view instance.

Definition at line 259 of file Game.cs.

4.76.3.9  Instance

**Game** Game.Instance  [static], [get]

The current game instance. If there is no instance, one is created.

Definition at line 37 of file Game.cs.

4.76.3.10  IsRadioActive

**bool** Game.IsRadioActive  [get]

Definition at line 130 of file Game.cs.

4.76.3.11  ItemFactoryInstance

**ItemFactory** Game.ItemFactoryInstance  [get]

Gets the item factory instance which stores all data about items.

The item factory instance.

Definition at line 220 of file Game.cs.
4.76.3.12  ItemPoolInstance

ItemPoolManager  Game.ItemPoolInstance  [get], [set]

Gets or sets the item pool instance.
The item pool instance.
Definition at line 288 of file Game.cs.

4.76.3.13  Loader

GameLoader  Game.Loader  [get]

Tracks and reports game loading.
Definition at line 268 of file Game.cs.

4.76.3.14  PauseInstance

PauseSystem  Game.PauseInstance  [get]

 Gets or private sets the pause instance.
The pause instance.
Definition at line 210 of file Game.cs.

4.76.3.15  Player

Player  Game.Player  [static], [get]

Shorcut to calling Instance.PlayerInstance
Definition at line 52 of file Game.cs.

4.76.3.16  PlayerInstance

Player  Game.PlayerInstance  [get]

The current player instance.
Definition at line 63 of file Game.cs.
4.76.3.17  RadioInstance

`Radio` `Game.RadioInstance [get], [set]

Definition at line 138 of file Game.cs.

4.76.3.18  Scheme

`ControlScheme` `Game.Scheme [get], [set]

Gets the control scheme configured for the game.

`Player` control scheme

Definition at line 249 of file Game.cs.

4.76.3.19  WaterLevelHeight

`float` `Game.WaterLevelHeight [get]

Gets the height of the water level.

The height of the water level.

Definition at line 184 of file Game.cs.

4.76.3.20  WeatherInstance

`WeatherSystem` `Game.WeatherInstance [get]

Gets the weather instance.

The weather instance.

Definition at line 114 of file Game.cs.

4.76.3.21  WorldItemFactoryInstance

`WorldItemFactory` `Game.WorldItemFactoryInstance [get]

Gets the item factory instance which creates items in the world.

The world item factory instance.

Definition at line 239 of file Game.cs.
4.76.4 Event Documentation

4.76.4.1 DebugModeSubscription

```
DebugModeDelegate Game.DebugModeSubscription
```

Definition at line 107 of file Game.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Game.cs

4.77 GameLoader Class Reference

Tracks all loading tasks and determines overall loading percentage.

Collaboration diagram for GameLoader:

```
+ PercentageComplete
+ CurrentTask
+ GameLoader()
+ CreateGameLoaderTask()
```

Public Member Functions

- `GameLoader()`
  
  Instantiate a new `GameLoader`

- `GameLoaderTask CreateGameLoaderTask(string name)`
  
  Creates a loading task and adds it to the list of tracked tasks.

Properties

- float `PercentageComplete` [get]
  
  Get overall loading percentage complete.

- string `CurrentTask` [get]
  
  Get the name of the currently processing task.
4.77.1 Detailed Description

Tracks all loading tasks and determines overall loading percentage.

Definition at line 8 of file GameLoader.cs.

4.77.2 Constructor & Destructor Documentation

4.77.2.1 GameLoader()

GameLoader.GameLoader ( )

Instatiate a new GameLoader

Definition at line 15 of file GameLoader.cs.

4.77.3 Member Function Documentation

4.77.3.1 CreateGameLoaderTask()

GameLoaderTask GameLoader.CreateGameLoaderTask ( string name )

Creates a loading task and adds it the the list of tracked tasks.

Parameters

| name | Name describing task. |

Returns

An instance of the task.

Definition at line 25 of file GameLoader.cs.

4.77.4 Property Documentation
4.77.4.1 CurrentTask

string GameLoader.CurrentTask  [get]

Get the name of the currently processing task.

Definition at line 59 of file GameLoader.cs.

4.77.4.2 PercentageComplete

float GameLoader.PercentageComplete  [get]

Get overall loading percentage complete.

Definition at line 36 of file GameLoader.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Loading/GameLoader.cs

4.78 GameLoaderTask Class Reference

Tracks information for a loading task.

Collaboration diagram for GameLoaderTask:

<table>
<thead>
<tr>
<th>GameLoaderTask</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Name</td>
</tr>
<tr>
<td>+ PercentageComplete</td>
</tr>
<tr>
<td>+ GameLoaderTask()</td>
</tr>
</tbody>
</table>

Public Member Functions

- GameLoaderTask (string name)
  
  Creates a task for the game loader to track.
Properties

- string **Name** [get]
  
  *Name* that describes the task.

- float **PercentageComplete** [get, set]
  
  The percentage completeness of the task.

4.78.1 Detailed Description

Tracks information for a loading task.

Definition at line 7 of file GameLoaderTask.cs.

4.78.2 Constructor & Destructor Documentation

4.78.2.1 GameLoaderTask()

```csharp
GameLoaderTask.GameLoaderTask ( 
  string name )
```

Creates a task for the game loader to track.

Parameters

- `name` Name that describes the task.

Definition at line 39 of file GameLoaderTask.cs.

4.78.3 Property Documentation

4.78.3.1 Name

```csharp
string GameLoaderTask.Name [get]
```

Name that describes the task.

Definition at line 15 of file GameLoaderTask.cs.
4.78.3.2 PercentageComplete

float GameLoaderTask.PercentageComplete [get], [set]

The percentage completeness of the task.

Definition at line 24 of file GameLoaderTask.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Loading/GameLoaderTask.cs

4.79 GameSettings Class Reference

Collaboration diagram for GameSettings:

```
GameSettings
+ SoundOn
+ VolumeValue
+ ProceduralCityGenerationSeed
+ Scheme
+ GameSettings()
```

Public Member Functions

- **GameSettings ()**
  
  *Initializes a new instance of the GameSettings class.*

Properties

- **bool SoundOn [get, set]**
  
  *Gets or sets a value indicating whether this AssemblyCSharp.GameSettings sound on.*

- **float VolumeValue [get, set]**
  
  *Gets or sets the volume value.*

- **int ProceduralCityGenerationSeed [get, set]**
  
  *Gets or sets the procedural city generation seed.*

- **ControlScheme Scheme [get, set]**
  
  *Gets or sets the scheme.*
4.79.1 Detailed Description

Definition at line 1 of file GameSettings.cs.

4.79.2 Constructor & Destructor Documentation

4.79.2.1 GameSettings()

GameSettings.GameSettings()

Initializes a new instance of the GameSettings class.

Definition at line 6 of file GameSettings.cs.

4.79.3 Property Documentation

4.79.3.1 ProceduralCityGenerationSeed

int GameSettings.ProceduralCityGenerationSeed [get], [set]

Gets or sets the procedural city generation seed.

The procedural city generation seed.

Definition at line 41 of file GameSettings.cs.

4.79.3.2 Scheme

ControlScheme GameSettings.Scheme [get], [set]

Gets or sets the scheme.

The scheme.

Definition at line 51 of file GameSettings.cs.
### 4.79.3.3 SoundOn

```csharp
bool GameSettings.SoundOn [get], [set]
```

Gets or sets a value indicating whether this AssemblyCSharp.GameSettings sound on.

`true` if sound on; otherwise, `false`.

Definition at line 21 of file GameSettings.cs.

### 4.79.3.4 VolumeValue

```csharp
float GameSettings.VolumeValue [get], [set]
```

Gets or sets the volume value.

The volume value.

Definition at line 31 of file GameSettings.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/GameSettings.cs

### 4.80 GameViewBehavior Class Reference

Inheritance diagram for GameViewBehavior:
Collaboration diagram for GameViewBehavior:

```plaintext
MonoBehaviour

GameViewBehavior
+ OnPauseClick()
+ OnInventoryClick()
+ OnInventoryOpen()
+ OnRadioClick()
+ OnResumeClick()
+ OnCraftingClick()
```

Public Member Functions

- void **OnPauseClick** ()
  
  Loads the pause panel.

- void **OnInventoryClick** ()
  
  Loads the inventory panel.

- void **OnInventoryOpen** ()

- void **OnRadioClick** ()
  
  Loads the radio panel.

- void **OnResumeClick** ()
  
  Returns to the game scene.

- void **OnCraftingClick** ()
  
  Loads the crafting panel.

### 4.80.1 Detailed Description

Definition at line 5 of file GameViewBehavior.cs.

### 4.80.2 Member Function Documentation
4.80.2.1 OnCraftingClick()

void GameViewBehavior.OnCraftingClick ( )

Loads the crafting panel.

Definition at line 188 of file GameViewBehavior.cs.

4.80.2.2 OnInventoryClick()

void GameViewBehavior.OnInventoryClick ( )

Loads the inventory panel.

Definition at line 117 of file GameViewBehavior.cs.

4.80.2.3 OnInventoryOpen()

void GameViewBehavior.OnInventoryOpen ( )

Definition at line 134 of file GameViewBehavior.cs.

4.80.2.4 OnPauseClick()

void GameViewBehavior.OnPauseClick ( )

Loads the pause panel.

Definition at line 104 of file GameViewBehavior.cs.

4.80.2.5 OnRadioClick()

void GameViewBehavior.OnRadioClick ( )

Loads the radio panel.

Definition at line 151 of file GameViewBehavior.cs.
4.80.6 OnResumeClick()

```csharp
void GameViewBehavior.OnResumeClick()
```

Returns to the game scene.

Definition at line 164 of file GameViewBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/GameViewScene/GameViewBehavior.cs

4.81 GenerationUtility Class Reference

Collaboration diagram for GenerationUtility:

```
+ DistributeSeedPoints()
+ ToAlignedVector3()
+ ToAlignedVector2()
+ AddNonDuplicateVertex()
+ IsPointInPolygon()
+ GetMidpoint()
+ SortVertices()
+ IsClockwise()
+ GetMostCommonVertex()
```

Static Public Member Functions

- static List<Vector2> DistributeSeedPoints (int seed, Bounds bounds, float distanceBetweenPoints)
  Generates a list of 2D points using poison distribution, aligned perpendicular to the y-axis.
- static Vector3 ToAlignedVector3(Vector2 vector)
  Converts 2D point aligned perpendicular to the y-axis to 3D point.
- static Vector3 ToAlignedVector2(Vector3 vector)
  Converts a 3D point to a point in 2D point aligned perpendicular to the y-axis.
- static List<Vector2> AddNonDuplicateVertex(List<Vector2> list, Vector2 vertex)
  Adds a vertex to a list if it's not a duplicate of another already in the list.
- static bool IsPointInPolygon(Vector2 point, ref Vector2[] polygon)
  Determines if a point is inside a polygon defined by the list of verts.
- static Vector2 GetMidpoint(List<Vector2> points)
  Get the midmpoint of a list of points by calculating the average.
- static List<Vector2> SortVertices(List<Vector2> points, Vector2 center)
  Sort the list of vertices clockwise around a center point.
- static int IsClockwise(Vector2 first, Vector2 second, Vector2 origin)
  Returns 1 if first comes before second in clockwise order. Returns -1 if second comes before first. Returns 0 if the points are identical.
- static Vector3 GetMostCommonVertex(District[] districts)
  Get the location of the vertex that is shared by the most districts. Does not garuntee any order.
4.81.1 Detailed Description

Definition at line 5 of file GenerationUtility.cs.

4.81.2 Member Function Documentation

4.81.2.1 AddNonDuplicateVertex()

static List<Vector2> GenerationUtility.AddNonDuplicateVertex ( 
    List<Vector2> list, 
    Vector2 vertex) [static]

Adds a vertex to a list if it's not a duplicate of another already in the list.

Parameters

<table>
<thead>
<tr>
<th>list</th>
<th>The list.</th>
</tr>
</thead>
<tbody>
<tr>
<td>vertex</td>
<td>The vertex.</td>
</tr>
</tbody>
</table>

Returns

The list.

Definition at line 57 of file GenerationUtility.cs.

4.81.2.2 DistributeSeedPoints()

static List<Vector2> GenerationUtility.DistributeSeedPoints ( 
    int seed, 
    Bounds bounds, 
    float distanceBetweenPoints ) [static]

Generates a list of 2D points using poison distribution, aligned perpendicular the the y-axis.

Parameters

<table>
<thead>
<tr>
<th>seed</th>
<th>The generation seed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>bounds</td>
<td>The bounds of the distribution.</td>
</tr>
<tr>
<td>distanceBetweenPoints</td>
<td>The minimum distance between each seed point.</td>
</tr>
</tbody>
</table>

Returns

The list of distributed points, perpendicular to y-axis.
4.81.2.3 GetMidpoint()

```csharp
static Vector2 GenerationUtility.GetMidpoint (  
    List< Vector2 > points ) [static]
```

Get the midpoint of a list of points by calculating the average.

**Parameters**

- `points` List of points from which to calculate the midpoint.

**Returns**

The midpoint.

Definition at line 102 of file GenerationUtility.cs.

4.81.2.4 GetMostCommonVertex()

```csharp
static Vector3 GenerationUtility.GetMostCommonVertex (  
    District[] districts ) [static]
```

Get the location of the vertex that is shared by the most districts. Does not guarantee any order.

**Parameters**

- `districts` List of populated districts.

**Returns**

Location of vertex shared by most districts.

Definition at line 170 of file GenerationUtility.cs.

4.81.2.5 IsClockwise()

```csharp
static int GenerationUtility.IsClockwise (  
    Vector2 first,  
    Vector2 second,  
    Vector2 origin ) [static]
```

Returns 1 if first comes before second in clockwise order. Returns -1 if second comes before first. Returns 0 if the points are identical.
4.81.2.6 IsPointInPolygon()

```csharp
static bool GenerationUtility.IsPointInPolygon( 
    Vector2 point, 
    ref Vector2[] polygon) [static]
```

Determines if a point is inside a polygon defined by the list of verts.

Parameters

<table>
<thead>
<tr>
<th>point</th>
<th>polygon</th>
</tr>
</thead>
</table>

Returns

Definition at line 72 of file GenerationUtility.cs.

4.81.2.7 SortVerticies()

```csharp
static List<Vector2> GenerationUtility.SortVerticies( 
    List<Vector2> points, 
    Vector2 center) [static]
```

Sort the list of vertices clockwise around a center point.

Parameters

<table>
<thead>
<tr>
<th>points</th>
<th>center</th>
</tr>
</thead>
</table>

The list of vertices to sort.
The center points around which the vertices are sorted.

Returns

The sorted list of points.

Definition at line 119 of file GenerationUtility.cs.
4.81.2.8 ToAlignedVector2()

```csharp
static Vector3 GenerationUtility.ToAlignedVector2 (Vector3 vector) // static
```

Converts a 3D point to a point in 2D point aligned perpendicular to the y-axis.

**Parameters**

- `vector` Point to convert.

**Returns**

- Point in 2D point aligned perpendicular to the y-axis

Definition at line 46 of file GenerationUtility.cs.

4.81.2.9 ToAlignedVector3()

```csharp
static Vector3 GenerationUtility.ToAlignedVector3 (Vector2 vector) // static
```

Converts 2D point aligned perpendicular to the y-axis to 3D point.

**Parameters**

- `vector` Point to convert.

**Returns**

- Point in 3D space.

Definition at line 36 of file GenerationUtility.cs.

The documentation for this class was generated from the following file:

- `Assets/Scripts/City/GenerationUtility.cs`
4.82 GoogleDrive Class Reference

Collaboration diagram for GoogleDrive:

<table>
<thead>
<tr>
<th>GoogleDrive</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ GoogleURL</td>
</tr>
<tr>
<td>+ GetterURL</td>
</tr>
<tr>
<td>+ MethodType</td>
</tr>
<tr>
<td>+ ConnectedToInternet()</td>
</tr>
<tr>
<td>+ GetOnlineDriveDocument()</td>
</tr>
<tr>
<td>+ CertificateValidationCallback()</td>
</tr>
</tbody>
</table>

Static Public Member Functions

- static bool ConnectedToInternet ()
  Checks if connected to internet. Warning: This should only as few times as possible as this is an expensive call.
- static string GetOnlineDriveDocument (string fileName)
  Gets google drive document if file found. On fail this will return the local document.
- static bool CertificateValidationCallback (System.Object sender, X509Certificate certificate, X509Chain chain, SslPolicyErrors sslPolicyErrors)
  Validates certification errors that can occur on Windows.

Static Public Attributes

- static string GoogleURL = "http://www.google.com"
- static string GetterURL = "https://glassprisongames-googlegetter.herokuapp.com"
- static string MethodType = "GET"

4.82.1 Detailed Description

Definition at line 9 of file GoogleDrive.cs.

4.82.2 Member Function Documentation
4.82.2.1 CertificateValidationCallback()

```csharp
static bool GoogleDrive.CertificateValidationCallback(
    System.Object sender,
    X509Certificate certificate,
    X509Chain chain,
    SslPolicyErrors sslPolicyErrors) [static]
```

Validates certification errors that can occur on Windows.

Definition at line 111 of file GoogleDrive.cs.

4.82.2.2 ConnectedToInternet()

```csharp
static bool GoogleDrive.ConnectedToInternet() [static]
```

Checks if connected to internet. Warning: This should only as few times as possible as this is an expensive call.

Returns

true, if connected to internet, false otherwise.

Definition at line 20 of file GoogleDrive.cs.

4.82.2.3 GetOnlineDriveDocument()

```csharp
static string GoogleDrive.GetOnlineDriveDocument(
    string fileName) [static]
```

Gets google drive document if file found. On fail this will return the local document.

Returns

The drive document.

Parameters

| fileName | File name. |

Definition at line 38 of file GoogleDrive.cs.

4.82.3 Member Data Documentation
4.82.3.1 GetterURL


Definition at line 12 of file GoogleDrive.cs.

4.82.3.2 GoogleURL

string GoogleDrive.GoogleURL = "http://www.google.com" [static]

Definition at line 11 of file GoogleDrive.cs.

4.82.3.3 MethodType

string GoogleDrive.MethodType = "GET" [static]

Definition at line 13 of file GoogleDrive.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/File/GoogleDrive.cs

4.83 GridLayoutManager Class Reference

Inheritance diagram for GridLayoutManager:
Collaboration diagram for GridLayoutManager:

MonoBehaviour

GridLayoutManager
+ ElementsPerRow
+ UseMaxRows
+ MaxRows
+ SetGridSize()
+ CheckGridSize()

Public Member Functions

- void SetGridSize ()
  
  Sets the size of the grid cells.
- void CheckGridSize ()

Public Attributes

- int ElementsPerRow
- bool UseMaxRows
- int MaxRows

4.83.1 Detailed Description

Definition at line 6 of file GridLayoutManager.cs.

4.83.2 Member Function Documentation

4.83.2.1 CheckGridSize()

void GridLayoutManager.CheckGridSize ()

Definition at line 75 of file GridLayoutManager.cs.
4.83.2.2 SetGridSize()

```csharp
void GridLayoutManager.SetGridSize()
```

Sets the size of the grid cells.

Definition at line 46 of file GridLayoutManager.cs.

4.83.3 Member Data Documentation

4.83.3.1 ElementsPerRow

```csharp
int GridLayoutManager.ElementsPerRow
```

Definition at line 9 of file GridLayoutManager.cs.

4.83.3.2 MaxRows

```csharp
int GridLayoutManager.MaxRows
```

Definition at line 15 of file GridLayoutManager.cs.

4.83.3.3 UseMaxRows

```csharp
bool GridLayoutManager.UseMaxRows
```

Definition at line 12 of file GridLayoutManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/Utility/GridLayoutManager.cs

4.84 HasSelectionInterface Interface Reference

Collaboration diagram for HasSelectionInterface:

```
HasSelectionInterface
```

```
+ ItemSelectedCallbackAction()
```
Public Member Functions

- void ItemSelectedCallbackAction (string itemName)

  Callback action that will be fired off when an item has been selected.

4.84.1 Detailed Description

Definition at line 1 of file HasSelectionInterface.cs.

4.84.2 Member Function Documentation

4.84.2.1 ItemSelectedCallbackAction()

void HasSelectionInterface.ItemSelectedCallbackAction ( 
  string itemName )

Callback action that will be fired off when an item has been selected.

Parameters

  | itemName | Item name. |

The documentation for this interface was generated from the following file:

- Assets/Scripts/Items/HasSelectionInterface.cs
4.85 HealthRateManager Class Reference

Inheritance diagram for HealthRateManager:

```
StatRate
+ Units
+ PerSeconds
+ CurrentStat
+ MaxStat

+ StatRate()
+ ChangeRateValues()
+ ApplyRateToStat()
# StatRate()

HealthRateManager
+ HealthRateManager()
+ UseDefaultHealthRate()
+ TakeFallDamage()
+ UseHungerZeroHealthReductionRate()
+ UseWarmthZeroHealthReductionRate()
+ UseHealthEnergy()
```
Collaboration diagram for HealthRateManager:

```
StatRate
+ Units
+ PerSeconds
+ CurrentStat
+ MaxStat
+ StatRate()
+ ChangeRateValues()
+ ApplyRateToStat()
# StatRate()

HealthRateManager
+ HealthRateManager()
+ UseDefaultHealthRate()
+ TakeFallDamage()
+ UseHungerZeroHealthReductionRate()
+ UseWarmthZeroHealthReductionRate()
+ UseHealthEnergy()
```

Public Member Functions

- `HealthRateManager()`
  Initializes a new instance of the `HealthRateManager` class.

- `void UseDefaultHealthRate()`
  Uses the default health rate.

- `void TakeFallDamage(int fallDamageAmount)`
  Takes the fall damage.

- `void UseHungerZeroHealthReductionRate()`
  Uses the hunger zero health reduction rate.

- `void UseWarmthZeroHealthReductionRate()`
  Uses the warmth zero health reduction rate.

- `void UseHealthEnergy(int amountOfHealthUnitsAffected)`
  Uses the health energy.

Additional Inherited Members

4.85.1 Detailed Description

Definition at line 3 of file HealthRateManager.cs.
4.85.2 Constructor & Destructor Documentation

4.85.2.1 HealthRateManager()

`HealthRateManager::HealthRateManager()`

Initializes a new instance of the `HealthRateManager` class.

Definition at line 21 of file `HealthRateManager.cs`.

4.85.3 Member Function Documentation

4.85.3.1 TakeFallDamage()

`void HealthRateManager::TakeFallDamage(int fallDamageAmount)`

Takes the fall damage.

**Parameters**

- **fallDamageAmount**: Fall damage amount.

Definition at line 49 of file `HealthRateManager.cs`.

4.85.3.2 UseDefaultHealthRate()

`void HealthRateManager::UseDefaultHealthRate()`

Uses the default health rate.

Definition at line 39 of file `HealthRateManager.cs`.

4.85.3.3 UseHealthEnergy()

`void HealthRateManager::UseHealthEnergy(int amountOfHealthUnitsAffected)`

Uses the health energy.
Parameters

| amountOfHealthUnitsAffected | Amount of health units affected. |

Definition at line 79 of file HealthRateManager.cs.

### 4.85.3.4 UseHungerZeroHealthReductionRate()

```csharp
void HealthRateManager.UseHungerZeroHealthReductionRate();
```

Uses the hunger zero health reduction rate.
Definition at line 60 of file HealthRateManager.cs.

### 4.85.3.5 UseWarmthZeroHealthReductionRate()

```csharp
void HealthRateManager.UseWarmthZeroHealthReductionRate();
```

Uses the warmth zero health reduction rate.
Definition at line 69 of file HealthRateManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Player/HealthRateManager.cs

### 4.86 HUDBehavior Class Reference

Inheritance diagram for HUDBehavior:

```
MonoBehaviour
  

  HUDBehavior
    + UpdateHealthSlider()
    + UpdateWarmthSlider()
    + UpdateHungerSlider()
```

Generated by Doxygen
Collaboration diagram for HUDBehavior:

```
MonoBehaviour

HUDBehavior
+ UpdateHealthSlider()
+ UpdateWarmthSlider()
+ UpdateHungerSlider()
```

**Public Member Functions**

- `void UpdateHealthSlider ()`
  
  Updates the health slider.

- `void UpdateWarmthSlider ()`
  
  Updates the warmth slider.

- `void UpdateHungerSlider ()`
  
  Updates the hunger slider.

### 4.86.1 Detailed Description

Definition at line 5 of file HUDBehavior.cs.

### 4.86.2 Member Function Documentation

#### 4.86.2.1 UpdateHealthSlider()

void HUDBehavior.UpdateHealthSlider ()

Updates the health slider.

Definition at line 29 of file HUDBehavior.cs.
4.86.2.2 UpdateHungerSlider()

```csharp
void HUDBehavior.UpdateHungerSlider()
```

Updates the hunger slider.

Definition at line 45 of file HUDBehavior.cs.

4.86.2.3 UpdateWarmthSlider()

```csharp
void HUDBehavior.UpdateWarmthSlider()
```

Updates the warmth slider.

Definition at line 37 of file HUDBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/GameViewScene/HUDBehavior.cs

4.87 HungerRateManager Class Reference

Inheritance diagram for HungerRateManager:

```
+ StatRate
 + Units
 + PerSeconds
 + CurrentStat
 + MaxStat
 + StatRate()
 + ChangeRateValues()
 + ApplyRateToStat()
 # StatRate()

+ HungerRateManager()
 + UseDefaultHungerReductionRate()
 + UseFoodEnergy()
```
Collaboration diagram for HungerRateManager:

```
StatRate
+ Units
+ PerSeconds
+ CurrentStat
+ MaxStat
+ StatRate()
+ ChangeRateValues()
+ ApplyRateToStat()
# StatRate()

HungerRateManager
+ HungerRateManager()
+ UseDefaultHungerReductionRate()
+ UseFoodEnergy()
```

Public Member Functions

- **HungerRateManager ()**
  *Initializes a new instance of the HungerRateManager class.*

- **void UseDefaultHungerReductionRate ()**
  *Uses the default hunger reduction rate.*

- **void UseFoodEnergy (int amountOfHungerUnitsAffected)**
  *Uses the food energy.*

Additional Inherited Members

4.87.1 Detailed Description

Definition at line 3 of file HungerRateManager.cs.

4.87.2 Constructor & Destructor Documentation
4.87.2.1 HungerRateManager()

HungerRateManager.HungerRateManager ( )

Initializes a new instance of the HungerRateManager class.

Definition at line 13 of file HungerRateManager.cs.

4.87.3 Member Function Documentation

4.87.3.1 UseDefaultHungerReductionRate()

void HungerRateManager.UseDefaultHungerReductionRate ( )

Uses the default hunger reduction rate.

Definition at line 27 of file HungerRateManager.cs.

4.87.3.2 UseFoodEnergy()

void HungerRateManager.UseFoodEnergy ( int amountOfHungerUnitsAffected )

Uses the food energy.

Parameters

| amountOfHungerUnitsAffected | Amount of hunger units affected. |

Definition at line 37 of file HungerRateManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Player/HungerRateManager.cs
Inheritance diagram for Idol:
Collaboration diagram for Idol:

```
MonoBehaviour

Tool
+ ToolName
+ InUse
# attachJoint
# toolName
# unequipActName
+ Use()
+ Equip()
+ Unequip()
+ SetUpTool()

Idol
+ SetUpTool()
+ Equip()
+ Unequip()
+ Use()
```

Public Member Functions

- **override void** `SetUpTool (BaseItem itemForTool)`
  
  Sets up the tool so that it is linked to the proper item in the inventory.

- **override void** `Equip ()`
  
  Equip this instance.

- **override void** `Unequip ()`
  
  Unequip this instance.

- **override void** `Use ()`
  
  Use this instance.

Additional Inherited Members

4.88.1 Detailed Description

Definition at line 6 of file Idol.cs.
4.88.2 Member Function Documentation

4.88.2.1 Equip()

```csharp
override void Idol.Equip() [virtual]
```

Equip this instance.

Implements Tool.

Definition at line 41 of file Idol.cs.

4.88.2.2 SetUpTool()

```csharp
override void Idol.SetUpTool(BaseItem itemForTool) [virtual]
```

Sets up the tool so that it is linked to the proper item in the inventory.

Parameters

- `itemForTool` Item for tool.

Reimplemented from Tool.

Definition at line 26 of file Idol.cs.

4.88.2.3 Unequip()

```csharp
override void Idol.Unequip() [virtual]
```

Unequip this instance.

Implements Tool.

Definition at line 50 of file Idol.cs.
4.88.2.4 Use()

```csharp
override void Idol.Use () [virtual]
```

Use this instance.

Implements Tool.

Definition at line 60 of file Idol.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/Idol.cs
Inheritance diagram for IdolCategory:
Collaboration diagram for IdolCategory:

Public Member Functions

- **override void ReadyCategory ()**
  Preps the category for use by loading attributes and actions into lists.

- **void Equip ()**
  Equip this item.

- **abstract void ApplyBenefit ()**
Applies the benefit.

- abstract void RemoveBenefit()
  
  Removes the benefit.

Additional Inherited Members

4.89.1 Detailed Description

Definition at line 6 of file IdolCategory.cs.

4.89.2 Member Function Documentation

4.89.2.1 ApplyBenefit()

abstract void IdolCategory.ApplyBenefit () [pure virtual]

Applies the benefit.

Implemented in WarmthIdolCategory.

4.89.2.2 Equip()

void IdolCategory.Equip ()

Equip this item.

Definition at line 32 of file IdolCategory.cs.

4.89.2.3 ReadyCategory()

override void IdolCategory.ReadyCategory () [virtual]

Preps the category for use by loading attributes and actions into lists.

Reimplemented from EquipableCategory.

Reimplemented in WarmthIdolCategory.

Definition at line 11 of file IdolCategory.cs.
4.89.2.4 RemoveBenefit()

abstract void IdolCategory.RemoveBenefit () [pure virtual]

Removes the benefit.

Implemented in WarmthIdolCategory.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/IdolCategory.cs

4.90 Ingredient Struct Reference

Collaboration diagram for Ingredient:

```
Ingredient
+ IngredientName
+ Amount
+ UseType
+ AssociatedStackId
+ Ingredient()
```

Public Member Functions

- Ingredient (SelectedIngredientButton selectionInformation, string type)  
  *Initializes a new instance of the Ingredient struct.*

Properties

- string IngredientName [get, set]  
  *Gets or sets the name of the ingredient.*
- int Amount [get, set]  
  *Gets or sets the amount of the ingredient needed by the recipe.*
- string UseType [get, set]  
  *Gets or sets the type of item that this ingredient is being used for.*
- string AssociatedStackId [get, set]  
  *Gets or sets the associated stack identifier.*
4.90.1 Detailed Description

Definition at line 1 of file Ingredient.cs.

4.90.2 Constructor & Destructor Documentation

4.90.2.1 Ingredient()

Ingredient.Ingredient (  
    SelectedIngredientButton selectionInformation,  
    string type )

Initializes a new instance of the Ingredient struct.

Parameters

<table>
<thead>
<tr>
<th>selectionInformation</th>
<th>The button that has stored all information about the ingredient that has been selected.</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>Type of item that the ingredient fulfills.</td>
</tr>
</tbody>
</table>

Definition at line 48 of file Ingredient.cs.

4.90.3 Property Documentation

4.90.3.1 Amount

int Ingredient.Amount [get], [set]

Gets or sets the amount of the ingredient needed by the recipe.

The amount.

Definition at line 18 of file Ingredient.cs.

4.90.3.2 AssociatedStackId

string Ingredient.AssociatedStackId [get], [set]

Gets or sets the associated stack identifier.

The associated stack identifier.

Definition at line 38 of file Ingredient.cs.
4.90.3.3 IngredientName

string Ingredient.IngredientName [get], [set]

Gets or sets the name of the ingredient.

The name of the ingredient.

Definition at line 8 of file Ingredient.cs.

4.90.3.4 UseType

string Ingredient.UseType [get], [set]

Gets or sets the type of item that this ingredient is being used for.

The type of the use.

Definition at line 28 of file Ingredient.cs.

The documentation for this struct was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/Ingredient.cs

4.91 IngredientButtonBehavior Class Reference

Inheritance diagram for IngredientButtonBehavior:
Collaboration diagram for IngredientButtonBehavior:

![Collaboration Diagram](image-url)

Public Member Functions

- delegate void `UpdateIngredientEvent()`  
  Delegate function that takes in a `baseItem`
- void `SetUpIngredient(string name, int amount, string stackId)`  
  Sets up ingredient.
- void `SelectIngredient()`  
  Selects the ingredient.
- void `DeselectIngredient()`  
  Deselects the ingredient.

Events

- `UpdateIngredientEvent UpdateIngredientSelection`  
  Event that can be subscribed to by functions of `UpdateItemEvent` format

4.91.1 Detailed Description

Definition at line 5 of file `IngredientButtonBehavior.cs`.

4.91.2 Member Function Documentation
4.91.2.1 DeselectIngredient()

void IngredientButtonBehavior.DeselectIngredient ( )

Deselects the ingredient.

Definition at line 86 of file IngredientButtonBehavior.cs.

4.91.2.2 SelectIngredient()

void IngredientButtonBehavior.SelectIngredient ( )

Selects the ingredient.

Definition at line 57 of file IngredientButtonBehavior.cs.

4.91.2.3 SetUpIngredient()

void IngredientButtonBehavior.SetUpIngredient ( string name, int amount, string stackId )

Sets up ingredient.

Parameters

<table>
<thead>
<tr>
<th>name</th>
<th>Name of item.</th>
</tr>
</thead>
<tbody>
<tr>
<td>amount</td>
<td>Amount of item.</td>
</tr>
<tr>
<td>stackId</td>
<td>Stack identifier of the stack this represents.</td>
</tr>
</tbody>
</table>

Definition at line 40 of file IngredientButtonBehavior.cs.

4.91.2.4 UpdateIngredientEvent()

delegate void IngredientButtonBehavior.UpdateIngredientEvent ( )

Delegate function that takes in a baseltem

4.91.3 Event Documentation
4.91.3.1 UpdateIngredientSelection

**UpdateIngredientEvent** IngredientButtonBehavior.UpdateIngredientSelection

Event that can be subscribed to by functions of UpdateItemEvent format

Definition at line 101 of file IngredientButtonBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/CraftingGUI/IngredientButtonBehavior.cs

4.92 InteractableObject Class Reference

Inheritance diagram for InteractableObject:

Collaboration diagram for InteractableObject:
Public Member Functions

- virtual void SetUp ()
  Sets up the InteractableObject.
- void PerformAction ()
  Have the player perform the object’s action.
- void SetAction (UnityAction newAction)
  Set the object’s action which the player may activate.

Protected Attributes

- bool setupComplete = false

Properties

- bool Show [get, set]
  Shows or hides text describing the interactable object action to the player.
- string Text [get, set]
  Text describing the interactive object’s action.

4.92.1 Detailed Description

Definition at line 6 of file InteractableObject.cs.

4.92.2 Member Function Documentation

4.92.2.1 PerformAction()

void InteractableObject::PerformAction ()

Have the player perform the object’s action.

Definition at line 101 of file InteractableObject.cs.

4.92.2.2 SetAction()

void InteractableObject::SetAction (UnityAction newAction)

Set the object’s action which the player may activate.
Parameters

| newAction | The action to be performed. |

Definition at line 113 of file InteractableObject.cs.

4.92.2.3 SetUp()

virtual void InteractableObject.SetUp ( ) [virtual]

Sets up the InteractableObject.

Reimplemented in FireInteractable, PickUpItem, InventoryInteractable, RaftInteractable, and ShelterInteractable.

Definition at line 45 of file InteractableObject.cs.

4.92.3 Member Data Documentation

4.92.3.1 setupComplete

bool InteractableObject.setupComplete = false [protected]

Definition at line 15 of file InteractableObject.cs.

4.92.4 Property Documentation

4.92.4.1 Show

bool InteractableObject.Show [get], [set]

Shows or hides text describing the interactable object action to the player.

Definition at line 60 of file InteractableObject.cs.
4.92.4.2 Text

string InteractableObject.Text [get], [set]

Text describing the interactive object's action.

Definition at line 83 of file InteractableObject.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/InteractableObject.cs

4.93 InteractableRadioModel Class Reference

Inheritance diagram for InteractableRadioModel:

```
MonoBehaviour

InteractableRadioModel
+ SetUpRadioCanvas()
+ ActivateRadio()
+ DeactivateRadio()
+ PushPowerButton()
+ PushVolumeUpButton()
+ PushVolumeDownButton()
+ SetKnobRotation()
+ SetSlider()
```
Collaboration diagram for InteractableRadioModel:

![Collaboration diagram](image)

**Public Member Functions**

- void `SetUpRadioCanvas()`  
  Set up all the button start positions.
- void `ActivateRadio()`  
  Activates the radio mesh's and moves the radio from the offscreen position to the onscreen position.
- void `DeactivateRadio()`  
  Calls the deactivate coroutine
- void `PushPowerButton()`  
  Play the power button push animation
- void `PushVolumeUpButton()`  
  Play the volume up button push animation
- void `PushVolumeDownButton()`  
  Play the volume down button push animation
- void `SetKnobRotation(float value)`  
  Sets the rotation of the knob, and sets the position of the slider based on a value between 0-1
- void `SetSlider(float degree)`  
  Slide the slider based on the degree of which the knob is turned.

### 4.93.1 Detailed Description

Definition at line 6 of file InteractableRadioModel.cs.
4.93.2 Member Function Documentation

4.93.2.1 ActivateRadio()

```csharp
void InteractableRadioModel.ActivateRadio()
```
Activates the radio mesh's and moves the radio from the offscreen position to the onscreen position.
Definition at line 100 of file InteractableRadioModel.cs.

4.93.2.2 DeactivateRadio()

```csharp
void InteractableRadioModel.DeactivateRadio()
```
Calls the deactivate coroutine
Definition at line 116 of file InteractableRadioModel.cs.

4.93.2.3 PushPowerButton()

```csharp
void InteractableRadioModel.PushPowerButton()
```
Play the power button push animation
Definition at line 147 of file InteractableRadioModel.cs.

4.93.2.4 PushVolumeDownButton()

```csharp
void InteractableRadioModel.PushVolumeDownButton()
```
Play the volume down button push animation
Definition at line 165 of file InteractableRadioModel.cs.

4.93.2.5 PushVolumeUpButton()

```csharp
void InteractableRadioModel.PushVolumeUpButton()
```
Play the volume up button push animation
Definition at line 156 of file InteractableRadioModel.cs.

4.93.2.6 SetKnobRotation()

```csharp
void InteractableRadioModel.SetKnobRotation(float value)
```
Sets the rotation of the knob, and sets the position of the slider based on a value between 0-1
Parameters

| value          | A number between 0-1 anything greater or less than will be clamped. |

Definition at line 193 of file InteractableRadioModel.cs.

4.93.2.7 SetSlider()

```csharp
void InteractableRadioModel.SetSlider ( float degree )
```

Slide the slider based on the degree of which the knob is turned.

Parameters

| degree |

Definition at line 208 of file InteractableRadioModel.cs.

4.93.2.8 SetUpRadioCanvas()

```csharp
void InteractableRadioModel.SetUpRadioCanvas ( )
```

Set up all the button start positions.

Definition at line 83 of file InteractableRadioModel.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Radio/InteractableRadioModel.cs
4.94 InteractableSetUp Class Reference

Inheritance diagram for InteractableSetUp:

Collaboration diagram for InteractableSetUp:

Public Attributes

- string ItemName
  
  The name of the item.

4.94.1 Detailed Description

Definition at line 4 of file InteractableSetUp.cs.
4.94.2 Member Data Documentation

4.94.2.1 ItemName

string InteractableSetUp.ItemName

The name of the item.

Definition at line 10 of file InteractableSetUp.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/InteractableSetUp.cs

4.95 Inventory Class Reference

Defines the inventory of items the player can access. Currently just considers all the gameobjects parented under item parent as part of the inventory. TODO: Enable this to be used for on hand and raft inventories. Add inventory size limit and ability to adjust the size. Make the inventory save strings of objects then use YAML file to define those items as needed.

Inheritance diagram for Inventory:
Collaboration diagram for Inventory:

```
ItemStack
+ Item
+ Id
+ Amount
+ ItemStack()
+ ItemStack()
+ UpdateStackAmountEvent()

#contents

Inventory
+ InventorySize
+ Inventory()
+ Inventory()
+ LoadInventory()
+ GetInventory()
+ UseItem()
+ GetStacks()
+ GetInventoryBaseItem()
+ GetItemsByType()
+ AddItem()
+ RemoveStack()
and 6 more...
```

### Public Member Functions

- **Inventory** (string name, int size)
  
  Initializes a new instance of the Inventory class.

- **Inventory** (string name, string inventoryFile, int size)
  
  Initializes a new instance of the Inventory class.

- **void LoadInventory ()**
  
  Loads the inventory with its contents based on the contents of the yaml file.

- **ItemStack [] GetInventory ()**
  
  Gets the contents of the inventory.

- **void UseItem (string item, int amount)**
  
  Uses the item. This means that the number of units in an item will decrease. If there are no units of the item left, then the item is removed from the inventory.

- **List< ItemStack > GetStacks (string name, int amount)**
  
  Gets stacks with the specified item name. Returns as many stacks as needed to fulfill the amount desired.

- **BaseItem GetInventoryBaseItem (string name)**
  
  Gets the BaseItem of an item in the inventory by its name.

- **List< string > GetItemsByType (List< string > types)**
  
  Gets the names of items given a type.
- **ItemStack AddItem (BaseItem newItem, int amount)**  
  Add item to inventory.
- **void RemoveStack (ItemStack stack)**  
  Removes the item from the inventory.
- **List<ItemStack> GetAllItemsWithTag (string itemTag)**  
  Gets all items in the inventory with a certain item tag.
- **void CombineStacks (ItemStack current, ItemStack target)**  
  Combines the stacks in an inventory. Adds the amount in the current stack to the target stack and removes the current stack, but only if adding to the target stack does not push it over the stack size.
- **int GetNextOpenSlot ()**  
  Gets the next open slot in the inventory.
- **void UpdateTypeAmount (List<string> types, int changedAmount)**  
  Updates the type amount.
- **bool CheckRecipePossible (Recipe recipe)**  
  Checks if recipe possible given items in the inventory.
- **bool CheckRequirementMet (Requirement requirement)**  
  Checks if the requirement is met.

### Protected Attributes

- **ItemStack [] contents**

### Properties

- **int InventorySize [get, set]**  
  Gets or sets the size of the inventory. Resizes the contents array as necessary. TODO: If inventory becomes smaller, the user should be able to choose what to discard. Also, should remove empty slots instead when shrinking.

#### 4.95.1 Detailed Description

Defines the inventory of items the player can access. Currently just considers all the gameobjects parented under item parent as part of the inventory. TODO: Enable this to be used for on hand and raft inventories. Add inventory size limit and ability to adjust the size. Make the inventory save strings of objects then use YAML file to define those items as needed.

Definition at line 12 of file Inventory.cs.

#### 4.95.2 Constructor & Destructor Documentation

##### 4.95.2.1 Inventory() [1/2]

```csharp
Inventory.Inventory (  
    string name,  
    int size  )
```

Initializes a new instance of the `Inventory` class.
4.95 Inventory Class Reference

Parameters

<table>
<thead>
<tr>
<th>name</th>
<th>Name.</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>Size.</td>
</tr>
</tbody>
</table>

Definition at line 34 of file Inventory.cs.

4.95.2.2 Inventory()

Inventory.Inventory (
    string name,
    string inventoryFile,
    int size )

Initializes a new instance of the Inventory class.

Parameters

<table>
<thead>
<tr>
<th>name</th>
<th>Name of inventory.</th>
</tr>
</thead>
<tbody>
<tr>
<td>inventoryFile</td>
<td>Inventory file.</td>
</tr>
</tbody>
</table>

Definition at line 52 of file Inventory.cs.

4.95.3 Member Function Documentation

4.95.3.1 AddItem()

ItemStack Inventory.AddItem (    BaseItem newItem,    int amount )

Add item to inventory.

Returns

The added item.

Parameters

<table>
<thead>
<tr>
<th>newItem</th>
<th>New item.</th>
</tr>
</thead>
<tbody>
<tr>
<td>amount</td>
<td>Amount.</td>
</tr>
</tbody>
</table>

Definition at line 245 of file Inventory.cs.
4.95.3.2  CheckRecipePossible()

bool Inventory.CheckRecipePossible (
    Recipe recipe
)

Checks if recipe possible given items in the inventory.

Returns

    true, if recipe possible was checked, false otherwise.

Parameters

    recipe  Recipe.

Definition at line 369 of file Inventory.cs.

4.95.3.3  CheckRequirementMet()

bool Inventory.CheckRequirementMet (
    Requirement requirement
)

Checks if the requirement is met.

Returns

    true, if requirement met was checked, false otherwise.

Parameters

    requirement  Requirement.

Definition at line 404 of file Inventory.cs.

4.95.3.4  CombineStacks()

void Inventory.CombineStacks (
    ItemStack current,
    ItemStack target
)

Combines the stacks in an inventory. Adds the amount in the current stack to the target stack and removes the
current stack, but only if adding to the target stack does not push it over the stack size.
4.95 Inventory Class Reference

Parameters

<table>
<thead>
<tr>
<th>current</th>
<th>Current</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>Target</td>
</tr>
</tbody>
</table>

Definition at line 305 of file Inventory.cs.

4.95.3.5 GetAllItemsWithTag()

```csharp
List<ItemStack> Inventory.GetAllItemsWithTag (string itemTag )
```

Gets all items in the inventory with a certain item tag.

Returns

All items with tag.

Parameters

| itemTag | Tag that contains desired items. |

Definition at line 281 of file Inventory.cs.

4.95.3.6 GetInventory()

```csharp
ItemStack [] Inventory.GetInventory ( )
```

Gets the contents of the inventory.

Returns

The inventory.

Definition at line 111 of file Inventory.cs.

4.95.3.7 GetInventoryBaseItem()

```csharp
BaseItem Inventory.GetInventoryBaseItem (string name )
```

Gets the BaseItem of an item in the inventory by its name

Returns

The inventory item.
Class Documentation

Parameters

| name  | Name. |

Definition at line 190 of file Inventory.cs.

4.95.3.8 GetItemsByType()

```csharp
List<string> Inventory.GetItemsByType (  
    List<string> types  )
```

Gets the names of items given a type.

Returns

The items by type.

Parameters

| type  | Type. |

Definition at line 214 of file Inventory.cs.

4.95.3.9 GetNextOpenSlot()

```csharp
int Inventory.GetNextOpenSlot ( )
```

Gets the next open slot in the inventory.

Returns

The next open slot.

Definition at line 318 of file Inventory.cs.

4.95.3.10 GetStacks()

```csharp
List<ItemStack> Inventory.GetStacks (  
    string name,  
    int amount )
```

Gets stacks with the specified item name. Returns as many stacks as needed to fulfill the amount desired.

Returns

The stacks.
Parameters

<table>
<thead>
<tr>
<th>name</th>
<th>Name of the item.</th>
</tr>
</thead>
<tbody>
<tr>
<td>amount</td>
<td>Amount of the item to get.</td>
</tr>
</tbody>
</table>

Definition at line 160 of file Inventory.cs.

4.95.3.11 LoadInventory()

```csharp
void Inventory.LoadInventory() {
}
```

Loads the inventory with its contents based on the contents of the yaml file.

Definition at line 72 of file Inventory.cs.

4.95.3.12 RemoveStack()

```csharp
void Inventory.RemoveStack(ItemStack stack) {
}
```

Removes the item from the inventory.

Parameters

| stack | Item to remove. |

Definition at line 258 of file Inventory.cs.

4.95.3.13 UpdateTypeAmount()

```csharp
void Inventory.UpdateTypeAmount(List<string> types, int changedAmount) {
}
```

Updates the type amount.

Parameters

<table>
<thead>
<tr>
<th>types</th>
<th>Types.</th>
</tr>
</thead>
<tbody>
<tr>
<td>changedAmount</td>
<td>Changed amount. Negative for removed amount, positive for added.</td>
</tr>
</tbody>
</table>

Definition at line 356 of file Inventory.cs.
4.95.3.14 UseItem()

void Inventory.UseItem (  
    string item,  
    int amount )

Uses the item. This means that the number of units in an item will decrease. If there are no units of the item left, then the item is removed from the inventory.

Parameters

<table>
<thead>
<tr>
<th>item</th>
<th>Name of item.</th>
</tr>
</thead>
<tbody>
<tr>
<td>amount</td>
<td>Amount of item to use.</td>
</tr>
</tbody>
</table>

Definition at line 122 of file Inventory.cs.

4.95.4 Member Data Documentation

4.95.4.1 contents

ItemStack[] Inventory.contents [protected]

Definition at line 23 of file Inventory.cs.

4.95.5 Property Documentation

4.95.5.1 InventorySize

int Inventory.InventorySize [get], [set]

Gets or sets the size of the inventory. Resizes the contents array as necessary. TODO: If inventory becomes smaller, the user should be able to choose what to discard. Also, should remove empty slots instead when shrinking.

The size of the inventory.

Definition at line 339 of file Inventory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/Inventory.cs
Inheritance diagram for `InventoryInteractable`:
Collaboration diagram for InventoryInteractable:

![Collaboration diagram]

Public Member Functions

- override void `SetUp()`

  Sets openInventory as an action that should fire off when `PerformAction()` is called.

Properties

- `Inventory AttachedInventory` [get, set]

  Gets or sets the attached inventory.

Additional Inherited Members

4.96.1 Detailed Description

Definition at line 4 of file InventoryInteractable.cs.

4.96.2 Member Function Documentation
4.96.2.1 SetUp()

```csharp
override void InventoryInteractable.SetUp() [virtual]
```

Sets openInventory as an action that should fire off when PerformAction is called.

Reimplemented from `InteractableObject`.

Definition at line 27 of file `InventoryInteractable.cs`.

4.96.3 Property Documentation

4.96.3.1 AttachedInventory

```csharp
Inventory InventoryInteractable.AttachedInventory [get], [set]
```

Gets or sets the attached inventory.

The attached inventory.

Definition at line 11 of file `InventoryInteractable.cs`.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/InventoryInteractable.cs

4.97 InventoryItemYAMLModel Class Reference

Collaboration diagram for `InventoryItemYAMLModel`:
Properties

- string StackId [get, set]
  
  Gets or sets the stack identifier.

- BaseItem Item [get, set]
  
  Gets or sets the base item.

- int ItemAmount [get, set]
  
  Gets or sets the amount of an item in the stack.

- List<ItemCategory> ItemCategories [get, set]
  
  Gets or sets the item categories attached to the base item.

4.97.1 Detailed Description

Definition at line 4 of file InventoryItemYAMLModel.cs.

4.97.2 Property Documentation

4.97.2.1 Item

BaseItem InventoryItemYAMLModel.Item [get], [set]

Gets or sets the base item.

The item.

Definition at line 21 of file InventoryItemYAMLModel.cs.

4.97.2.2 ItemAmount

int InventoryItemYAMLModel.ItemAmount [get], [set]

Gets or sets the amount of an item in the stack.

The item amount.

Definition at line 31 of file InventoryItemYAMLModel.cs.
4.97.2.3 ItemCategories

List<ItemCategory> InventoryItemYAMLModel.ItemCategories [get], [set]

Gets or sets the item categories attached to the base item.

The item categories.

Definition at line 41 of file InventoryItemYAMLModel.cs.

4.97.2.4 StackId

string InventoryItemYAMLModel.StackId [get], [set]

Gets or sets the stack identifier.

The stack identifier.

Definition at line 11 of file InventoryItemYAMLModel.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/InventoryItemYAMLModel.cs

4.98 InventoryUI Class Reference

Inheritance diagram for InventoryUI:
Collaboration diagram for InventoryUI:

```
ItemStack
+ Item
+ Id
+ Amount
+ ItemStack()
+ UpdateStackAmountEvent()

Inventory
+ InventorySize
+ Inventory()
+ Inventory()
+ LoadInventory()
+ GetInventory()
+ UseItem()
+ GetStacks()  
+ GetInventoryBaseItem()  
+ GetItemsByType()  
+ AddItem()
+ RemoveStack() and 6 more...

MonoBehaviour

InventoryUI
+ ItemsToDiscard
+ ItemSpriteManager
+ DisplayInventory()
+ RefreshInventoryPanel()
+ LoadNewInventory()
+ GetStackUI()

+ TargetInventory
```

Public Member Functions

- **void DisplayInventory ()**
  
  Displays the inventory.

- **void RefreshInventoryPanel ()**
  
  Refreshes the inventory display and updates the slots on the panel.

- **void LoadNewInventory (Inventory newInventory)**
  
  Loads a new inventory into the ui panel.

- **ItemStackUI GetStackUI (string id)**
  
  Gets the stack UI component given a stack id.
Public Attributes

- **Inventory TargetInventory**
  The target inventory that will be shown by the UI.

Properties

- **List ItemStack > ItemsToDiscard [get, set]**
  Gets or sets the items to discard.
- **SpriteManager ItemSpriteManager [get]**
  Gets the item sprite manager.

4.98.1 Detailed Description

Definition at line 7 of file InventoryUI.cs.

4.98.2 Member Function Documentation

4.98.2.1 DisplayInventory()

```csharp
void InventoryUI.DisplayInventory { }
```

Displays the inventory.

Definition at line 101 of file InventoryUI.cs.

4.98.2.2 GetStackUI()

```csharp
ItemStackUI InventoryUI.GetStackUI { string id }
```

Gets the stack UI component given a stack id.

Returns

The stack UI.

Parameters

- **id** Identifier.

Definition at line 268 of file InventoryUI.cs.
4.98.2.3 LoadNewInventory()

```csharp
void InventoryUI.LoadNewInventory (Inventory newInventory )
```

Loads a new inventory into the UI panel.

**Parameters**

- `newInventory` New inventory.

Definition at line 178 of file InventoryUI.cs.

4.98.2.4 RefreshInventoryPanel()

```csharp
void InventoryUI.RefreshInventoryPanel ( )
```

Refreshes the inventory display and updates the slots on the panel.

Definition at line 119 of file InventoryUI.cs.

4.98.3 Member Data Documentation

4.98.3.1 TargetInventory

```csharp
Inventory InventoryUI.TargetInventory
```

The target inventory that will be shown by the UI.

Definition at line 32 of file InventoryUI.cs.

4.98.4 Property Documentation

4.98.4.1 ItemSpriteManager

```csharp
SpriteManager InventoryUI.ItemSpriteManager [get]
```

Gets the item sprite manager.

The item sprite manager.

Definition at line 49 of file InventoryUI.cs.
4.98.4.2 ItemsToDiscard

List<ItemStack> InventoryUI.ItemsToDiscard [get], [set]

Gets or sets the items to discard.

The items to discard.

Definition at line 39 of file InventoryUI.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts.GUI/InventoryGUI/InventoryUI.cs

4.99 InventoryYAMLModel Class Reference

Collaboration diagram for InventoryYAMLModel:

![Collaboration Diagram](image)

Properties

- string **InventoryName** [get, set]
  
  Gets or sets the name of the inventory.

- List<InventoryItemYAMLModel> **Items** [get, set]
  
  Gets or sets the items in the inventory that are in InventoryYAMLModel form.

- int **InventorySize** [get, set]
  
  Gets or sets the size of the inventory.

4.99.1 Detailed Description

Definition at line 5 of file InventoryYAMLModel.cs.

4.99.2 Property Documentation
4.99.2.1 InventoryName

string InventoryYAMLModel.InventoryName [get], [set]

Gets or sets the name of the inventory.

The name of the inventory.

Definition at line 12 of file InventoryYAMLModel.cs.

4.99.2.2 InventorySize

int InventoryYAMLModel.InventorySize [get], [set]

Gets or sets the size of the inventory.

The size of the inventory.

Definition at line 32 of file InventoryYAMLModel.cs.

4.99.2.3 Items

List<InventoryItemYAMLModel> InventoryYAMLModel.Items [get], [set]

Gets or sets the items in the inventory that are in InventoryYAMLModel form.

The items.

Definition at line 22 of file InventoryYAMLModel.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/InventoryYAMLModel.cs
Inheritance diagram for InventoryYamlParser:
Collaboration diagram for InventoryYamlParser:

```
CraftingSystemSerializer
    + Filename
    # categoryNames
    # categoryTypes
    # uriPrefix
    + SetUpCategoryInformation()

InventoryYamlParser
    + InventoryYamlParser()
    + LoadInventories()
    + GetInventoryContents()
    + AddNewInventory()
    + SaveInventoriesToFile()
    + SaveInventory()
```

Public Member Functions

- **InventoryYamlParser** (string file)
  
  Awake this instance.

- **void** LoadInventories ()
  
  Loads the inventories from the yaml file.

- **List< InventoryItemYAMLModel >** GetInventoryContents (string inventoryID)
  
  Gets the inventory's contents given an inventory.

- **void** AddNewInventory (string inventoryID)
  
  Creates a new inventory.

- **void** SaveInventoriesToFile ()
  
  Serializes the contents of inventoryYamlData and updates the YAML file. TODO: Get it to actually write to a file, may require saving inventory data to player prefs

- **void** SaveInventory (Inventory inventory, string inventoryName)
  
  Saves the inventory passed to it to inventoryYamlData.

Additional Inherited Members

4.100.1 Detailed Description

Definition at line 12 of file InventoryYamlParser.cs.
4.100.2 Constructor & Destructor Documentation

4.100.2.1 InventoryYamlParser()

`InventoryYamlParser.InventoryYamlParser ( string file )`

Awake this instance.

Definition at line 19 of file InventoryYamlParser.cs.

4.100.3 Member Function Documentation

4.100.3.1 AddNewInventory()

```csharp
void InventoryYamlParser.AddNewInventory ( string inventoryID )
```

Creates a new inventory.

Parameters

- `inventoryID`: Inventory ID.

Definition at line 66 of file InventoryYamlParser.cs.

4.100.3.2 GetInventoryContents()

```csharp
List<InventoryItemYAMLModel> InventoryYamlParser.GetInventoryContents ( string inventoryID )
```

Gets the inventory's contents given an inventory.

Returns

The inventory contents.

Parameters

- `inventoryID`: Inventory ID.
4.100.3.3 LoadInventories()

```csharp
void InventoryYamlParser.LoadInventories()
```

Loads the inventories from the yaml file.

Definition at line 33 of file InventoryYamlParser.cs.

4.100.3.4 SaveInventoriesToFile()

```csharp
void InventoryYamlParser.SaveInventoriesToFile()
```

Serializes the contents of inventoryYamlData and updates the YAML file. TODO: Get it to actually write to a file, may require saving inventory data to player prefs

Definition at line 76 of file InventoryYamlParser.cs.

4.100.3.5 SaveInventory()

```csharp
void InventoryYamlParser.SaveInventory(
    Inventory inventory,
    string inventoryName)
```

Saves the inventory passed to it to inventoryYamlData.

**Parameters**

<table>
<thead>
<tr>
<th>inventory</th>
<th>Inventory.</th>
</tr>
</thead>
<tbody>
<tr>
<td>inventoryName</td>
<td>Inventory name.</td>
</tr>
</tbody>
</table>

Definition at line 99 of file InventoryYamlParser.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/InventoryYamlParser.cs

### 4.101 ItemAction Class Reference

Struct that defines an action being the name of the action and the UnityAction. The action may also have an id that specifies which subcategory of action it falls under.
Collaboration diagram for ItemAction:

```
<table>
<thead>
<tr>
<th>ItemAction</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ AssignedAction</td>
</tr>
<tr>
<td>+ ActionName</td>
</tr>
<tr>
<td>+ SubActions</td>
</tr>
<tr>
<td>+ ActionComplete</td>
</tr>
<tr>
<td>+ Conditions</td>
</tr>
<tr>
<td>+ TypeUsed</td>
</tr>
<tr>
<td>+ ItemAction()</td>
</tr>
</tbody>
</table>
```

Public Member Functions

- **ItemAction** (string name, UnityAction desiredAction)
  
  Initializes a new instance of the Action class.

Properties

- UnityAction **AssignedAction** [get, set]
  
  Gets or sets the assigned action to perform.

- string **ActionName** [get, set]
  
  Gets or sets the name of the action.

- List<ItemAction> **SubActions** [get, set]
  
  Gets or sets the sub actions of this action.

- bool **ActionComplete** [get, set]
  
  Gets or sets a value indicating whether this Action has been completed.

- List<ItemCondition> **Conditions** [get, set]
  
  Conditions for the action to be possible

- List<string> **TypeUsed** [get, set]
  
  What item types can be consumed by this action. Generally used for action that require use of another item to fulfill, like adding fuel.

4.101.1 Detailed Description

Struct that defines an action being the name of the action and the UnityAction. The action may also have an id that specifies which subcategory of action it falls under.

Definition at line 10 of file ItemAction.cs.

4.101.2 Constructor & Destructor Documentation
4.101.2.1 ItemAction()

ItemAction.ItemAction (  
    string name,  
    UnityAction desiredAction )

Initializes a new instance of the Action class.

Parameters

<table>
<thead>
<tr>
<th>name</th>
<th>Name.</th>
</tr>
</thead>
<tbody>
<tr>
<td>desiredAction</td>
<td>Action.</td>
</tr>
</tbody>
</table>

Definition at line 17 of file ItemAction.cs.

4.101.3 Property Documentation

4.101.3.1 ActionComplete

bool ItemAction.ActionComplete [get], [set]

Gets or sets a value indicating whether this Action has been completed.

true if action complete; otherwise, false.

Definition at line 63 of file ItemAction.cs.

4.101.3.2 ActionName

string ItemAction.ActionName [get], [set]

Gets or sets the name of the action.

The name of the action.

Definition at line 43 of file ItemAction.cs.

4.101.3.3 AssignedAction

UnityAction ItemAction.AssignedAction [get], [set]

Gets or sets the assigned action to perform.

The assigned action.

Definition at line 33 of file ItemAction.cs.
4.101.3.4 Conditions

List<ItemCondition> ItemAction.Conditions [get], [set]

Conditions for the action to be possible

The conditions.

Definition at line 73 of file ItemAction.cs.

4.101.3.5 SubActions

List<ItemAction> ItemAction.SubActions [get], [set]

Gets or sets the sub actions of this action.

The sub actions.

Definition at line 53 of file ItemAction.cs.

4.101.3.6 TypeUsed

List<string> ItemAction.TypeUsed [get], [set]

What item types can be consumed by this action. Generally used for action that require use of another item to fulfill, like adding fuel.

The types used.

Definition at line 83 of file ItemAction.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ItemAction.cs
4.102 ItemActionButtonUI Class Reference

Inheritance diagram for ItemActionButtonUI:

```
MonoBehaviour

ItemActionButtonUI
+ SubActions
+ SetAction()
+ StartAction()
+ PerformAction()
+ ShowSubActions()
```

Collaboration diagram for ItemActionButtonUI:

```
MonoBehaviour

ItemActionButtonUI
+ SubActions
+ SetAction()
+ StartAction()
+ PerformAction()
+ ShowSubActions()
```

Public Member Functions

- void SetAction (UnityAction actionToSet, string name)
Sets the action that will be completed when the button is clicked.

- void **StartAction** ()
  Prepares the action. If no subactions, prompts for a number to modify. Otherwise, displays up subactions.

- void **PerformAction** ()
  Performs the action. Refreshes the panel that displays the item's information.

- void **ShowSubActions** ()
  A special action separate from those that can be performed by items. This is used on Action Buttons that are representing a category of actions to perform. When clicked, it will populate the Subcategory Actions section of the item info panel and display the action buttons whose actionSubcategoryID is the same as this ActionButton's.

**Properties**

- List&lt;**ItemAction**&gt; **SubActions** [get, set]
  Gets or sets the sub actions.

### 4.102.1 Detailed Description

Definition at line 7 of file ItemActionButtonUI.cs.

### 4.102.2 Member Function Documentation

#### 4.102.2.1 PerformAction()"}

```csharp
void ItemActionButtonUI.PerformAction ( )
```

Performs the action. Refreshes the panel that displays the item's information.

Definition at line 49 of file ItemActionButtonUI.cs.

#### 4.102.2.2 SetAction()"}

```csharp
void ItemActionButtonUI.SetAction ( UnityAction actionToSet, string name )
```

Sets the action that will be completed when the button is clicked.

**Parameters**

| **actionToSet** | The action that will completed. |
| **name**        | Name of the action.             |
4.102.3 ShowSubActions()

void ItemActionButtonUI.ShowSubActions ()

A special action separate from those that can be performed by items. This is used on Action Buttons that are representing a category of actions to perform. When clicked, it will populate the Subcategory Actions section of the item info panel and display the action buttons whose actionSubcategoryID is the same as this ActionButton's.

Definition at line 79 of file ItemActionButtonUI.cs.

4.102.4 StartAction()

void ItemActionButtonUI.StartAction ()

Prepares the action. If no subactions, prompts for a number to modify. Otherwise, displays up subactions.

Definition at line 29 of file ItemActionButtonUI.cs.

4.102.3 Property Documentation

4.102.3.1 SubActions

List<ItemAction> ItemActionButtonUI.SubActions [get], [set]

Gets or sets the sub actions.

The sub actions.

Definition at line 62 of file ItemActionButtonUI.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/InventoryGUI/ItemActionButtonUI.cs
4.103 ItemAttributeUI Class Reference

Inheritance diagram for ItemAttributeUI:

Collaboration diagram for ItemAttributeUI:

Public Member Functions

- void `setAttributeName` (string name)
Sets the name of the attribute.

- void **setAttributeValue** (float value)
  
  Sets the attribute value.

- void **ShowHover** ()
  
  Shows the hover panel with the slider value.

- void **HideHover** ()
  
  Hides the hover panel with the slider value.

### Public Attributes

- Text **AttributeName**
- int **RangeValue**

  *This may be set by attributes directly, so public.*

#### 4.103.1 Detailed Description

Definition at line 5 of file ItemAttributeUI.cs.

#### 4.103.2 Member Function Documentation

##### 4.103.2.1 **HideHover()**

void ItemAttributeUI.HideHover ()

Hides the hover panel with the slider value.

Definition at line 85 of file ItemAttributeUI.cs.

##### 4.103.2.2 **setAttributeName()**

void ItemAttributeUI.setAttributeName (string name)

Sets the name of the attribute.

**Parameters**

| name | Name |

Definition at line 40 of file ItemAttributeUI.cs.
4.103.2.3 SetAttributeValue()

```csharp
void ItemAttributeUI.SetAttributeValue (float value)
```

Sets the attribute value.

Parameters

<table>
<thead>
<tr>
<th>value</th>
<th>Value</th>
</tr>
</thead>
</table>

Definition at line 49 of file ItemAttributeUI.cs.

4.103.2.4 ShowHover()

```csharp
void ItemAttributeUI.ShowHover ()
```

Shows the hover panel with the slider value.

Definition at line 77 of file ItemAttributeUI.cs.

4.103.3 Member Data Documentation

4.103.3.1 AttributeName

**Text** ItemAttributeUI.AttributeName

Definition at line 8 of file ItemAttributeUI.cs.

4.103.3.2 RangeValue

**int** ItemAttributeUI.RangeValue

This may be set by attributes directly, so public.

Definition at line 14 of file ItemAttributeUI.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/InventoryGUI/ItemAttributeUI.cs
Abstract class for the classes which contain suites of actions and attributes which can be used to define a category of items. Examples include solid, liquid, and plant.

Inheritance diagram for ItemCategory:
Collaboration diagram for ItemCategory:

```
CollectableItem
+ GetPossibleActions()

BaseItem
+ DirtyFlag
+ UpdateExistingFlag
+ RemovalFlag
+ DiscardFlag
+ TypeName
+ Types
+ FlavorText
+ InventorySprite
+ WorldModel
+ ActionModifiedSprites
+ ModifyingActionNames
+ ActionModifiedModels
+ Rarity
+ UpdateItemEvent()
+ BaseItem()
+ BaseItem()
+ BaseItem()
+ InitializeBaseItem()
+ SetUpBaseItem()
+ AddItemCategory()
+ GetItemAttributes()
+ GetPossibleActions()
+ GetNumberOfActionsCompleted()
+ #baseItem

ItemCategory
+ Attributes
+ Actions
+ SetBaseItem()
+ GetPossibleActions()
+ ReadyCategory()
+ GetDuplicate()
+ SetActionComplete()
+ GetAttribute()
+ #finishDuplication()
```

Public Member Functions

- void SetBaseItem (BaseItem bitem)
  
  *Sets the component that this is connected to. May either be another item category or a base item class.*

- override List<ItemAction> GetPossibleActions ()
  
  *Gets all the action that an item can perform.*

- virtual void ReadyCategory ()

Generated by Doxygen
Preps the category for use by loading attributes and actions into lists.

- virtual ItemCategory GetDuplicate ()
  
  Gets a copy of the ItemCategory.

- void SetActionComplete (string name)
  
  Sets the action as completed.

- Attribute GetAttribute (string name)
  
  Gets the attribute value of an attribute in the category by the attribute's name.

Protected Member Functions

- void finishDuplication (ItemCategory newCategory)
  
  Finishes the duplication of the category by copying over the Actions and Attributes.

Protected Attributes

- Baseliem baseitem
  
  base item which controls all item category components

Properties

- List<Attribute> Attributes [get, set]
  
  Gets or sets the attributes.

- List<ItemAction> Actions [get, set]
  
  Gets or sets the actions.

4.104.1 Detailed Description

Abstract class for the classes which contain suites of actions and attributes which can be used to define a category of items. Examples include solid, liquid, and plant.

Definition at line 9 of file ItemCategory.cs.

4.104.2 Member Function Documentation

4.104.2.1 finishDuplication()

void ItemCategory.finishDuplication (ItemCategory newCategory) [protected]

Finishes the duplication of the category by copying over the Actions and Attributes.

Parameters

- newCategory New category.
4.104.2.2 GetAttribute()

```csharp
public Attribute GetAttribute(string name)
```

Gets the attribute value of an attribute in the category by the attribute's name.

**Returns**

The attribute value.

**Parameters**

- `name`: Name.

Definition at line 125 of file ItemCategory.cs.

---

4.104.2.3 GetDuplicate()

```csharp
public ItemCategory GetDuplicate()
```

Gets a copy of the ItemCategory.

**Returns**

The duplicate.


Definition at line 91 of file ItemCategory.cs.

---

4.104.2.4 GetPossibleActions()

```csharp
public override List<ItemAction> GetPossibleActions()
```

Gets all the action that an item can perform.

**Returns**

The possible actions of an item in a Dictionary keyed by the action name.

Reimplemented from CollectableItem.

Definition at line 50 of file ItemCategory.cs.
4.104.2.5 ReadyCategory()

virtual void ItemCategory.ReadyCategory ( ) [virtual]

Preps the category for use by loading attributes and actions into lists.


Definition at line 83 of file ItemCategory.cs.

4.104.2.6 SetActionComplete()

void ItemCategory.SetActionComplete ( string name )

Sets the action as completed.

Parameters

| name | Name of the action. |

Definition at line 101 of file ItemCategory.cs.

4.104.2.7 SetBaseItem()

void ItemCategory.SetBaseItem ( BaseItem bItem )

Sets the component that this is connected to. May either be another item category or a base item class.

Parameters

| item | The item this is linked to. |
| baseItem | The base item component this item category is used by. |

Definition at line 41 of file ItemCategory.cs.

4.104.3 Member Data Documentation
4.104.3.1  baseItem

*BaseItem* ItemCategory.baseItem  [protected]

base item which controls all item category components

Definition at line 14 of file ItemCategory.cs.

4.104.4  Property Documentation

4.104.4.1  Actions

*List*< ItemAction> ItemCategory.Actions  [get], [set]

Gets or sets the actions.

The actions.

Definition at line 31 of file ItemCategory.cs.

4.104.4.2  Attributes

*List*< Attribute> ItemCategory.Attributes  [get], [set]

Gets or sets the attributes.

The attributes.

Definition at line 21 of file ItemCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ItemCategory.cs

4.105  ItemCondition Class Reference

Collaboration diagram for ItemCondition:

```
<table>
<thead>
<tr>
<th>ItemCondition</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ AttributeName</td>
</tr>
<tr>
<td>+ ThresholdValue</td>
</tr>
<tr>
<td>+ ConditionValue</td>
</tr>
<tr>
<td>+ ItemCondition()</td>
</tr>
<tr>
<td>+ ItemCondition()</td>
</tr>
<tr>
<td>+ CheckCondition()</td>
</tr>
<tr>
<td>+ CheckCondition()</td>
</tr>
</tbody>
</table>
```
Public Member Functions

- **ItemCondition** (string attribute, float threshold, BooleanOperator.BooleanOperatorDelegate boolOp)
  
  Initializes a new instance of the ItemCondition class.

- **ItemCondition** (float condition, float threshold, BooleanOperator.BooleanOperatorDelegate boolOp)
  
  Initializes a new instance of the ItemCondition class.

- **bool CheckCondition** (float value)
  
  Checks to see if the condition is met. Compares the value against the ThresholdValue according to the boolean operator.

- **bool CheckCondition** ()
  
  Checks to see if the condition is met. Compares the ConditionValue to the ThresholdValue.

Properties

- **string AttributeName** [get, set]
  
  Gets or sets the name of the attribute.

- **float ThresholdValue** [get, set]
  
  Gets or sets the threshold value.

- **float ConditionValue** [get, set]
  
  Gets or sets the condition value.

4.105.1 Detailed Description

Definition at line 1 of file ItemCondition.cs.

4.105.2 Constructor & Destructor Documentation

4.105.2.1 **ItemCondition()** [1/2]

ItemCondition.ItemCondition ( 
    string attribute, 
    float threshold, 
    BooleanOperator.BooleanOperatorDelegate boolOp )

Initializes a new instance of the ItemCondition class.

Parameters

<table>
<thead>
<tr>
<th>attribute</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>threshold</td>
<td>Threshold</td>
</tr>
<tr>
<td>boolOp</td>
<td>Bool op</td>
</tr>
</tbody>
</table>

Definition at line 11 of file ItemCondition.cs.
4.105.2.2 ItemCondition() [2/2]

ItemCondition.ItemCondition (  
    float condition,  
    float threshold,  
    BooleanOperator.BooleanOperatorDelegate boolOp )

Initializes a new instance of the ItemCondition class.

Parameters

<table>
<thead>
<tr>
<th>conditionVariable</th>
<th>Condition variable.</th>
</tr>
</thead>
<tbody>
<tr>
<td>threshold</td>
<td>Threshold.</td>
</tr>
<tr>
<td>boolOp</td>
<td>Bool op.</td>
</tr>
</tbody>
</table>

Definition at line 24 of file ItemCondition.cs.

4.105.3 Member Function Documentation

4.105.3.1 CheckCondition() [1/2]

bool ItemCondition.CheckCondition (  
    float value )

Checks to see if the condition is met. Compares the value against the ThresholdValue according to the boolean operator.

Returns

true, if condition was checked, false otherwise.

Parameters

| value | Value. |

Definition at line 66 of file ItemCondition.cs.

4.105.3.2 CheckCondition() [2/2]

bool ItemCondition.CheckCondition ( )

Checks to see if the condition is met. Compares the ConditionValue to the ThresholdValue.

Returns

true, if condition was checked, false otherwise.

Definition at line 75 of file ItemCondition.cs.
4.105.4 Property Documentation

4.105.4.1 AttributeName

string ItemCondition.AttributeName [get], [set]

Gets or sets the name of the attribute.

The name of the attribute.

Definition at line 36 of file ItemCondition.cs.

4.105.4.2 ConditionValue

float ItemCondition.ConditionValue [get], [set]

Gets or sets the condition value.

The condition value.

Definition at line 56 of file ItemCondition.cs.

4.105.4.3 ThresholdValue

float ItemCondition.ThresholdValue [get], [set]

Gets or sets the threshold value.

The threshold value.

Definition at line 46 of file ItemCondition.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ItemCondition.cs
4.106 ItemDiscarder Class Reference

Collaboration diagram for ItemDiscarder:

```
ItemDiscarder

+ DiscardItems()
```

Public Member Functions

- void `DiscardItems(List<ItemStack> itemsToDiscard)`
  
  Discards the items from the inventory and places them in the world.

4.106.1 Detailed Description

Definition at line 5 of file ItemDiscarder.cs.

4.106.2 Member Function Documentation

4.106.2.1 DiscardItems()

```csharp
void ItemDiscarder.DiscardItems (  
    List<ItemStack> itemsToDiscard  
)
```

Discards the items from the inventory and places them in the world.

Parameters

- `itemsToDiscard` Items to discard.

Definition at line 16 of file ItemDiscarder.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/ItemDiscarder.cs

Generated by Doxygen
4.107  ItemDistrictModel Class Reference

Collaboration diagram for ItemDistrictModel:

```
ItemDistrictModel
+ DistrictName
+ LandItems
+ WaterItems
```

Properties

- string DistrictName  [get, set]
  
  Gets or sets the name of the district.

- List<string> LandItems  [get, set]
  
  Gets or sets the items that can appear on land in the district.

- List<string> WaterItems  [get, set]
  
  Gets or sets items that can appear on water in the district.

4.107.1 Detailed Description

Definition at line 3 of file ItemDistrictModel.cs.

4.107.2 Property Documentation

4.107.2.1 DistrictName

string ItemDistrictModel.DistrictName  [get], [set]

Gets or sets the name of the district.

The name of the district.

Definition at line 10 of file ItemDistrictModel.cs.
4.107.2.2 LandItems

List&lt;string&gt; ItemDistrictModel.LandItems [get], [set]

Gets or sets the items that can appear on land in the district.

The items.

Definition at line 20 of file ItemDistrictModel.cs.

4.107.2.3 WaterItems

List&lt;string&gt; ItemDistrictModel.WaterItems [get], [set]

Gets or sets items that can appear on water in the district.

The water items.

Definition at line 30 of file ItemDistrictModel.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/ItemDistrictModel.cs

4.108 ItemFactory Class Reference

Factory that creates an base items given its name and complex crafted items given recipe and ingredients. Currently just a placeholder class only able to craft a fishing rod.

Collaboration diagram for ItemFactory:

```
ItemFactory
+ ItemDatabase
+ LandItemsByDistrict
+ WaterItemsByDistrict
+ LandDistrictItemRarityInfo
+ WaterDistrictItemRarityInfo

+ CraftItem()
+ ItemFactory()
+ Craft()
+ GetBaseItem()
+ GetBaseItem()
+ GetWeightedRandomItemIndex()
+ GetWeightedRandomBaseItem()
+ GetItemNamesByTag()
```
Public Member Functions

- delegate void CraftItem (List<BaseItem> ingredients)
  Temporary delegate function that handles crafting a specific object given a list of items
- ItemFactory ()
  Start this instance. Loading in the craftingList by YAML.
- void Craft (Recipe recipe, List<Ingredient> ingredients, Inventory targetInventory)
  Crafts the item based on the requirements of the recipe and the ingredients provided. The final stats of the resultant item is determined by the level that an item has achieved.
- BaseItem GetBaseItem (InventoryItemYAMLModel item)
  Gets the base item based on the item's InventoryYAMLInfo. This is generally used for uncrafted items. These items may have had their names changed after an action has been applied to them, as such, their unmodifiedName will need to be used to get their blueprints from the itemDatabase.
- BaseItem GetBaseItem (string itemName)
  Gets the base item by the item's name.
- int GetWeightedRandomItemIndex (string district, bool onWater)
  Gets the index of the weighted random item in a district.
- BaseItem GetWeightedRandomBaseItem (string district, bool onLand)
  Gets a weighted random base item in a district.
- List<string> GetItemNamesByTag (string desiredTag)
  Gets the item names by tag.

Properties

- Dictionary<string, BaseItem> ItemDatabase [get]
  Gets the item database.
- Dictionary<string, List<string>> LandItemsByDistrict [get]
  Gets the land items by district.
- Dictionary<string, List<string>> WaterItemsByDistrict [get]
  Gets the items that generates in the water by district.
- Dictionary<string, DistrictItemRarityConfiguration> LandDistrictItemRarityInfo [get]
  Gets the rarity info of items that appear on land by district.
- Dictionary<string, DistrictItemRarityConfiguration> WaterDistrictItemRarityInfo [get]
  Gets the rarity info of items that appear on water by district.

4.108.1 Detailed Description

Factory that creates an base items given its name and complex crafted items given recipe and ingredients. Currently just a placeholder class only able to craft a fishing rod.

Definition at line 10 of file ItemFactory.cs.

4.108.2 Constructor & Destructor Documentation
4.108.2.1 ItemFactory()

ItemFactory.ItemFactory ( )

Start this instance. Loading in the craftingList by YAML.

Definition at line 84 of file ItemFactory.cs.

4.108.3 Member Function Documentation

4.108.3.1 Craft()

void ItemFactory.Craft ( 
    Recipe recipe, 
    List< Ingredient > ingredients, 
    Inventory targetInventory )

Crafts the item based on the requirements of the recipe and the ingredients provided. The final stats of the resultant item is determined by the level that an item has achieved.

Different parts of the recipe affects different stats, which is specified by affectingItems. If the sum of the values from the affectingItems exceeds a crafting stat's threshold values for a level, then then the crafting stat is considered that level. For example, if the threshold for "flexibility" is 3 or a "Good" level, and "Excellent" is 6, and the affecting items are all of the "Rope" type. If the user has selected 5 "Rope" type items, each with flexibility 1, then resultant sum is 5. It exceeds the "Good" level, but not the "Excellent" level, so the the "flexibility" crafting stat is considered "Good".

If all the crafting stats are "Good" or above, then the resultant item will be the "Good" version of the item. If all are "Excellent", then the resultant item will be the "Excellent" version of the item. Otherwise, it is just the "Poor" version of the item that will be created.

Parameters

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recipe</td>
<td>Recipe for the item.</td>
</tr>
<tr>
<td>ingredients</td>
<td>Ingredients to be used. Required by CraftItem delegate functions.</td>
</tr>
</tbody>
</table>

Definition at line 129 of file ItemFactory.cs.

4.108.3.2 CraftItem()

delegate void ItemFactory.CraftItem ( 
    List< BaseItem > ingredients )

Temporary delegate function that handles crafting a specific object given a list of items
4.108.3.3  GetBaseItem() [1/2]

```csharp
BaseItem ItemFactory.GetBaseItem (InventoryItemYAMLModel item)
```

Gets the base item based on the item's InventoryYAMLInfo. This is generally used for uncrafted items. These items may have had their names changed after an action has been applied to them, as such, their unmodifiedName will need to be used to get their blueprints from the itemDatabase.

For example, "River Reed" is stored within the ItemListYAML file, but "River Reed Thread", which is the result of the "Weave Rope" action being applied to "River Reed" is not. Instead, the blueprint for "River Reed" is taken instead.

**Returns**

The base item.

**Parameters**

- **item** Item.

Definition at line 312 of file ItemFactory.cs.

4.108.3.4  GetBaseItem() [2/2]

```csharp
BaseItem ItemFactory.GetBaseItem (string itemName)
```

Gets the base item by the item's name.

**Returns**

The base item.

**Parameters**

- **itemName** Item name.

Definition at line 327 of file ItemFactory.cs.

4.108.3.5  GetItemNamesByTag()

```csharp
List<string> ItemFactory.GetItemNamesByTag (string desiredTag)
```

Gets the item names by tag.
Returns

The item names by tag.

Parameters

<table>
<thead>
<tr>
<th>desiredTag</th>
<th>Desired tag</th>
</tr>
</thead>
</table>

Definition at line 413 of file ItemFactory.cs.

4.108.3.6  GetWeightedRandomBaseItem()

```csharp
BaseItem ItemFactory.GetWeightedRandomBaseItem ( string district, bool onLand )
```

Gets a weighted random base item in a district.

Returns

The weighted random base item.

Parameters

<table>
<thead>
<tr>
<th>district</th>
<th>District</th>
</tr>
</thead>
<tbody>
<tr>
<td>onLand</td>
<td>Whether or not the item index is intended for items that only generate on land.</td>
</tr>
</tbody>
</table>

Definition at line 394 of file ItemFactory.cs.

4.108.3.7  GetWeightedRandomItemIndex()

```csharp
int ItemFactory.GetWeightedRandomItemIndex ( string district, bool onWater )
```

Gets the index of the weighted random item in a district.

Returns

The weighted random item index.

Parameters

<table>
<thead>
<tr>
<th>district</th>
<th>District</th>
</tr>
</thead>
<tbody>
<tr>
<td>onWater</td>
<td>Whether or not the item index is intended for items that only generate on water.</td>
</tr>
</tbody>
</table>
Definition at line 376 of file ItemFactory.cs.

### 4.108.4 Property Documentation

#### 4.108.4.1 ItemDatabase

`Dictionary<string, BaseItem> ItemFactory.ItemDatabase [get]`

Gets the item database.

The item database.

Definition at line 22 of file ItemFactory.cs.

#### 4.108.4.2 LandDistrictItemRarityInfo

`Dictionary<string, DistrictItemRarityConfiguration> ItemFactory.LandDistrictItemRarityInfo [get]`

Gets the rarity info of items that appear on land by district

The land district item rarity info.

Definition at line 52 of file ItemFactory.cs.

#### 4.108.4.3 LandItemsByDistrict

`Dictionary<string, List<string>> ItemFactory.LandItemsByDistrict [get]`

Gets the land items by district.

The items that generates on land by district.

Definition at line 32 of file ItemFactory.cs.

#### 4.108.4.4 WaterDistrictItemRarityInfo

`Dictionary<string, DistrictItemRarityConfiguration> ItemFactory.WaterDistrictItemRarityInfo [get]`

Gets the rarity info of items that appear on water by district

The water district item rarity info.

Definition at line 62 of file ItemFactory.cs.
4.108.4.5 WaterItemsByDistrict

Dictionary<string, List<string>> > ItemFactory.WaterItemsByDistrict [get]

Gets the items that generates in the water by district.

The water items by district.

Definition at line 42 of file ItemFactory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ItemFactory.cs

4.109 ItemGenerator Class Reference

Inheritance diagram for ItemGenerator:
Collaboration diagram for ItemGenerator:

![Collaboration Diagram]

Public Member Functions

- **List<float> GetItemExtents(List<GameObject> items)**
  Returns the extents of each gameobject. Used during sampling point generation to take into account the extents of the object when dealing with minDistance. Only needs the extents of the objects, so only half the size. Assuming that pivot is in center.

- **void SetSeed(int newSeed)**
  Sets the seed.

- **void AddTemplatesToItemPool()**
Protected Member Functions

- List<float> getRarityInformation (string district)
  
  Gets the rarity information for all items in a district.

- Bounds calculateBounds (GameObject item)
  
  Calculates the bounds of a given gameobject.

Protected Attributes

- ItemPoolManager poolManager
- Dictionary<string, DistrictItemConfiguration> districtItemInfo
  
  The item templates used to create the objects in the world

4.109.1 Detailed Description

Definition at line 5 of file ItemGenerator.cs.

4.109.2 Member Function Documentation

4.109.2.1 AddTemplatesToItemPool()

void ItemGenerator.AddTemplatesToItemPool ()

Definition at line 101 of file ItemGenerator.cs.

4.109.2.2 calculateBounds()

Bounds ItemGenerator.calculateBounds (GameObject item) [protected]

Calculates the bounds of a given gameobject.

Returns

The bounds.

Parameters

item GameObject to check.

the first mesh is already a part of combinedMesh, so go to the second
4.109.2.3 GetItemExtents()

List<float> ItemGenerator.GetItemExtents ( List<GameObject> items )

Returns the extents of each gameobject. Used during sampling point generation to take into account the extents of the object when dealing with minDistance. Only needs the extents of the objects, so only half the size. Assuming that pivot is in center.

Returns
The extents of the items.

Parameters
- items: Items.

4.109.2.4 getRarityInformation()

List<float> ItemGenerator.getRarityInformation ( string district ) [protected]

Gets the rarity information for all items in a district.

Returns
The rarity information.

Parameters
- district: District that the items belong to.

4.109.2.5 SetSeed()

void ItemGenerator.SetSeed ( int newSeed )

Sets the seed.
4.110 ItemPlacementSamplePoint Class Reference

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>newSeed</td>
<td>Seed.</td>
</tr>
</tbody>
</table>

Definition at line 96 of file ItemGenerator.cs.

4.109.3 Member Data Documentation

4.109.3.1 districtItemInfo

Dictionary&lt; string, DistrictItemConfiguration &gt; ItemGenerator.districtItemInfo [protected]

The item templates used to create the objects in the world

Definition at line 14 of file ItemGenerator.cs.

4.109.3.2 poolManager

ItemPoolManager ItemGenerator.poolManager [protected]

Definition at line 9 of file ItemGenerator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/ItemGenerator.cs

4.110 ItemPlacementSamplePoint Class Reference

Collaboration diagram for ItemPlacementSamplePoint:
Public Types

- enum `PointType` { `PointType.ITEM`, `PointType.DOOR`, `PointType.SHELTER` }

Properties

- `Vector2 LocalTargetSurfaceLocation` [get, set]
  
  Gets or sets the local position on the target surface.

- `float MinDistance` [get, set]
  
  Gets or sets the minimum distance away that another object needs to be.

- `Tuple<int, int> GridPoint` [get, set]
  
  Gets or sets the grid point.

- `Vector3 WorldSpaceLocation` [get, set]
  
  Gets or sets the world space location.

- `int ItemIndex` [get, set]
  
  Gets or sets the index of the item.

- `float Size` [get, set]
  
  Gets or sets the size.

- `string District` [get, set]
  
  Gets or sets the district.

- `PointType Type` [get, set]
  
  Gets or sets the type of object that will be placed at this location. Either Door, Shelter, or Item.

4.110.1 Detailed Description

Definition at line 4 of file ItemPlacementSamplePoint.cs.

4.110.2 Member Enumeration Documentation

4.110.2.1 PointType

enum `ItemPlacementSamplePoint.PointType` [strong]

Enumerator

- `ITEM`
- `DOOR`
- `SHELTER`

Definition at line 6 of file ItemPlacementSamplePoint.cs.

4.110.3 Property Documentation
4.110.3.1 District

string ItemPlacementSamplePoint.District [get], [set]

Gets or sets the district.

The district.

Definition at line 73 of file ItemPlacementSamplePoint.cs.

4.110.3.2 GridPoint

Tuple<int, int> ItemPlacementSamplePoint.GridPoint [get], [set]

Gets or sets the grid point.

The grid point.

Definition at line 33 of file ItemPlacementSamplePoint.cs.

4.110.3.3 ItemIndex

int ItemPlacementSamplePoint.ItemIndex [get], [set]

Gets or sets the index of the item.

The index of the item.

Definition at line 53 of file ItemPlacementSamplePoint.cs.

4.110.3.4 LocalTargetSurfaceLocation

Vector2 ItemPlacementSamplePoint.LocalTargetSurfaceLocation [get], [set]

Gets or sets the local position on the target surface.

The local target surface location.

Definition at line 13 of file ItemPlacementSamplePoint.cs.
4.110.3.5 MinDistance

float ItemPlacementSamplePoint.MinDistance [get], [set]

Gets or sets the minimum distance away that another object needs to be.

The minimum distance.

Definition at line 23 of file ItemPlacementSamplePoint.cs.

4.110.3.6 Size

float ItemPlacementSamplePoint.Size [get], [set]

Gets or sets the size.

The size.

Definition at line 63 of file ItemPlacementSamplePoint.cs.

4.110.3.7 Type

PointType ItemPlacementSamplePoint.Type [get], [set]

Gets or sets the type of object that will be placed at this location. Either Door, Shelter, or Item.

The type.

Definition at line 83 of file ItemPlacementSamplePoint.cs.

4.110.3.8 WorldSpaceLocation

Vector3 ItemPlacementSamplePoint.WorldSpaceLocation [get], [set]

Gets or sets the world space location.

The world space location.

Definition at line 43 of file ItemPlacementSamplePoint.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/ItemPlacementSamplePoint.cs
4.111 ItemPoolInfo Class Reference

Collaboration diagram for ItemPoolInfo:

```
<table>
<thead>
<tr>
<th>ItemPoolInfo</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Locations</td>
</tr>
<tr>
<td>+ Activated</td>
</tr>
<tr>
<td>+ ItemNames</td>
</tr>
<tr>
<td>+ Items</td>
</tr>
<tr>
<td>+ ItemPoolInfo()</td>
</tr>
<tr>
<td>+ RemoveItemInfo()</td>
</tr>
<tr>
<td>+ AddItemInfo()</td>
</tr>
</tbody>
</table>
```

**Public Member Functions**

- **ItemPoolInfo** (Vector3 location, string itemName, bool activated)
  
  *Initializes a new instance of the ItemPoolInfo class.*

- **void RemoveItemInfo** (int index)
  
  *Removes the item info from the pool info.*

- **void AddItemInfo** (GameObject item)
  
  *Adds the item info from the pool information.*

**Properties**

- **List< Vector3 >** Locations [get, set]
  
  *Gets or sets the locations of items in the cell.*

- **bool Activated** [get, set]
  
  *Gets or sets a value indicating whether this ItemPoolInfo is activated.*

- **List< string >** ItemNames [get, set]
  
  *Gets or sets the item names in this cell.*

- **List< GameObject >** Items [get, set]
  
  *Gets or sets the gameobjects used to represent items in this cell.*

4.111.1 Detailed Description

Definition at line 5 of file ItemPoolInfo.cs.

4.111.2 Constructor & Destructor Documentation

Generated by Doxygen
### 4.111.2.1 ItemPoolInfo()

```csharp
ItemPoolInfo.ItemPoolInfo (  
  Vector3 location,  
  string itemName,  
  bool activated  
)
```

Initializes a new instance of the `ItemPoolInfo` class.

**Parameters**

<table>
<thead>
<tr>
<th>location</th>
<th>Location.</th>
</tr>
</thead>
<tbody>
<tr>
<td>itemName</td>
<td>Item name.</td>
</tr>
<tr>
<td>activated</td>
<td>If set to <code>true</code> activated.</td>
</tr>
</tbody>
</table>

Definition at line 53 of file `ItemPoolInfo.cs`.

### 4.111.3 Member Function Documentation

#### 4.111.3.1 AddItemInfo()

```csharp
void ItemPoolInfo.AddItemInfo (  
  GameObject item  
)
```

Adds the item info from the pool information.

**Parameters**

| item | Item. |

Definition at line 79 of file `ItemPoolInfo.cs`.

#### 4.111.3.2 RemoveItemInfo()

```csharp
void ItemPoolInfo.RemoveItemInfo (  
  int index  
)
```

Removes the item info from the pool info.

**Parameters**

| index | Index. |

Definition at line 68 of file `ItemPoolInfo.cs`. 

---

Generated by Doxygen
4.111.4 Property Documentation

4.111.4.1 Activated

bool ItemPoolInfo.Activated [get], [set]

Gets or sets a value indicating whether this ItemPoolInfo is activated.

true if activated; otherwise, false.

Definition at line 22 of file ItemPoolInfo.cs.

4.111.4.2 ItemNames

List<string> ItemPoolInfo.ItemNames [get], [set]

Gets or sets the item names in this cell.

The item names.

Definition at line 32 of file ItemPoolInfo.cs.

4.111.4.3 Items

List<GameObject> ItemPoolInfo.Items [get], [set]

Gets or sets the gameobjects used to represent items in this cell.

The items.

Definition at line 42 of file ItemPoolInfo.cs.

4.111.4.4 Locations

List<Vector3> ItemPoolInfo.Locations [get], [set]

Gets or sets the locations of items in the cell.

The locations.

Definition at line 12 of file ItemPoolInfo.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/ItemPoolInfo.cs

Generated by Doxygen
4.112 ItemPoolManager Class Reference

Inheritance diagram for ItemPoolManager:

```
MonoBehaviour

# grid
+ SetUpItemPoolManager()
+ StartManagingPool()
+ AddToGrid()
+ RemoveItemFromWorld()
+ AddItemFromWorld()
+ AddItemToWorld()
+ GetItemPool()
# PointToGrid()
```
Public Member Functions

- void **SetUpItemPoolManager** (float width, float depth, Vector3 center)
  
  *Sets up item pool manager.*

- IEnumerator **StartManagingPool** ()
  
  *Populates the starting item pool and activates the items in range. Does so one frame after start.*

- void **AddToGrid** (Vector3 location, string item, bool activated)
  
  *Adds an item’s information to grid.*

- void **RemoveItemFromWorld** (GameObject item)
  
  *Removes an item from the world. Put back into item pool.*

- void **AddItemFromWorld** (GameObject item)
  
  *Adds an item that has just appeared in the world and was not previously accounted for.*

- void **AddItemToPool** (string itemName, GameObject item)
  
  *Adds GameObject to the pool.*

- ItemPoolInfo [ , ] **GetItemPool** ()
  
  *Gets the item pool.*

Protected Member Functions

- **Tuple**< int, int > **PointToGrid** (Vector2 location)
  
  *Get the coord access points to grid from a worldspace point. (0, 0) is at the lower left corner of the city. All values returned should be positive!*
Protected Attributes

- `ItemPoolInfo[,] grid`
  
  The grid that contains information about what space is occupied in the city.

4.112.1 Detailed Description

Definition at line 5 of file `ItemPoolManager.cs`.

4.112.2 Member Function Documentation

4.112.2.1 `AddItemFromWorld()`

```csharp
void ItemPoolManager.AddItemFromWorld (
  GameObject item )
```

Adds an item that has just appeared in the world and was not previously accounted for.

Parameters

- `item` Item.

Definition at line 371 of file `ItemPoolManager.cs`.

4.112.2.2 `AddItemToPool()`

```csharp
void ItemPoolManager.AddItemToPool (
  string itemName,
  GameObject item )
```

Adds gameobject to the pool.

Parameters

- `itemName` Item name.
- `item` Item.

Definition at line 426 of file `ItemPoolManager.cs`. 
4.112.2.3 AddToGrid()

```csharp
void ItemPoolManager.AddToGrid(
    Vector3 location,
    string item,
    bool activated)
```

Adds an item's information to grid.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>Location.</td>
</tr>
<tr>
<td>item</td>
<td>Item.</td>
</tr>
<tr>
<td>activated</td>
<td>If set to true activated.</td>
</tr>
</tbody>
</table>

Definition at line 123 of file ItemPoolManager.cs.

4.112.2.4 GetItemPool()

```csharp
ItemPoolInfo[,] ItemPoolManager.GetItemPool();
```

Gets the item pool.

**Returns**

The item pool.

Definition at line 441 of file ItemPoolManager.cs.

4.112.2.5 PointToGrid()

```csharp
Tuple<int, int> ItemPoolManager.PointToGrid(
    Vector2 location); [protected]
```

Get the coord access points to grid from a worldspace point. (0, 0) is at the lower left corner of the city. All values returned should be positive!

**Returns**

The coord points used to access the ItemPoolInfo for that cell in the grid.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>Location of the point.</td>
</tr>
</tbody>
</table>
Definition at line 151 of file ItemPoolManager.cs.

4.112.2.6 RemoveItemFromWorld()

```csharp
void ItemPoolManager.RemoveItemFromWorld (GameObject item)
```

Removes an item from the world. Put back into item pool.

Parameters

- `item`: Item.

Definition at line 346 of file ItemPoolManager.cs.

4.112.2.7 SetUpItemPoolManager()

```csharp
void ItemPoolManager.SetUpItemPoolManager (float width, float depth, Vector3 center)
```

Sets up item pool manager.

Parameters

- `width`: Width.
- `depth`: Depth.
- `center`: Center.

Definition at line 66 of file ItemPoolManager.cs.

4.112.2.8 StartManagingPool()

```csharp
IEnumerator ItemPoolManager.StartManagingPool()
```

Populates the starting item pool and activates the items in range. Does so one frame after start.

Returns

- The managing pool.

Definition at line 106 of file ItemPoolManager.cs.
4.112.3 Member Data Documentation

4.112.3.1 grid

`ItemPoolInfo[,] ItemPoolManager.grid [protected]`

The grid that contains information about what space is occupied in the city.

Definition at line 38 of file ItemPoolManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/ItemPoolManager.cs

4.113 ItemRarity Class Reference

Collaboration diagram for ItemRarity:

```
ItemRarity
+ Common
+ Uncommon
+ Rare
+ Legendary
+ GetRarity()
```

Static Public Member Functions

- static float `GetRarity (string rarity)`
  
  *Gets the rarity value given a rarity string value.*

Static Public Attributes

- static string `Common = "common"
- static string `Uncommon = "uncommon"
- static string `Rare = "rare"
- static string `Legendary = "legendary`
4.113.1 Detailed Description

Definition at line 3 of file ItemRarity.cs.

4.113.2 Member Function Documentation

4.113.2.1 GetRarity()

static float ItemRarity.GetRarity (  
    string rarity ) [static]

Gets the rarity value given a rarity string value.

Returns
    The rarity.

Parameters
    rarity  Rarity.

Definition at line 18 of file ItemRarity.cs.

4.113.3 Member Data Documentation

4.113.3.1 Common

string ItemRarity.Common = "common" [static]

Definition at line 5 of file ItemRarity.cs.

4.113.3.2 Legendary

string ItemRarity.Legendary = "legendary" [static]

Definition at line 8 of file ItemRarity.cs.
4.113.3 Rare

```csharp
string ItemRarity.Rare = "rare" [static]
```

Definition at line 7 of file ItemRarity.cs.

4.113.4 Uncommon

```csharp
string ItemRarity.Uncommon = "uncommon" [static]
```

Definition at line 6 of file ItemRarity.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ItemRarity.cs

### 4.114 ItemSerializer Class Reference

Inheritance diagram for ItemSerializer:

![Inheritance diagram for ItemSerializer](image)
Collaboration diagram for ItemSerializer:

```
<table>
<thead>
<tr>
<th>CraftingSystemSerializer</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Filename</td>
</tr>
<tr>
<td># categoryNames</td>
</tr>
<tr>
<td># categoryTypes</td>
</tr>
<tr>
<td># urlPrefix</td>
</tr>
<tr>
<td>+ SetUpCategoryInformation()</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ItemSerializer</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ ItemSerializer()</td>
</tr>
<tr>
<td>+ DeserializeItemInformation()</td>
</tr>
<tr>
<td>+ DeserializeDistrictItemData()</td>
</tr>
</tbody>
</table>
```

Public Member Functions

- **ItemSerializer** (string itemListFile, string districtItemFile)
  
  *Initializes a new instance of the ItemSerializer class.*

- **Dictionary**< string, BaseItem > **DeserializeItemInformation**()
  
  *Deserializes the items that can be created from the text file.*

- **void** **DeserializeDistrictItemData** (ref Dictionary< string, List< string >> landDistrictStorage, ref Dictionary< string, List< string >> waterDistrictStorage)
  
  *Deserializes the district item data.*

Additional Inherited Members

4.114.1 Detailed Description

Definition at line 14 of file ItemSerializer.cs.

4.114.2 Constructor & Destructor Documentation

4.114.2.1 ItemSerializer()

```
ItemSerializer.ItemSerializer (   
    string itemListFile,   
    string districtItemFile )
```

*Initializes a new instance of the ItemSerializer class.*
Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>itemListFile</td>
<td>File that contains all items.</td>
</tr>
<tr>
<td>districtItemFile</td>
<td>File that contains what items appear in each district.</td>
</tr>
</tbody>
</table>

Definition at line 23 of file ItemSerializer.cs.

### 4.114.3 Member Function Documentation

#### 4.114.3.1 DeserializedistrictItemData()

```csharp
void ItemSerializer.DeserializeDistrictItemData (  
    ref Dictionary<string, List<string>> landDistrictStorage,  
    ref Dictionary<string, List<string>> waterDistrictStorage )
```

Deserializes the district item data.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>landDistrictStorage</td>
<td>Dictionary where information about items appearing on land should appear.</td>
</tr>
<tr>
<td>waterDistrictStorage</td>
<td>Dictionary where information about items that can appear floating in water should appear.</td>
</tr>
</tbody>
</table>

Definition at line 78 of file ItemSerializer.cs.

#### 4.114.3.2 DeserializedItemInformation()

```csharp
Dictionary<string, BaseItem> ItemSerializer.DeserializeItemInformation ( )
```

Deserializes the items that can be created from the text file.

Returns

The item information.

Definition at line 38 of file ItemSerializer.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/ItemSerializer.cs
4.115 ItemSpawner Class Reference

Inheritance diagram for ItemSpawner:

```
MonoBehaviour

InteractableObject
+ Show
+ Text
  # setupComplete
+ SetUp()
+ PerformAction()
+ SetAction()

ItemSpawner
+ SpawnRadius
+ SpawnAfterActions
+ SpawnWithoutInteraction
+ MaxSpawnNumber
+ MinSpawnNumber
+ PosterPositions
+ PosterRotationModMax
+ SetUpSpawner()
```
Public Member Functions

- void SetUpSpawner (float size, string districtName)
  
  Sets up spawner. Sets the action as the interactable object action, and gets the size of the item around which items should be spawned.

Public Attributes

- float SpawnRadius
- bool SpawnAfterActions
- bool SpawnWithoutInteraction
- int MaxSpawnNumber
- int MinSpawnNumber

Properties

- Transform [] PosterPositions [get]
- float PosterRotationModMax [get]
4.115.1 Detailed Description

Definition at line 7 of file ItemSpawner.cs.

4.115.2 Member Function Documentation

4.115.2.1 SetUpSpawner()

```csharp
void ItemSpawner.SetUpSpawner (  
    float size,  
    string districtName )
```

Sets up spawner. Sets the action as the interactable object action, and gets the size of the item around which items should be spawned.

**Parameters**

| `centralItem` | The bounds of the gameobject this script is attached to. |

Definition at line 108 of file ItemSpawner.cs.

4.115.3 Member Data Documentation

4.115.3.1 MaxSpawnNumber

```csharp
int ItemSpawner.MaxSpawnNumber
```

Definition at line 27 of file ItemSpawner.cs.

4.115.3.2 MinSpawnNumber

```csharp
int ItemSpawner.MinSpawnNumber
```

Definition at line 30 of file ItemSpawner.cs.
4.115.3.3 SpawnAfterActions

bool ItemSpawner.SpawnAfterActions

Definition at line 21 of file ItemSpawner.cs.

4.115.3.4 SpawnRadius

float ItemSpawner.SpawnRadius

Definition at line 18 of file ItemSpawner.cs.

4.115.3.5 SpawnWithoutInteraction

bool ItemSpawner.SpawnWithoutInteraction

Definition at line 24 of file ItemSpawner.cs.

4.115.4 Property Documentation

4.115.4.1 PosterPositions

Transform [] ItemSpawner.PosterPositions [get]

Definition at line 73 of file ItemSpawner.cs.

4.115.4.2 PosterRotationModMax

float ItemSpawner.PosterRotationModMax [get]

Definition at line 83 of file ItemSpawner.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/ItemSpawner.cs
4.116  ItemStack Class Reference

Collaboration diagram for ItemStack:

```
+ Item
+ Id
+ Amount

+ ItemStack()
+ ItemStack()
+ UpdateStackAmountEvent()
```

Public Member Functions

- `ItemStack (BaseItem item, int stackAmount, string identifier)`
  Initializes a new instance of the Stack class.
- `ItemStack ()`
  Initializes a new instance of the Stack class.
- `delegate void UpdateStackAmountEvent (int amount)`
  delegate function that takes in an amount

Properties

- `BaseItem Item [get, set]`
  Gets or sets the item.
- `string Id [get, set]`
  Gets or sets the identifier.
- `int Amount [get, set]`
  Gets or sets the amount. When amount is changed, fires the UpdateStackAmount event

Events

- `UpdateStackAmountEvent UpdateStackAmount`
  event that can be subscribed to by function of UpdateStackAmountEvent format

4.116.1  Detailed Description

Definition at line 3 of file ItemStack.cs.
4.116.2 Constructor & Destructor Documentation

4.116.2.1 ItemStack() [1/2]

ItemStack.ItemStack (  
    BaseItem item,  
    int stackAmount,  
    string identifier )

Initializes a new instance of the Stack class.

Parameters

<table>
<thead>
<tr>
<th>item</th>
<th>Item.</th>
</tr>
</thead>
<tbody>
<tr>
<td>stackAmount</td>
<td>Stack amount.</td>
</tr>
<tr>
<td>identifier</td>
<td>Identifier.</td>
</tr>
</tbody>
</table>

Definition at line 16 of file ItemStack.cs.

4.116.2.2 ItemStack() [2/2]

ItemStack.ItemStack ( )

Initializes a new instance of the Stack class.

Definition at line 26 of file ItemStack.cs.

4.116.3 Member Function Documentation

4.116.3.1 UpdateStackAmountEvent()

delegate void ItemStack.UpdateStackAmountEvent (  
    int amount )

delegate function that takes in an amount

4.116.4 Property Documentation
4.116.4.1 Amount

```csharp
int ItemStack.Amount [get], [set]
```

Gets or sets the amount. When amount is changed, fires the UpdateStackAmount event

The amount.

Definition at line 55 of file `ItemStack.cs`.

4.116.4.2 Id

```csharp
string ItemStack.Id [get], [set]
```

Gets or sets the identifier.

The identifier.

Definition at line 45 of file `ItemStack.cs`.

4.116.4.3 Item

```csharp
BaseItem ItemStack.Item [get], [set]
```

Gets or sets the item.

The item.

Definition at line 35 of file `ItemStack.cs`.

4.116.5 Event Documentation

4.116.5.1 UpdateStackAmount

```csharp
UpdateStackAmountEvent ItemStack.UpdateStackAmount
```

event that can be subscribed to by function of `UpdateStackAmountEvent` format

Definition at line 83 of file `ItemStack.cs`.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ItemStack.cs
Item stack detail panel behavior.

Inheritance diagram for ItemStackDetailPanelBehavior:

```
MonoBehaviour

+ Attributes
+ PossibleActions
+ PossibleSubActions

+ SetSelectedItem()
+ ClearAttributesAndActions()
+ SetItemData()
+ UpdateSelectedAmount()
+ ClosePanel()
+ DisplayAttributesAndAction()
+ RefreshItemPanel()
+ RefreshItemActions()
+ CreateSubAction()
+ ClearSubActionPanel()
```
Collaboration diagram for ItemStackDetailPanelBehavior:

```
MonoBehaviour

+ Attributes
+ PossibleActions
+ PossibleSubActions

+ SetSelectedItem()
+ ClearAttributesAndActions()
+ SetItemData()
+ UpdateSelectedAmount()
+ ClosePanel()
+ DisplayAttributesAndAction()
+ RefreshItemPanel()
+ RefreshItemActions()
+ CreateSubAction()
+ ClearSubActionPanel()
```

Public Member Functions

- **void** `SetSelectedItem` (GameObject selectedItem)
  
  Sets the selected item.

- **void** `ClearAttributesAndActions` ()
  
  Clears the attributes and actions.

- **void** `SetItemData` (GameObject item)
  
  Sets the item data.

- **void** `UpdateSelectedAmount` (int amountToModify)
  
  Updates the selected amount.

- **void** `ClosePanel` ()
  
  Closes the panel.

- **void** `DisplayAttributesAndAction` ()
  
  Displays the attributes and action.

- **void** `RefreshItemPanel` ()
  
  Refreshes the item panel.

- **void** `RefreshItemActions` ()
  
  Refreshes the item actions.

- **void** `CreateSubAction` (List<ItemAction> actions)
  
  Creates the sub actions for a given category of action.

- **void** `ClearSubActionPanel` ()
  
  Clears the subcategory action panel.
Properties

- List<ItemAttributeUI> Attributes [get]
  Gets the attributes.

- List<ItemActionButtonUI> PossibleActions [get]
  Gets the possible actions.

- Dictionary<string, List<ItemActionButtonUI>> PossibleSubActions [get]
  Gets the possible sub actions.

4.117.1 Detailed Description

Item stack detail panel behavior.

Definition at line 11 of file ItemStackDetailPanelBehavior.cs.

4.117.2 Member Function Documentation

4.117.2.1 ClearAttributesAndActions()

void ItemStackDetailPanelBehavior.ClearAttributesAndActions ( )

Clears the attributes and actions.

Definition at line 115 of file ItemStackDetailPanelBehavior.cs.

4.117.2.2 ClearSubActionPanel()

void ItemStackDetailPanelBehavior.ClearSubActionPanel ( )

Clears the subcategory action panel.

Definition at line 339 of file ItemStackDetailPanelBehavior.cs.

4.117.2.3 ClosePanel()

void ItemStackDetailPanelBehavior.ClosePanel ( )

Closes the panel.

Definition at line 169 of file ItemStackDetailPanelBehavior.cs.

4.117.2.4 CreateSubAction()

void ItemStackDetailPanelBehavior.CreateSubAction ( 
    List<ItemAction> actions )

Creates the sub actions for a given category of action.
Parameters

| id   | Identifier. |

Definition at line 321 of file ItemStackDetailPanelBehavior.cs.

4.117.2.5 DisplayAttributesAndAction()

```csharp
void ItemStackDetailPanelBehavior.DisplayAttributesAndAction ( )
```

Displays the attributes and action.

Definition at line 185 of file ItemStackDetailPanelBehavior.cs.

4.117.2.6 RefreshItemActions()

```csharp
void ItemStackDetailPanelBehavior.RefreshItemActions ( )
```

Refreshs the item actions.

Definition at line 264 of file ItemStackDetailPanelBehavior.cs.

4.117.2.7 RefreshItemPanel()

```csharp
void ItemStackDetailPanelBehavior.RefreshItemPanel ( )
```

Refreshs the item panel.

Definition at line 242 of file ItemStackDetailPanelBehavior.cs.

4.117.2.8 SetItemData()

```csharp
void ItemStackDetailPanelBehavior.SetItemData ( GameObject item )
```

Sets the item data.

Parameters

| item | Item. |
4.117.9 SetSelectedItem()

```csharp
void ItemStackDetailPanelBehavior.SetSelectedItem (GameObject selectedItem)
```
Sets the selected item.

Parameters

- **selectedItem** - Selected item.

Definition at line 101 of file ItemStackDetailPanelBehavior.cs.

4.117.10 UpdateSelectedAmount()

```csharp
void ItemStackDetailPanelBehavior.UpdateSelectedAmount (int amountToModify)
```
Updates the selected amount.

Parameters

- **amountToModify** - Amount to modify.

Definition at line 161 of file ItemStackDetailPanelBehavior.cs.

4.117.3 Property Documentation

4.117.3.1 Attributes

```csharp
List<ItemAttributeUI> ItemStackDetailPanelBehavior.Attributes [get]
```
Gets the attributes.

The attributes.

Definition at line 58 of file ItemStackDetailPanelBehavior.cs.
4.117.3.2 PossibleActions

List<ItemActionButtonUI> ItemStackDetailPanelBehavior.PossibleActions [get]

Gets the possible actions.

The possible actions.

Definition at line 68 of file ItemStackDetailPanelBehavior.cs.

4.117.3.3 PossibleSubActions

Dictionary<string, List<ItemActionButtonUI>> ItemStackDetailPanelBehavior.PossibleSubActions [get]

Gets the possible sub actions.

The possible sub actions.

Definition at line 78 of file ItemStackDetailPanelBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/InventoryGUI/ItemStackDetailPanelBehavior.cs
Inheritance diagram for ItemStackUI:
Class Documentation

Collaboration diagram for ItemStackUI:

![Collaboration Diagram](image)

Public Member Functions

- **void** SetUpInventoryItem (ItemStack baseStack)
  
  Subscribes to name change event

- **void** Unsubscribe (ItemStack baseStack)
  
  Unsubscribe this instance from all event subscriptions. Must be called before destroying this structure! Unable to use destructor due to significant lag.

- **void** RefreshInventoryItem (ItemStack baseStack)
  
  Refreshs the inventory item with new information from stack

- **void** HandleItemNameTextChangeEvent (BaseItem item)
  
  Function that will be used to handle changes to the item's stats

- **void** HandleItemAmountTextChangeEvent (int newAmount)
  
  Function that will be used to handle changes to the item stack's amount

- **void** HandleItemIconChangeEvent (BaseItem item)
  
  Function that will be used to handle changes to the item's inventory sprite.

- **string** GetItemName ()

Generated by Doxygen
4.118 ItemStackUI Class Reference

Gets the name of the item.

- List< Attribute > GetItemAttributes ()
  Gets the item's attributes.
- List< ItemAction > GetPossibleActions ()
  Gets the possible actions that can be applied to the item.
- void PreserveOriginal (int numToModify)
  Creates a copy of the item and sets that as the targetItem. All actions will be applied to the targetItem.
- void UpdateTargetStack (int numToModify)
  Updates the target stack with the proper amount.
- void CheckForModification ()
  Checks to see if the targetItem has been modified. If so, then a new item has been created and needs to be added to
  the inventory. Otherwise, merge the duplicate item back into the original. The targetItem is the original item again.
- ItemStack GetStack ()
  Gets the base item.
- int GetMaxAmount ()
  Gets the max amount usable in this stack. If preserve original has already been called, then the original stack's
  value will be used. Otherwise, the target stack's value will be used.
- void SetHoverPanelActive (bool active)
  Sets the hover panel active or inactive.

Public Attributes

- UnityAction Action
- GameObject HoverPanel
- Text itemName
- Text ItemAmount
- Image InventorySprite

4.118.1 Detailed Description

Definition at line 7 of file ItemStackUI.cs.

4.118.2 Member Function Documentation

4.118.2.1 CheckForModification()

void ItemStackUI.CheckForModification ()

Checks to see if the targetItem has been modified. If so, then a new item has been created and needs to be added to
the inventory. Otherwise, merge the duplicate item back into the original. The targetItem is the original item again.

Definition at line 189 of file ItemStackUI.cs.
4.118.2.2 GetItemAttributes()

`List<Attribute> ItemStackUI.GetItemAttributes();`

Gets the item's attributes.

Returns

The item's attributes.

Definition at line 144 of file ItemStackUI.cs.

4.118.2.3 GetItemName()

`string ItemStackUI.GetItemName();`

Gets the name of the item.

Returns

The item name.

Definition at line 135 of file ItemStackUI.cs.

4.118.2.4 GetMaxAmount()

`int ItemStackUI.GetMaxAmount();`

Gets the max amount usable in this stack. If preserve original has already been called, then the original stack's value will be used. Otherwise, the target stack's value will be used.

Returns

The max amount.

Definition at line 264 of file ItemStackUI.cs.

4.118.2.5 GetPossibleActions()

`List<ItemAction> ItemStackUI.GetPossibleActions();`

Gets the possible actions that can be applied to the item.

Returns

The possible actions.

Definition at line 153 of file ItemStackUI.cs.
4.118.2.6 GetStack()

```csharp
ItemStack ItemStackUI.GetStack()
```

Gets the base item.

Returns

The base item.

Definition at line 253 of file ItemStackUI.cs.

4.118.2.7 HandleItemAmountTextChangeEvent()

```csharp
void ItemStackUI.HandleItemAmountTextChangeEvent (int newAmount)
```

Function that will be used to handle changes to the item stack's amount

Parameters

- `stack` Stack.

Definition at line 107 of file ItemStackUI.cs.

4.118.2.8 HandleItemIconChangeEvent()

```csharp
void ItemStackUI.HandleItemIconChangeEvent (BaseItem item)
```

Function that will be used to handle changes to the item's inventory sprite.

Parameters

- `item` Item.

Definition at line 116 of file ItemStackUI.cs.

4.118.2.9 HandleItemNameTextChangeEvent()

```csharp
void ItemStackUI.HandleItemNameTextChangeEvent (BaseItem item)
```

Function that will be used to handle changes to the item's stats
Parameters

| *item* | Item |

Definition at line 98 of file ItemStackUI.cs.

4.118.2.10  PreserveOriginal()

```csharp
void ItemStackUI.PreserveOriginal ( int numToModify )
```

Creates a copy of the item and sets that as the targetItem. All actions will be applied to the targetItem.

Definition at line 162 of file ItemStackUI.cs.

4.118.2.11  RefreshInventoryItem()

```csharp
void ItemStackUI.RefreshInventoryItem ( ItemStack baseStack )
```

Refreshes the inventory item with new information from stack

Parameters

| *baseStack* | Base stack |

Definition at line 76 of file ItemStackUI.cs.

4.118.2.12  SetHoverPanelActive()

```csharp
void ItemStackUI.SetHoverPanelActive ( bool active )
```

Sets the hover panel active or inactive.

Parameters

| *active* | If set to true active |

Definition at line 278 of file ItemStackUI.cs.
4.118.2.13  SetUpInventoryItem()

void ItemStackUI.SetUpInventoryItem (  
   ItemStack baseStack  )

Subscribes to name change event

Definition at line 34 of file ItemStackUI.cs.

4.118.2.14  Unsubscribe()

void ItemStackUI.Unsubscribe (  
   ItemStack baseStack  )

Unsubscribe this instance from all event subscriptions. Must be called before destroying this structure! Unable to use destructor due to significant lag.

Definition at line 65 of file ItemStackUI.cs.

4.118.2.15  UpdateTargetStack()

void ItemStackUI.UpdateTargetStack (  
   int numToModify  )

Updates the target stack with the proper amount.

Parameters

| numToModify | Number to modify. |

Definition at line 176 of file ItemStackUI.cs.

4.118.3  Member Data Documentation

4.118.3.1  Action

UnityAction ItemStackUI.Action

Definition at line 9 of file ItemStackUI.cs.
4.118.3.2  HoverPanel

GameObject ItemStackUI.HoverPanel
Definition at line 10 of file ItemStackUI.cs.

4.118.3.3  InventorySprite

Image ItemStackUI.InventorySprite
Definition at line 13 of file ItemStackUI.cs.

4.118.3.4  ItemAmount

Text ItemStackUI.ItemAmount
Definition at line 12 of file ItemStackUI.cs.

4.118.3.5  ItemName

Text ItemStackUI.ItemName
Definition at line 11 of file ItemStackUI.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/InventoryGUI/ItemStackUI.cs

4.119  ItemTypes Class Reference

Collaboration diagram for ItemTypes:

<table>
<thead>
<tr>
<th>ItemTypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Rope</td>
</tr>
<tr>
<td>+ Hook</td>
</tr>
<tr>
<td>+ Sharp</td>
</tr>
<tr>
<td>+ Rod</td>
</tr>
<tr>
<td>+ Container</td>
</tr>
<tr>
<td>+ Edible</td>
</tr>
<tr>
<td>+ Equipable</td>
</tr>
<tr>
<td>+ Igniter</td>
</tr>
<tr>
<td>+ Fuel</td>
</tr>
<tr>
<td>+ FloatableObject</td>
</tr>
<tr>
<td>and 6 more...</td>
</tr>
</tbody>
</table>
Static Public Attributes

- static string **Rope** = "rope"
  Rope like objects.
- static string **Hook** = "hook"
  Hooked objects.
- static string **Sharp** = "sharp"
  Hard, sharp objects.
- static string **Rod** = "rod"
  Long, thing objects.
- static string **Container** = "container"
  Objects that can hold other objects.
- static string **Edible** = "edible"
  Edible objects.
- static string **Equipable** = "equipable"
  Objects that can be equipped.
- static string **Igniter** = "igniter"
  Objects that can start a fire.
- static string **Fuel** = "fuel"
  Objects that can fuel a fire.
- static string **FloatableObject** = "floatable object"
  Objects that can float.
- static string **AdvancedTool** = "advanced tool"
  Objects that is the most advanced tools.
- static string **Medicinal** = "medicinal"
  Objects that can heal.
- static string **Blade** = "blade"
  Advanced sharp objects that are tools.
- static string **Cloth** = "cloth"
  Objects that are cloth.
- static string **BaseSolid** = "base solid"
  Basic solid objects that can be used as raw materials for carving, as base materials, etc. Like wooden blocks that can be carved into handles or idols.
- static string [][] **Types** = new string[]{Rope, Hook, Sharp, Rod, Container, Edible, Equipable, Igniter, Fuel, AdvancedTool, FloatableObject, Medicinal, Blade, Cloth, BaseSolid}

### 4.119.1 Detailed Description

Definition at line 1 of file ItemTypes.cs.

### 4.119.2 Member Data Documentation

#### 4.119.2.1 AdvancedTool

```csharp
string ItemTypes.AdvancedTool = "advanced tool" [static]
```

Objects that is the most advanced tools.

Definition at line 56 of file ItemTypes.cs.
4.119.2.2 BaseSolid

string ItemTypes.BaseSolid = "base solid" [static]

Basic solid objects that can be used as raw materials for carving, as base materials, etc. Like wooden blocks that can be carved into handles or idols.

Definition at line 77 of file ItemTypes.cs.

4.119.2.3 Blade

string ItemTypes.Blade = "blade" [static]

Advanced sharp objects that are tools.

Definition at line 66 of file ItemTypes.cs.

4.119.2.4 Cloth

string ItemTypes.Cloth = "cloth" [static]

Objects that are cloth.

Definition at line 71 of file ItemTypes.cs.

4.119.2.5 Container

string ItemTypes.Container = "container" [static]

Objects that can hold other objects.

Definition at line 26 of file ItemTypes.cs.

4.119.2.6 Edible

string ItemTypes.Edible = "edible" [static]

Edible objects.

Definition at line 31 of file ItemTypes.cs.
4.119.2.7 Equipable

```csharp
string ItemTypes.Equipable = "equipable" [static]
```

Objects that can be equipped.

Definition at line 36 of file ItemTypes.cs.

4.119.2.8 FloatableObject

```csharp
string ItemTypes.FloatableObject = "floatable object" [static]
```

Objects that can float.

Definition at line 51 of file ItemTypes.cs.

4.119.2.9 Fuel

```csharp
string ItemTypes.Fuel = "fuel" [static]
```

Objects that can fuel a fire.

Definition at line 46 of file ItemTypes.cs.

4.119.2.10 Hook

```csharp
string ItemTypes.Hook = "hook" [static]
```

Hooked objects.

Definition at line 11 of file ItemTypes.cs.

4.119.2.11 Igniter

```csharp
string ItemTypes.Igniter = "igniter" [static]
```

Objects that can start a fire.

Definition at line 41 of file ItemTypes.cs.
4.119.2.12 Medicinal

string ItemTypes.Medicinal = "medicinal" [static]

Objects that can heal.
Definition at line 61 of file ItemTypes.cs.

4.119.2.13 Rod

string ItemTypes.Rod = "rod" [static]

Long, thing objects.
Definition at line 21 of file ItemTypes.cs.

4.119.2.14 Rope

string ItemTypes.Rope = "rope" [static]

Rope like objects.
Definition at line 6 of file ItemTypes.cs.

4.119.2.15 Sharp

string ItemTypes.Sharp = "sharp" [static]

Hard, sharp objects.
Definition at line 16 of file ItemTypes.cs.

4.119.2.16 Types

string [] ItemTypes.Types = new string[]{Rope, Hook, Sharp, Rod, Container, Edible, Equipable, Igniter, Fuel, AdvancedTool, FloatableObject, Medicinal, Blade, Cloth, BaseSolid} [static]

Definition at line 79 of file ItemTypes.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ItemTypes.cs
4.120 ItemYAMLMap Class Reference

Collaboration diagram for ItemYAMLMap:

Properties

- string **itemName** [get, set]
  
  Gets or sets the name of the item.

- **BaseItem** [get, set]
  
  Gets or sets the base item.

- List&lt; **ItemCategory** &gt; **ItemCategories** [get, set]
  
  Gets or sets the categories attached to the base item.

4.120.1 Detailed Description

Definition at line 4 of file ItemYAMLMap.cs.

4.120.2 Property Documentation

4.120.2.1 BaseItem

**BaseItem** ItemYAMLMap.BaseItem [get], [set]

Gets or sets the base item.

The base item.

Definition at line 21 of file ItemYAMLMap.cs.
4.120.2.2 ItemCategories

List<ItemCategory> ItemYAMLMap.ItemCategories [get], [set]

Gets or sets the categories attached to the base item.

The item categories.

Definition at line 31 of file ItemYAMLMap.cs.

4.120.2.3 ItemName

string ItemYAMLMap.ItemName [get], [set]

Gets or sets the name of the item.

The name of the item.

Definition at line 11 of file ItemYAMLMap.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/ItemYAMLMap.cs
4.121  JellyFishSpawner Class Reference

Inheritance diagram for JellyFishSpawner:
Additional Inherited Members

4.121.1 Detailed Description

Definition at line 4 of file JellyFishSpawner.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Creatures/JellyFishSpawner.cs
Public Member Functions

- `IEnumerator SetKey(GameObject keyButtonPressed)`

  *Sets the key.*
• void OnKeyButtonClick ()
  Sets the button labels.
• void OnCloseClick ()
  Saves user input key codes and closes key code configuration panel.

4.122.1 Detailed Description

Definition at line 6 of file KeyCodeConfigMenuBehavior.cs.

4.122.2 Member Function Documentation

4.122.2.1 OnCloseClick()

void KeyCodeConfigMenuBehavior.OnCloseClick ( )

Saves user input key codes and closes key code configuration panel.

Definition at line 269 of file KeyCodeConfigMenuBehavior.cs.

4.122.2.2 OnKeyButtonClick()

void KeyCodeConfigMenuBehavior.OnKeyButtonClick ( )

Sets the button labels.

Definition at line 258 of file KeyCodeConfigMenuBehavior.cs.

4.122.2.3 SetKey()

IEnumerator KeyCodeConfigMenuBehavior.SetKey ( GameObject keyButtonPressed )

Sets the key.

Returns
  The key.

Parameters

  keyButtonPressed [Key button pressed.]
4.123 LandMovement Class Reference

Definition at line 160 of file KeyCodeConfigMenuBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/MainMenuScene/KeyCodeConfigMenuBehavior.cs

## 4.123 LandMovement Class Reference

Inheritance diagram for LandMovement:

![Inheritance Diagram](image)
Collaboration diagram for LandMovement:

```
MonoBehaviour

Movement
+ Speed
+ CurrentFallDamage
+ RigidBody
+ AccumulatedFallDamage
+ climbHeight
+ Idle()
+ Move()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ OnStateEnter()
+ OnStateExit()
+ GetRaycastHeight()

LandMovement
+ Move()
+ Idle()
+ Jump()
+ Climb()
+ JumpForce()
+ JumpLand()
+ GetClimbHeight()
+ GetRaycastHeight()
+ OnStateEnter()
+ OnStateExit()
```

**Public Member Functions**

- `override void Move(Vector3 direction, bool sprinting, Animator playerAnimator)`
  
  *Walks the player in the specified direction.*

- `override void Idle(Animator playerAnimator)`

  *Plays the idle animation*

- `override void Jump(Animator playerAnimator)`

  *The player aesthetic jumps*

- `override void Climb(Animator playerAnimator)`
The player's climb animation plays

- void **JumpForce** ()
  The player's rigidbody gets the jump force applied. Called via the animator.
- void **JumpLand** ()
  The sets jumping back to false. Called via the animator.
- override float **GetClimbHeight** ()
  The height the player can climb while in this movement state
- override float **GetRaycastHeight** ()
  The height of the climbing raycast while in this movement state
- override void **OnStateEnter** ()
  Called when the player enters the state.
- override void **OnStateExit** ()
  Called when the player exits the state.

Additional Inherited Members

4.123.1 Detailed Description

Definition at line 7 of file LandMovement.cs.

4.123.2 Member Function Documentation

4.123.2.1 Climb()

```csharp
override void LandMovement.Climb ( 
  Animator playerAnimator ) [virtual]
```

The player's climb animation plays

Parameters

- **playerAnimator** The player's animator

Implements **Movement**.
Definition at line 178 of file LandMovement.cs.

4.123.2.2 GetClimbHeight()

```csharp
override float LandMovement.GetClimbHeight ( ) [virtual]
```

The height the player can climb while in this movement state

Implements **Movement**.
Definition at line 203 of file LandMovement.cs.
4.123.2.3 GetRaycastHeight()

```csharp
override float LandMovement.GetRaycastHeight() [virtual]
```

The height of the climbing raycast while in this movement state

Implements Movement.

Definition at line 211 of file LandMovement.cs.

4.123.2.4 Idle()

```csharp
override void LandMovement.Idle(Animator playerAnimator) [virtual]
```

Plays the idle animation

Parameters

| playerAnimator | The player's animator |

Implements Movement.

Definition at line 157 of file LandMovement.cs.

4.123.2.5 Jump()

```csharp
override void LandMovement.Jump(Animator playerAnimator) [virtual]
```

The player aesthetic jumps

Parameters

| playerAnimator | The player's animator |

Implements Movement.

Definition at line 167 of file LandMovement.cs.

4.123.2.6 JumpForce()

```csharp
void LandMovement.JumpForce()
```

Generated by Doxygen
The player’s rigidbody gets the jump force applied. Called via the animator.

Definition at line 187 of file LandMovement.cs.

4.123.2.7 JumpLand()

```csharp
void LandMovement.JumpLand()
```

The sets jumping back to false. Called via the animator.

Definition at line 195 of file LandMovement.cs.

4.123.2.8 Move()

```csharp
override void LandMovement.Move(
    Vector3 direction,
    bool sprinting,
    Animator playerAnimator) [virtual]
```

Walks the player in the specified direction.

Parameters

- **direction**: The direction to walk the player.

Implements `Movement`.

Definition at line 96 of file LandMovement.cs.

4.123.2.9 OnStateEnter()

```csharp
override void LandMovement.OnStateEnter() [virtual]
```

Called when the player enters the state.

Implements `Movement`.

Definition at line 219 of file LandMovement.cs.
4.123.2.10 OnStateExit()

```csharp
override void LandMovement.OnStateExit () [virtual]
```

Called when the player exits the state.

Implements Movement.

Definition at line 227 of file LandMovement.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/LandMovement.cs
Inheritance diagram for LightCategory:

```
LightCategory
| + MaxFuel
| + Brightness
| + CurrentFuelLevel
| + BurnRate
| + GetDuplicate()
| + ReadyCategory()
| + AddFuel()
| + CalculateRemainingFuel()

EquipableCategory
| + Equiped
| # equipActionName
| # unequipActionName
| # equippedAttributeName
| + GetDuplicate()
| + ReadyCategory()
| + Equip()
| + UnEquip()

ItemCategory
| + Attributes
| + Actions
| # baseItem
| + SetBaseItem()
| + GetPossibleActions()
| + ReadyCategory()
| + GetDuplicate()
| + SetActionComplete()
| + GetAttribute()
| # finishDuplication()

CollectableItem
| + GetPossibleActions()

Generated by Doxygen
```
Collaboration diagram for LightCategory:

Public Member Functions

- **override** ItemCategory GetDuplicate ()
  
  *Gets a copy of the ItemCategory.*

- **override** void ReadyCategory ()
  
  *Preps the category for use by loading attributes and actions into lists.*

- void AddFuel (float fuel)
4.124 LightCategory Class Reference

```csharp
Add the fuel.

- float CalculateRemainingFuel()

  Calculates remaining fuel int the fire. This is not an action that can be done in the inventory! This is only used in the world.

Properties

- float MaxFuel [get, set]
  Gets or sets the max fuel.
- float Brightness [get, set]
  Gets or sets the brightness.
- float CurrentFuelLevel [get, set]
  Gets or sets the current fuel level.
- float BurnRate [get, set]
  Gets or sets the burn rate.

Additional Inherited Members

4.124.1 Detailed Description

Definition at line 6 of file LightCategory.cs.

4.124.2 Member Function Documentation

4.124.2.1 AddFuel()

```csharp
void LightCategory.AddFuel (  
    float fuel  )
```

Add the fuel.

Parameters

燃料 [Fuel]

Definition at line 112 of file LightCategory.cs.

4.124.2.2 CalculateRemainingFuel()

```csharp
float LightCategory.CalculateRemainingFuel (  )
```

Calculates remaining fuel int the fire. This is not an action that can be done in the inventory! This is only used in the world.
Returns

The fire.

Definition at line 122 of file LightCategory.cs.

4.124.2.3 GetDuplicate()

override ItemCategory LightCategory.GetDuplicate ( ) [virtual]

Gets a copy of the ItemCategory.

Returns

The duplicate.

Reimplemented from EquipableCategory.

Definition at line 60 of file LightCategory.cs.

4.124.2.4 ReadyCategory()

override void LightCategory.ReadyCategory ( ) [virtual]

Preps the category for use by loading attributes and actions into lists.

Reimplemented from EquipableCategory.

Definition at line 86 of file LightCategory.cs.

4.124.3 Property Documentation

4.124.3.1 Brightness

float LightCategory.Brightness [get], [set]

Gets or sets the brightness.

The brightness.

Definition at line 23 of file LightCategory.cs.
4.124.3.2 BurnRate

float LightCategory.BurnRate [get], [set]

Gets or sets the burn rate.

The burn rate.

Definition at line 43 of file LightCategory.cs.

4.124.3.3 CurrentFuelLevel

float LightCategory.CurrentFuelLevel [get], [set]

Gets or sets the current fuel level.

The current fuel level.

Definition at line 33 of file LightCategory.cs.

4.124.3.4 MaxFuel

float LightCategory.MaxFuel [get], [set]

Gets or sets the max fuel.

The max fuel.

Definition at line 13 of file LightCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/LightCategory.cs

Generated by Doxygen
4.125 LightItem Class Reference

Inheritance diagram for LightItem:

```
MonoBehaviour

  Tool
  + ToolName
  + InUse
  # attachJoint
  # toolName
  # unequipActName

  + Use()
  + Equip()
  + Unequip()
  + SetUpTool()

  LightItem
  + On

  + SetUpTool()
  + Equip()
  + Unequip()
  + Use()
  + AddFuel()
  + ToggleOn()
```
Collaboration diagram for LightItem:

```
MonoBehaviour

Tool
+ ToolName
+ InUse
# attachJoint
# toolName
# unequipActName
+ Use()
+ Equip()
+ Unequip()
+ SetUpTool()

LightItem
+ On
+ SetUpTool()
+ Equip()
+ Unequip()
+ Use()
+ AddFuel()
+ ToggleOn()
```

Public Member Functions

- override void `SetUpTool(BaseItem itemForTool)`
  
  *Sets up the tool so that it is linked to the proper item in the inventory.*

- override void `Equip()`
  
  *Equip this instance.*

- override void `Unequip()`
  
  *Unequip this instance.*

- override void `Use()`
  
  *Use this instance.*

- void `AddFuel()`
  
  *Adds fuel to the light item.*

- void `ToggleOn()`
  
  *Switches the lantern from on to off and vice versa.*
Properties

- bool On [get, set]

  Gets or sets a value indicating whether the light is on.

Additional Inherited Members

4.125.1 Detailed Description

Definition at line 6 of file LightItem.cs.

4.125.2 Member Function Documentation

4.125.2.1 AddFuel()

    void LightItem.AddFuel()

  Adds fuel to the light item.

  Definition at line 120 of file LightItem.cs.

4.125.2.2 Equip()

    override void LightItem.Equip()

  Equip this instance.

  Implements Tool.

  Definition at line 92 of file LightItem.cs.

4.125.2.3 SetUpTool()

    override void LightItem.SetUpTool(BaseItem itemForTool)

  Sets up the tool so that it is linked to the proper item in the inventory.

Parameters

- **ItemForTool** Item for tool.
Reimplemented from Tool.

Definition at line 44 of file LightItem.cs.

4.125.2.4 ToggleOn()

```csharp
void LightItem.ToggleOn();
```

Switches the lantern from on to off and vice versa.

Definition at line 132 of file LightItem.cs.

4.125.2.5 Unequip()

```csharp
override void LightItem.Unequip();
```

Unequip this instance.

Implements Tool.

Definition at line 101 of file LightItem.cs.

4.125.2.6 Use()

```csharp
override void LightItem.Use();
```

Use this instance.

Implements Tool.

Definition at line 112 of file LightItem.cs.

4.125.3 Property Documentation

4.125.3.1 On

```csharp
bool LightItem.On {get}; {set}
```

Gets or sets a value indicating whether the light it on.

true if on; otherwise, false.

Definition at line 74 of file LightItem.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/LightItem.cs

Generated by Doxygen
4.126  Lightning Class Reference

Inheritance diagram for Lightning:

```
MonoBehaviour
   |
   V
Lightning
   |
   V

MonoBehaviour
   |
   V
Lightning
   |
   V
```

Collaboration diagram for Lightning:

```
MonoBehaviour
   |
   V
Lightning
   |
   V
```

4.126.1  Detailed Description

Definition at line 5 of file Lightning.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/Lightning.cs
4.127 LightningEffect Class Reference

Inheritance diagram for LightningEffect:

Collaboration diagram for LightningEffect:

4.127.1 Detailed Description

Definition at line 4 of file LightningEffect.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/Effects/LightningEffect.cs
4.128  LoadingScreen Class Reference

Inheritance diagram for LoadingScreen:

```
MonoBehaviour

LoadingScreen
```

Collaboration diagram for LoadingScreen:

```
MonoBehaviour

LoadingScreen
```

4.128.1  Detailed Description

Definition at line 6 of file LoadingScreen.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Loading/LoadingScreen.cs
4.129 LoadingScreenFlavorText Class Reference

Collaboration diagram for LoadingScreenFlavorText:

```
LoadingScreenFlavorText
+ Text
```

Properties

- string Text [get, set]
  
  Text to display.

4.129.1 Detailed Description

Definition at line 1 of file LoadingScreenFlavorText.cs.

4.129.2 Property Documentation

4.129.2.1 Text

string LoadingScreenFlavorText.Text [get], [set]

Text to display.

Definition at line 9 of file LoadingScreenFlavorText.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Loading/LoadingScreenFlavorText.cs
4.130 LoadingScreenFlavorTextManager Class Reference

Collaboration diagram for LoadingScreenFlavorTextManager:

```
<table>
<thead>
<tr>
<th>LoadingScreenFlavorTextManager</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ LoadingScreenFlavorTextManager(filename)</td>
</tr>
<tr>
<td>+ GetRandom()</td>
</tr>
</tbody>
</table>
```

Public Member Functions

- **LoadingScreenFlavorTextManager**(string filename)
  
  Creates a new flavor text manager and loads flavor text in.

- **LoadingScreenFlavorText GetRandom()**
  
  Returns the a random flavor text.

4.130.1 Detailed Description

Definition at line 5 of file LoadingScreenFlavorTextManager.cs.

4.130.2 Constructor & Destructor Documentation

4.130.2.1 LoadingScreenFlavorTextManager()

```
LoadingScreenFlavorTextManager.LoadingScreenFlavorTextManager ( string filename )
```

Creates a new flavor text manager and loads flavor text in.

**Parameters**

| **filename** | Location of file with flavor text. |

Definition at line 13 of file LoadingScreenFlavorTextManager.cs.
4.130.3 Member Function Documentation

4.130.3.1 GetRandom()

LoadingScreenFlavorText LoadingScreenFlavorTextManager.GetRandom()

Returns the a random flavor text.

Returns
A random flavor text.

Definition at line 23 of file LoadingScreenFlavorTextManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Loading/LoadingScreenFlavorTextManager.cs

4.131 LoadingScreenFlavorTextYAMLParser Class Reference

Collaboration diagram for LoadingScreenFlavorTextYAMLParser:

```
LoadingScreenFlavorTextYAMLParser
+ LoadingScreenFlavorTextYAMLParser()
+ Load()
```

Public Member Functions

- **LoadingScreenFlavorTextYAMLParser** (string file)
  
  *Awake this instance.*

- **List< LoadingScreenFlavorText > Load ()**
  
  *Loads the flavor texts from the yaml file into list*

4.131.1 Detailed Description

Definition at line 14 of file LoadingScreenFlavorTextYAMLParser.cs.
4.131.2 Constructor & Destructor Documentation

4.131.2.1 LoadingScreenFlavorTextYAMLParser()

LoadingScreenFlavorTextYAMLParser.LoadingScreenFlavorTextYAMLParser ( 
  string file )

Awake this instance.

Definition at line 21 of file LoadingScreenFlavorTextYAMLParser.cs.

4.131.3 Member Function Documentation

4.131.3.1 Load()

List<LoadingScreenFlavorText> LoadingScreenFlavorTextYAMLParser.Load ( )

Loads the flavor texts from the yaml file into list

Returns
  The list of flavor texts.

Definition at line 30 of file LoadingScreenFlavorTextYAMLParser.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/LoadingScreenFlavorTextYAMLParser.cs
4.132 MainMenuBehavior Class Reference

Inheritance diagram for MainMenuBehavior:

Collaboration diagram for MainMenuBehavior:

Public Member Functions

- void OnSettingsClick ()
Loads the settings panel.

- void OnHelpClick()
  Loads the help panel.

- void OnExitClick()
  Loads the end credits panel.

- void OnStartClick()
  Loads the game scene.

- void OnBackToMainClick()
  Loads the main menu panel

4.132.1 Detailed Description

Definition at line 6 of file MainMenuBehavior.cs.

4.132.2 Member Function Documentation

4.132.2.1 OnBackToMainClick()

void MainMenuBehavior.OnBackToMainClick ()

Loads the main menu panel

Definition at line 73 of file MainMenuBehavior.cs.

4.132.2.2 OnExitClick()

void MainMenuBehavior.OnExitClick ()

Loads the end credits panel.

Definition at line 54 of file MainMenuBehavior.cs.

4.132.2.3 OnHelpClick()

void MainMenuBehavior.OnHelpClick ()

Loads the help panel.

Definition at line 43 of file MainMenuBehavior.cs.
4.132.2.4 OnSettingsClick()

void MainMenuBehavior.OnSettingsClick()

Loads the settings panel.

Definition at line 32 of file MainMenuBehavior.cs.

4.132.2.5 OnStartClick()

void MainMenuBehavior.OnStartClick()

Loads the game scene.

Definition at line 65 of file MainMenuBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/MainMenuScene/MainMenuBehavior.cs
4.133 MedicineCategory Class Reference

Inheritance diagram for MedicineCategory:

```
CollectableItem
   + GetPossibleActions()

ItemCategory
   + Attributes
   + Actions
   # baseItem

   + SetBaseItem()
   + GetPossibleActions()
   + ReadyCategory()
   + GetDuplicate()
   + SetActionComplete()
   + GetAttribute()
   # finishDuplication()

MedicineCategory
   + HealthGain
   + Sickness

   + GetDuplicate()
   + ReadyCategory()
   + Heal()
   + CurePneumonia()
   + CureFoodPoisoning()
   + CureAll()
```
Collaboration diagram for MedicineCategory:

Public Member Functions

- override ItemCategory GetDuplicate ()
  
  Creates a copy of this medicine category.

- override void ReadyCategory ()
  
  Readies the item category by adding the attributes and actions it can complete.

- void Heal ()
Heals the player based on the HealthRegain.

- void **CurePneumonia** ()
  Cures the player of pneumonia.
- void **CureFoodPoisoning** ()
  Cures the player of food poisoning.
- void **CureAll** ()
  Cures all illness and maxes health.

**Properties**

- float **HealthGain** [get, set]
  Gets or sets the HealthGain.
- string **Sickness** [get, set]
  Gets or sets the Sickness specialization (pneumonia, food poisoning, none, all).

**Additional Inherited Members**

4.133.1 Detailed Description

Definition at line 5 of file MedicineCategory.cs.

4.133.2 Member Function Documentation

4.133.2.1 **CureAll()**

void MedicineCategory.CureAll ( )

Cures all illness and maxes health.

Definition at line 149 of file MedicineCategory.cs.

4.133.2.2 **CureFoodPoisoning()**

void MedicineCategory.CureFoodPoisoning ( )

Cures the player of food poisoning.

Definition at line 136 of file MedicineCategory.cs.
4.133.2.3 CurePneumonia()

```csharp
void MedicineCategory.CurePneumonia()
```

Cures the player of pneumonia.

Definition at line 123 of file MedicineCategory.cs.

4.133.2.4 GetDuplicate()

```csharp
override ItemCategory MedicineCategory.GetDuplicate() [virtual]
```

Creates a copy of this medicine category.

Returns

The duplicate.

Reimplemented from `ItemCategory`.

Definition at line 38 of file MedicineCategory.cs.

4.133.2.5 Heal()

```csharp
void MedicineCategory.Heal()
```

Heals the player based on the HealthRegain.

Definition at line 114 of file MedicineCategory.cs.

4.133.2.6 ReadyCategory()

```csharp
override void MedicineCategory.ReadyCategory() [virtual]
```

Readies the item category by adding the attributes and actions it can complete.

Reimplemented from `ItemCategory`.

Definition at line 78 of file MedicineCategory.cs.

4.133.3 Property Documentation
4.133.3.1 HealthGain

float MedicineCategory.HealthGain [get], [set]

Gets or sets the HealthGain.

The health gain.

Definition at line 12 of file MedicineCategory.cs.

4.133.3.2 Sickness

string MedicineCategory.Sickness [get], [set]

Gets or sets the Sickness specialization (pnuemonia, food poisoning, none, all).

Definition at line 21 of file MedicineCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/MedicineCategory.cs
Inheritance diagram for Movement:

```plaintext
MonoBehaviour

Movement
+ Speed
+ CurrentFallDammage
+ RigidBody
+ AccumulatedFallDamage
+ climbHeight

+ Idle()
+ Move()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ OnStateEnter()
+ OnStateExit()
+ GetRaycastHeight()

LandMovement
+ Move()
+ Idle()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ GetRaycastHeight()
+ OnStateEnter()
+ OnStateExit()

RaftMovement
+ IsActive
+ BoardRaftText
+ DisembarkRaftText
+ PlayerAnimator
+ PlayerStandHeight

+ Idle()
+ Move()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ OnStateEnter()
+ OnStateExit()
+ GetRaycastHeight()
+ SetMaxSpeed()

WaterMovement
+ SplashManager
+ SwimmingHeight

+ Idle()
+ Move()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ GetRaycastHeight()
+ OnStateEnter()
+ OnStateExit()
```
Collaboration diagram for Movement:

```
MonoBehaviour

Movement

+ Speed
+ CurrentFallDammage
# RigidBody
# AccumulatedFallDamage
# climbHeight

+ Idle()
+ Move()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ OnStateEnter()
+ OnStateExit()
+ GetRaycastHeight()
```

Public Member Functions

- abstract void `Idle` (Animator playerAnimator)
- abstract void `Move` (Vector3 direction, bool sprinting, Animator playerAnimator)
- abstract void `Jump` (Animator playerAnimator)
- abstract void `Climb` (Animator playerAnimator)
- abstract float `GetClimbHeight` ()
- abstract void `OnStateEnter` ()
- abstract void `OnStateExit` ()
- abstract float `GetRaycastHeight` ()

Protected Attributes

- Rigidbody `RigidBody`
- float `AccumulatedFallDammage`
- float `climbHeight` = 0.356f

Properties

- float `Speed` [get, set]
  
  *Gets or sets the speed of the movement.*

- float `CurrentFallDammage` [get]
  
  *Gets accumulated fall damage and resets to 0*
4.134 Movement Class Reference

4.134.1 Detailed Description

Definition at line 4 of file Movement.cs.

4.134.2 Member Function Documentation

4.134.2.1 Climb()

abstract void Movement.Climb ( Animator playerAnimator ) [pure virtual]

Implemented in LandMovement, RaftMovement, and WaterMovement.

4.134.2.2 GetClimbHeight()

abstract float Movement.GetClimbHeight ( ) [pure virtual]

Implemented in LandMovement, RaftMovement, and WaterMovement.

4.134.2.3 GetRaycastHeight()

abstract float Movement.GetRaycastHeight ( ) [pure virtual]

Implemented in LandMovement, RaftMovement, and WaterMovement.

4.134.2.4 Idle()

abstract void Movement.Idle ( Animator playerAnimator ) [pure virtual]

Implemented in LandMovement, RaftMovement, and WaterMovement.

4.134.2.5 Jump()

abstract void Movement.Jump ( Animator playerAnimator ) [pure virtual]

Implemented in LandMovement, RaftMovement, and WaterMovement.
4.134.2.6  Move()

abstract void Movement.Move ( Vector3 direction, bool sprinting, Animator playerAnimator ) [pure virtual]

Implemented in RaftMovement, LandMovement, and WaterMovement.

4.134.2.7  OnStateEnter()

abstract void Movement.OnStateEnter ( ) [pure virtual]

Implemented in LandMovement, RaftMovement, and WaterMovement.

4.134.2.8  OnStateExit()

abstract void Movement.OnStateExit ( ) [pure virtual]

Implemented in LandMovement, RaftMovement, and WaterMovement.

4.134.3  Member Data Documentation

4.134.3.1  AccumulatedFallDamage

float Movement.AccumulatedFallDamage [protected]

Definition at line 18 of file Movement.cs.

4.134.3.2  climbHeight

float Movement.climbHeight = 0.356f [protected]

Definition at line 21 of file Movement.cs.
4.134.3  Rigidbody

Rigidbody Movement.Rigidbody [protected]
Definition at line 6 of file Movement.cs.

4.134.4  Property Documentation

4.134.4.1  CurrentFallDammage

float Movement.CurrentFallDammage [get]
Gets accumulated fall damage and resets to 0
Definition at line 45 of file Movement.cs.

4.134.4.2  Speed

float Movement.Speed [get], [set]
Gets or sets the speed of the movement.
The speed.
Definition at line 13 of file Movement.cs.
The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/Movement.cs

4.135  MysteryAnnouncement Class Reference

Mystery announcement base class

Collaboration diagram for MysteryAnnouncement:
Properties

- **AnnouncementCategory Category** [get, set]
  
  Gets or sets the category.

- **String EventPath** [get, set]
  
  Gets or sets the event path for this clip. If it's pre-generated, this comes in the YAML.

- **string Text** [get, set]
  
  Gets or sets the text that the TTS bot will say.

- **string Voice** [get, set]
  
  Gets or sets the voice that the announcement will be in.

- **string Speed** [get, set]
  
  Gets or sets the speed of the TTS playback.

4.135.1 Detailed Description

Mystery announcement base class

Definition at line 6 of file MysteryAnnouncement.cs.

4.135.2 Property Documentation

4.135.2.1 Category

**AnnouncementCategory MysteryAnnouncement.Category** [get], [set]

Gets or sets the category.

The category.

Definition at line 13 of file MysteryAnnouncement.cs.

4.135.2.2 EventPath

**String MysteryAnnouncement.EventPath** [get], [set]

Gets or sets the event path for this clip. If it's pre-generated, this comes in the YAML.

The event path.

Definition at line 24 of file MysteryAnnouncement.cs.
### 4.135.2.3 Speed

```csharp
string MysteryAnnouncement.Speed [get], [set]
```

Gets or sets the speed of the TTS playback.

The speed.

Definition at line 54 of file MysteryAnnouncement.cs.

### 4.135.2.4 Text

```csharp
string MysteryAnnouncement.Text [get], [set]
```

Gets or sets the text that the TTS bot will say.

The text.

Definition at line 34 of file MysteryAnnouncement.cs.

### 4.135.2.5 Voice

```csharp
string MysteryAnnouncement.Voice [get], [set]
```

Gets or sets the voice that the announcement will be in.

The voice.

Definition at line 44 of file MysteryAnnouncement.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Radio/Announcements/MysteryAnnouncement.cs
Base interactable note class for all notes.

Inheritance diagram for Note:
Collaboration diagram for Note:

```
MonoBehaviour

InteractableObject
+ Show
+ Text
# setupComplete
+ SetUp()
+ PerformAction()
+ SetAction()

Note
+ DisplayText
+ SetUpNote()
+ Read()
```

**Public Member Functions**

- void `SetUpNote()`
  
  Sets up the `InteractableObject`.

- void `Read()`
  
  Toggle visibility of `Note` UI panel onscreen.

**Properties**

- string `DisplayText` [get, set]
  
  Gets or sets the display text from the `NoteData` object.

**Additional Inherited Members**

4.136.1 Detailed Description

Base interactable note class for all notes.

Definition at line 9 of file Note.cs.
4.136.2 Member Function Documentation

4.136.2.1 Read()

```csharp
void Note.Read ( )
```

Toggle visibility of Note UI panel onscreen.

Definition at line 49 of file Note.cs.

4.136.2.2 SetUpNote()

```csharp
void Note SetUpNote ( )
```

Sets up the InteractableObject.

Definition at line 32 of file Note.cs.

4.136.3 Property Documentation

4.136.3.1 DisplayText

```csharp
string Note.DisplayText [get], [set]
```

Gets or sets the display text from the NoteData object.

The display text.

Definition at line 16 of file Note.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Notes/Note.cs
Base class for all notes.

Collaboration diagram for NoteData:

```
+ interactable
+ Title
+ Type
+ Text
+ WorldModel
+ InventorySprite
```

Properties

- **Note interactable** [get, set]
  
  Gets or sets the interactable Note object.

- **string Title** [get, set]
  
  Gets or sets the title of the note

- **string Type** [get, set]
  
  Gets or sets the type of the note

- **string Text** [get, set]
  
  Gets or sets the body text of the note

- **string WorldModel** [get, set]
  
  Gets or sets the model that represents the item in the world.

- **string InventorySprite** [get, set]
  
  Gets or sets the model that represents the item in the inventory.

4.137.1 Detailed Description

Base class for all notes.

Definition at line 4 of file NoteData.cs.

4.137.2 Property Documentation
4.137.2.1 interactable

```csharp
Note NoteData.interactable [get], [set]
```

Gets or sets the interactable Note object.

The interactable.

Definition at line 11 of file NoteData.cs.

4.137.2.2 InventorySprite

```csharp
string NoteData.InventorySprite [get], [set]
```

Gets or sets the model that represents the item in the inventory.

The inventory sprite model.

Definition at line 61 of file NoteData.cs.

4.137.2.3 Text

```csharp
string NoteData.Text [get], [set]
```

Gets or sets the body text of the note.

The body text of the note.

Definition at line 41 of file NoteData.cs.

4.137.2.4 Title

```csharp
string NoteData.Title [get], [set]
```

Gets or sets the title of the note.

The title of the note.

Definition at line 21 of file NoteData.cs.
4.137.2.5 Type

string NoteData.Type [get], [set]

Gets or sets the type of the note

The type of the note

Definition at line 31 of file NoteData.cs.

4.137.2.6 WorldModel

string NoteData.WorldModel [get], [set]

Gets or sets the model that represents the item in the world.

The world model.

Definition at line 51 of file NoteData.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Notes/NoteData.cs

4.138 NoteFactory Class Reference

Collaboration diagram for NoteFactory:

```
NoteFactory
+ NoteFactory() + LoadTemplates() + CreateInteractableItem()
```

Public Member Functions

- **NoteFactory ()**
  
  *Initializes a new instance of the NoteFactory class.*

- **void LoadTemplates ()**

  *Loads the note templates based on what world models are referenced in the NoteListYaml*

- **GameObject CreateInteractableItem (NoteData noteToCreate)**

  *Creates an interactable note item that is ready to be placed in the world.*
4.138.1 Detailed Description

Definition at line 6 of file NoteFactory.cs.

4.138.2 Constructor & Destructor Documentation

4.138.2.1 NoteFactory()

NoteFactory.NoteFactory ( )

Initializes a new instance of the NoteFactory class.

Definition at line 21 of file NoteFactory.cs.

4.138.3 Member Function Documentation

4.138.3.1 CreateInteractableItem()

GameObject NoteFactory.CreateInteractableItem ( NoteData noteToCreate )

Creates an interactable note item that is ready to be placed in the world.

Returns
   The interactable item.

Parameters
   noteToCreate [Note to create.]

Definition at line 50 of file NoteFactory.cs.

4.138.3.2 LoadTemplates()

void NoteFactory.LoadTemplates ( )

Loads the note templates based on what world models are referenced in the NoteListYaml

Definition at line 34 of file NoteFactory.cs.

The documentation for this class was generated from the following file:
   • Assets/Scripts/Notes/NoteFactory.cs
Note user interface controller. TODO: Make this a child of game UI

Inheritance diagram for NoteUIController:

Collaboration diagram for NoteUIController:
Public Member Functions

- void SetText (string text)
  Sets the text.
- void DisableUI ()
  Sets the UI active to false.

Public Attributes

- GameObject NoteUIPanel
  The note user interface panel.
- Text NoteText
  The note text.

Static Public Attributes

- static NoteUIController Instance
  The instance.

4.139.1 Detailed Description

Note user interface controller. TODO: Make this a child of game UI

Definition at line 9 of file NoteUIController.cs.

4.139.2 Member Function Documentation

4.139.2.1 DisableUI()

void NoteUIController.DisableUI ()

Sets the UI active to false.

Definition at line 47 of file NoteUIController.cs.

4.139.2.2 SetText()

void NoteUIController.SetText ( string text )

Sets the text.
Parameters

text | Text.

Definition at line 39 of file NoteUIController.cs.

4.139.3 Member Data Documentation

4.139.3.1 Instance

NoteUIController NoteUIController.Instance [static]

The instance.

Definition at line 14 of file NoteUIController.cs.

4.139.3.2 NoteText

Text NoteUIController.NoteText

The note text.

Definition at line 24 of file NoteUIController.cs.

4.139.3.3 NoteUIPanel

GameObject NoteUIController.NoteUIPanel

The note user interface panel.

Definition at line 19 of file NoteUIController.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Notes/NoteUIController.cs
4.140  NoteYAMLParser Class Reference

Collaboration diagram for NoteYAMLParser:

```
NoteYAMLParser
+ NoteYAMLParser()
+ LoadNotes()
```

Public Member Functions

- **NoteYAMLParser** (string file)
  
  Awake this instance.

- **List< NoteData > LoadNotes ()**
  
  Loads the notes from the yaml file into list

4.140.1  Detailed Description

Definition at line 14 of file NoteYAMLParser.cs.

4.140.2  Constructor & Destructor Documentation

4.140.2.1  NoteYAMLParser()

```
NoteYAMLParser.NoteYAMLParser (  
    string file )
```

Awake this instance.

Definition at line 21 of file NoteYAMLParser.cs.

4.140.3  Member Function Documentation
4.140.3.1 LoadNotes()

```csharp
List<NoteData> NoteYAMLParser.LoadNotes();
```

Loads the notes from the yaml file into list

Returns
The notes in list of `Note` objects.

Definition at line 30 of file NoteYAMLParser.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/NoteYAMLParser.cs

4.141 NullParent Class Reference

Sometimes you want something to be in a prefab, but not have a parent in the hierarchy. This script helps with that.

Inheritance diagram for NullParent:
4.141.1 Detailed Description

Sometimes you want something to be in a prefab, but not have a parent in the hierarchy. This script helps with that.

Definition at line 7 of file NullParent.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/NullParent.cs

4.142 OverworldItemOptionSelection Class Reference

Collaboration diagram for OverworldItemOptionSelection:
Public Member Functions

- **OverworldItemOptionSelection** (bool hasMultipleActions)
  Initializes a new instance of the OverworldItemOptionSelection class.
- **void ShowPossibleActions** ()
  Shows the possible actions in a gui on screen.
- **void ShowPossibleItems** (List<string> itensTypes, UnityAction callback)
  Shows the possible items in a gui on screen.
- **void AddPossibleAction** (ItemAction newAction)
  Adds an action that may be selected.
- **void OptionSelectedCallbackAction** (string selectionName)
  Callback action that will be fired off when an item has been selected.
- **void Reset** ()
  Resets this instance.

Properties

- **string SelectedItem** [get, set]
  Gets or sets the selected item.
- **GameObject TargetContentPanel** [get, set]
  Gets or sets the target content panel.
- **GameObject TargetContainerPanel** [get, set]
  Gets or sets the target container panel.

4.142.1 Detailed Description

Definition at line 6 of file OverworldItemOptionSelection.cs.

4.142.2 Constructor & Destructor Documentation

4.142.2.1 OverworldItemOptionSelection()

OverworldItemOptionSelection.OverworldItemOptionSelection (bool hasMultipleActions)

Initializes a new instance of the OverworldItemOptionSelection class.

Parameters

- **hasMultipleActions** If set to true has multiple actions.

Definition at line 50 of file OverworldItemOptionSelection.cs.
4.142.3 Member Function Documentation

4.142.3.1 AddPossibleAction()

```csharp
void OverworldItemOptionSelection.AddPossibleAction ( ItemAction newAction )
```

Adds an action that may be selected.

**Parameters**

- `newAction` New action.

Definition at line 81 of file `OverworldItemOptionSelection.cs`.

4.142.3.2 OptionSelectedCallbackAction()

```csharp
void OverworldItemOptionSelection.OptionSelectedCallbackAction ( string selectionName )
```

Callback action that will be fired off when an item has been selected.

**Parameters**

- `itemName` Item name.

Definition at line 90 of file `OverworldItemOptionSelection.cs`.

4.142.3.3 Reset()

```csharp
void OverworldItemOptionSelection.Reset ( )
```

Resets this instance.

Definition at line 144 of file `OverworldItemOptionSelection.cs`.

4.142.3.4 ShowPossibleActions()

```csharp
void OverworldItemOptionSelection.ShowPossibleActions ( )
```

Shows the possible actions in a gui on screen.

Definition at line 60 of file `OverworldItemOptionSelection.cs`.

Generated by Doxygen
4.142.3.5 ShowPossibleItems()

```csharp
void OverworldItemOptionSelection.ShowPossibleItems(
    List<string> itemTypes,
    UnityAction callback)
```

Shows the possible items in a gui on screen.

| itemTypes | List of item types that are possible items. |
| callback  | Callback.                                   |

Definition at line 71 of file OverworldItemOptionSelection.cs.

4.142.4 Property Documentation

4.142.4.1 SelectedItem

```csharp
string OverworldItemOptionSelection.SelectedItem [get], [set]
```

Gets or sets the selected item.

The selected item.

Definition at line 21 of file OverworldItemOptionSelection.cs.

4.142.4.2 TargetContainerPanel

```csharp
GameObject OverworldItemOptionSelection.TargetContainerPanel [get], [set]
```

Gets or sets the target container panel.

The target container panel.

Definition at line 41 of file OverworldItemOptionSelection.cs.

4.142.4.3 TargetContentPanel

```csharp
GameObject OverworldItemOptionSelection.TargetContentPanel [get], [set]
```

Gets or sets the target content panel.

The target content panel.

Definition at line 31 of file OverworldItemOptionSelection.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/OverworldItemOptionSelection.cs

Generated by Doxygen
4.143  PauseSystem Class Reference

Collaboration diagram for PauseSystem:

<table>
<thead>
<tr>
<th>PauseSystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ IsPaused</td>
</tr>
</tbody>
</table>
| + PauseDelegateUpdate()
| + MenuPauseDelegateUpdate()
| + ResumeDelegateUpdate()
| + PauseSystem()
| + Pause()
| + MenuPause()
| + Resume() |

Public Member Functions

- delegate void PauseDelegateUpdate ()
- delegate void MenuPauseDelegateUpdate ()
- delegate void ResumeDelegateUpdate ()
- PauseSystem ()

  Initializes a new instance of the PauseSystem class and starts the instance.
- void Pause ()

  Notify delegates to pause
- void MenuPause ()

  Pause that occurs when a menu is open.
- void Resume ()

  Notify delegates that game is resumed

Properties

- bool IsPaused  [get]

  Gets a value indicating whether this game is paused.

Events

- PauseDelegateUpdate PauseUpdate

  Occurs when game is paused.
- MenuPauseDelegateUpdate MenuPauseUpdate

  Occurs when a menu is open.
- ResumeDelegateUpdate ResumeUpdate

  Occurs when unpause/menu closed update.
4.143 PauseSystem Class Reference

4.143.1 Detailed Description

Definition at line 1 of file PauseSystem.cs.

4.143.2 Constructor & Destructor Documentation

4.143.2.1 PauseSystem()

PauseSystem.PauseSystem ( )

Initializes a new instance of the PauseSystem class and starts the instance.

Definition at line 37 of file PauseSystem.cs.

4.143.3 Member Function Documentation

4.143.3.1 MenuPause()

void PauseSystem.MenuPause ( )

Pause that occurs when a menu is open.

Definition at line 57 of file PauseSystem.cs.

4.143.3.2 MenuPauseDelegateUpdate()

delegate void PauseSystem.MenuPauseDelegateUpdate ( )

4.143.3.3 Pause()

void PauseSystem.Pause ( )

Notify delegates to pause

Definition at line 45 of file PauseSystem.cs.
4.143.3.4 PauseDelegateUpdate()

delegate void PauseSystem.PauseDelegateUpdate();

4.143.3.5 Resume()

void PauseSystem.Resume();

Notify delegates that game is resumed

Definition at line 70 of file PauseSystem.cs.

4.143.3.6 ResumeDelegateUpdate()

delegate void PauseSystem.ResumeDelegateUpdate();

4.143.4 Property Documentation

4.143.4.1 IsPaused

bool PauseSystem.IsPaused { get }

Gets a value indicating whether this game is paused.

true if this instance is paused; otherwise, false.

Definition at line 28 of file PauseSystem.cs.

4.143.5 Event Documentation

4.143.5.1 MenuPauseUpdate

MenuPauseDelegateUpdate PauseSystem.MenuPauseUpdate

Occurs when a menu is open.

Definition at line 15 of file PauseSystem.cs.
4.143.2 PauseUpdate

*PauseDelegateUpdate* PauseSystem.PauseUpdate

Occurs when game is paused.

Definition at line 9 of file PauseSystem.cs.

4.143.3 ResumeUpdate

*ResumeDelegateUpdate* PauseSystem.ResumeUpdate

Occurs when unpaused/menu closed update.

Definition at line 21 of file PauseSystem.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/PauseSystem.cs

4.144 PickUpItem Class Reference

Inheritance diagram for PickUpItem:

```
MonoBehaviour

InteractableObject
+ Show
+ Text
  # setupComplete
+ SetUp()
+ PerformAction()
+ SetAction()

PickUpItem
+ Item
+ Amount
+ SetUp()
```
Collaboration diagram for PickUpItem:

Public Member Functions

- `override void SetUp()`

  Sets pick up as an action that should fire off when PerformAction is called.

Properties

- `BaseItem Item [get, set]`

  Gets or sets the base item that this object should pick up.

- `int Amount [get, set]`

  How many of the item will be picked up.

Additional Inherited Members

4.144.1 Detailed Description

Definition at line 5 of file PickUpItem.cs.
4.144.2 Member Function Documentation

4.144.2.1 SetUp()

override void PickUpItem.SetUp ( ) [virtual]

Sets pick up as an action that should fire off when PerformAction is called.

Reimplemented from InteractableObject.

Definition at line 44 of file PickUpItem.cs.

4.144.3 Property Documentation

4.144.3.1 Amount

int PickUpItem.Amount [get], [set]

How many of the item will be picked up.

The amount.

Definition at line 36 of file PickUpItem.cs.

4.144.3.2 Item

BaseItem PickUpItem.Item [get], [set]

Gets or sets the base item that this object should pick up.

The item.

Definition at line 19 of file PickUpItem.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/PickUpItem.cs
4.145 PlantCategory Class Reference

Contains attributes and actions that befit a plant category item.

Inheritance diagram for PlantCategory:
Public Member Functions

- **override** ItemCategory GetDuplicate ()
  
  Creates a copy of this plant category.

- **override** void ReadyCategory ()
  
  Readies the item category by adding the attributes and actions it can complete.

- **void** Cook ()
Cooks the item. Lowers toughness and increases water content. If the item has high enough water content, it becomes soup. Otherwise, it becomes Cooked X.

- void Dry ()
  Dries this item. Lowers the waterContent greatly and increases toughness.

- void Eat ()
  Consumes the item. May cause illness.

Properties

- float WaterContent [get, set]
  Gets or sets the water content of the item.

- float Toughness [get, set]
  Gets or sets the toughness.

- float Sweet [get, set]
  Gets or sets the sweetness.

- float Bitter [get, set]
  Gets or sets the bitterness.

- float Sour [get, set]
  Gets or sets the sourness.

- float Salty [get, set]
  Gets or sets the saltiness.

- float Spicy [get, set]
  Gets or sets the spiciness.

- float StomachEffect [get, set]
  Gets or sets the stomach effect.

- float PneumoniaEffect [get, set]
  Gets or sets the pneumonia effect.

Additional Inherited Members

4.145.1 Detailed Description

Contains attributes and actions that befit a plant category item.

Definition at line 8 of file PlantCategory.cs.

4.145.2 Member Function Documentation

4.145.2.1 Cook()

void PlantCategory.Cook ( )

Cooks the item. Lowers toughness and increases water content. If the item has high enough water content, it becomes soup. Otherwise, it becomes Cooked X.

Definition at line 199 of file PlantCategory.cs.
4.145.2.2 Dry()

void PlantCategory.Dry ()

Dries this item. Lowers the waterContent greatly and increases toughness.

Definition at line 220 of file PlantCategory.cs.

4.145.2.3 Eat()

void PlantCategory.Eat ()

Consumes the item. May cause illness.

Definition at line 237 of file PlantCategory.cs.

4.145.2.4 GetDuplicate()

override ItemCategory PlantCategory.GetDuplicate () [virtual]

Creates a copy of this plant category.

Returns

The duplicate.

Reimplemented from ItemCategory.

Definition at line 139 of file PlantCategory.cs.

4.145.2.5 ReadyCategory()

override void PlantCategory.ReadyCategory () [virtual]

Readies the item category by adding the attributes and actions it can complete.

Reimplemented from ItemCategory.

Definition at line 170 of file PlantCategory.cs.

4.145.3 Property Documentation
4.145.3.1 Bitter

float PlantCategory.Bitter [get], [set]

Gets or sets the bitterness.

The bitter.

Definition at line 45 of file PlantCategory.cs.

4.145.3.2 PneumoniaEffect

float PlantCategory.PneumoniaEffect [get], [set]

Gets or sets the pneumonia effect.

The pneumonia effect.

Definition at line 95 of file PlantCategory.cs.

4.145.3.3 Salty

float PlantCategory.Salty [get], [set]

Gets or sets the saltiness.

The salty.

Definition at line 65 of file PlantCategory.cs.

4.145.3.4 Sour

float PlantCategory.Sour [get], [set]

Gets or sets the sourness.

The sour.

Definition at line 55 of file PlantCategory.cs.
4.145.3.5  Spicy

float PlantCategory.Spicy [get], [set]

Gets or sets the spiciness.

The spicy.

Definition at line 75 of file PlantCategory.cs.

4.145.3.6  StomachEffect

float PlantCategory.StomachEffect [get], [set]

Gets or sets the stomach effect.

The stomach effect.

Definition at line 85 of file PlantCategory.cs.

4.145.3.7  Sweet

float PlantCategory.Sweet [get], [set]

Gets or sets the sweetness.

The sweet.

Definition at line 35 of file PlantCategory.cs.

4.145.3.8  Toughness

float PlantCategory.Toughness [get], [set]

Gets or sets the toughness.

The toughness.

Definition at line 25 of file PlantCategory.cs.
4.145.3.9 WaterContent

float PlantCategory.WaterContent [get], [set]

Gets or sets the water content of the item.

The content of the water.

Definition at line 15 of file PlantCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/PlantCategory.cs

4.146 Player Class Reference

Collaboration diagram for Player:

<table>
<thead>
<tr>
<th>Player</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ FoodPoisoningChance</td>
</tr>
<tr>
<td>+ Controller</td>
</tr>
<tr>
<td>+ IsInWorld</td>
</tr>
<tr>
<td>+ WorldTransform</td>
</tr>
<tr>
<td>+ WorldPosition</td>
</tr>
<tr>
<td>+ Health</td>
</tr>
<tr>
<td>+ MaxHealth</td>
</tr>
<tr>
<td>+ Warmth</td>
</tr>
<tr>
<td>+ MaxWarmth</td>
</tr>
<tr>
<td>+ Hunger</td>
</tr>
<tr>
<td>+ MaxHunger</td>
</tr>
<tr>
<td>+ Inventory</td>
</tr>
<tr>
<td>+ Toolbox</td>
</tr>
<tr>
<td>+ HealthStatus</td>
</tr>
<tr>
<td>+ Player()</td>
</tr>
<tr>
<td>+ ResetStatus()</td>
</tr>
</tbody>
</table>

Public Member Functions

- **Player ()**
  
  *Player constructor. Initializes all fields to full.*

- **void ResetStatus ()**

  *Resets player to initial status.*

Public Attributes

- **const float FoodPoisoningChance = 0.3f**
Properties

- **PlayerController Controller** [get, set]
  
  Gets or sets the player controller if we are in a scene with the player controller.

- **bool IsInWorld** [get]
  
  Returns true if the player is instantiated in the game scene.

- **Transform WorldTransform** [get, set]
  
  Player's transform in the game world. Returns null if the player is not in the scene, check IsInWorld before using the value directly.

- **Vector3 WorldPosition** [get]
  
  Gets the position of the player in the world

- **int Health** [get, set]
  
  The player's current health. Used for physical damage.

- **int MaxHealth** [get]
  
  The player's maximum health. Used for physical damage.

- **int Warmth** [get, set]
  
  The player's current warmth. Warmth is reduced when the player is exposed to cold.

- **int MaxWarmth** [get]
  
  The player's maximum warmth. Warmth is reduced when the player is exposed to cold.

- **int Hunger** [get, set]
  
  The player's current hunger. Hunger is reduced when the player does not eat.

- **int MaxHunger** [get]
  
  The player's maximum hunger. Hunger is reduced when the player does not eat.

- **PlayerInventory Inventory** [get]
  
  Gets the player's on-person inventory.

- **PlayerTools Toolbox** [get, set]
  
  Controls the player's tools.

- **PlayerHealthStatus HealthStatus** [get, set]
  
  The player's current health status.

### 4.146.1 Detailed Description

Definition at line 14 of file Player.cs.

### 4.146.2 Constructor & Destructor Documentation

#### 4.146.2.1 Player()

Player.Player ( )

Player constructor. Initializes all fields to full.

Definition at line 31 of file Player.cs.
4.146.3 Member Function Documentation

4.146.3.1 ResetStatus()

```cpp
void Player.ResetStatus ( )
```

Resets player to initial status.

Definition at line 39 of file Player.cs.

4.146.4 Member Data Documentation

4.146.4.1 FoodPoisoningChance

```cpp
const float Player.FoodPoisoningChance = 0.3f
```

Definition at line 25 of file Player.cs.

4.146.5 Property Documentation

4.146.5.1 Controller

```cpp
PlayerController Player.Controller [get], [set]
```

Gets or sets the player controller if we are in a scene with the player controller.

The controller

Definition at line 59 of file Player.cs.

4.146.5.2 Health

```cpp
int Player.Health [get], [set]
```

The player's current health. Used for physical damage.

Definition at line 113 of file Player.cs.
4.146.5.3 HealthStatus

`PlayerHealthStatus` `Player.HealthStatus [get], [set]`

The player's current health status.

Definition at line 203 of file Player.cs.

4.146.5.4 Hunger

`int Player.Hunger [get], [set]`

The player's current hunger. Hunger is reduced when the player does not eat.

Definition at line 165 of file Player.cs.

4.146.5.5 Inventory

`PlayerInventory` `Player.Inventory [get]`

Gets the player's on-person inventory.

The player's inventory.

Definition at line 185 of file Player.cs.

4.146.5.6 IsInWorld

`bool Player.IsInWorld [get]`

Returns true if the player is instantiated in the game scene.

Definition at line 68 of file Player.cs.

4.146.5.7 MaxHealth

`int Player.MaxHealth [get]`

The player's maximum health. Used for physical damage.

Definition at line 135 of file Player.cs.
4.146.5.8 MaxHunger

```csharp
int Player.MaxHunger [get]
```

The player's maximum hunger. Hunger is reduced when the player does not eat.

Definition at line 175 of file Player.cs.

4.146.5.9 MaxWarmth

```csharp
int Player.MaxWarmth [get]
```

The player's maximum warmth. Warmth is reduced when the player is exposed to cold.

Definition at line 155 of file Player.cs.

4.146.5.10 Toolbox

```csharp
PlayerTools Player.Toolbox [get], [set]
```

Controls the player’s tools.

Definition at line 194 of file Player.cs.

4.146.5.11 Warmth

```csharp
int Player.Warmth [get], [set]
```

The player's current warmth. Warmth is reduced when the player is exposed to cold.

Definition at line 145 of file Player.cs.

4.146.5.12 WorldPosition

```csharp
Vector3 Player.WorldPosition [get]
```

Gets the position of the player in the world

The world position.

Definition at line 90 of file Player.cs.
4.146.5.13 WorldTransform

Transform Player.WorldTransform [get], [set]

Player's transform in the game world. Returns null if the player is not in the scene, check IsInWorld before using the value directly.

Definition at line 80 of file Player.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Player/Player.cs

4.147 PlayerAnimationEvents Class Reference

This script has functions called by the player's animator.

Inheritance diagram for PlayerAnimationEvents:
Collaboration diagram for PlayerAnimationEvents:

```
MonoBehaviour

PlayerAnimationEvents
+ CallJumpForce()
+ CallJumpLand()
```

Public Member Functions

- void `CallJumpForce()`
  
  *The player's rigidbody gets the jump force applied*

- void `CallJumpLand()`
  
  *Sets the player's jumping bool to false;

### 4.147.1 Detailed Description

This script has functions called by the player's animator.

This lets us time functions to be called during animations.

Definition at line 9 of file PlayerAnimationEvents.cs.

### 4.147.2 Member Function Documentation

#### 4.147.2.1 CallJumpForce()

```c++
void PlayerAnimationEvents.CallJumpForce()
```

*The player's rigidbody gets the jump force applied*

Definition at line 17 of file PlayerAnimationEvents.cs.
4.147.2.2 CallJumpLand()

```csharp
void PlayerAnimationEvents.CallJumpLand ()
```

Sets the player's jumping bool to false;

Definition at line 25 of file PlayerAnimationEvents.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/PlayerAnimationEvents.cs

4.148 PlayerController Class Reference

Inheritance diagram for PlayerController:

```
MonoBehaviour
+

PlayerController
+ ViewRadius
+ ViewAngle
+ PlayerStatManager
+ PlayerIKSetUp
+ ClimbingRaycastMask
+ IsGrounded
+ Interactable
+ PlayerAnimator
+ IsOnLand
+ IsInWater
+ IsOnRaft
+ IsInShelter
+ IsByFire
+ IsWaterInView
+ IsReading
+
+ BoardRaft()
+ DisembarkRaft()
+ CheckClosestInteractable()
+ DirFromAngle()
+ ClosestItem()
+ Resume()
+ Pause()
+ MenuPause()
+ StartWalkingSound()
+ StopWalkingSound()
+ SetIsOnLand()
```
Collaboration diagram for PlayerController:

```plaintext
MonoBehaviour

+ PlayerStatManager
+ ViewRadius
+ ViewAngle
+ PlayerIKSetUp
+ ClimbingRaycastMask
+ IsGrounded
+ Interactable
+ PlayerAnimator
+ IsOnLand
+ IsInWater
+ IsOnRaft
+ IsInShelter
+ IsByFire
+ IsWaterInView
+ IsReading

+ BoardRaft()
+ DisembarkRaft()
+ CheckClosestInteractable()
+ DirFromAngle()
+ ClosestItem()
+ Resume()
+ Pause()
+ MenuPause()
+ StartWalkingSound()
+ StopWalkingSound()
+ SetIsOnLand()

PlayerStatManager
+ WarmthRate
+ HungerRate
+ HealthRate
+ StopStats
+ PlayerStatManager()
+ ApplyCorrectHealthReductionRate()
+ StartStatUpdates()
+ StopStatUpdates()
```

Public Member Functions

- void `BoardRaft` ([RaftMovement](#) raftMovement)  
  _Player_ boards the raft and assumes raft controls until the player dismounts.
- void `DisembarkRaft` ([RaftMovement](#) raftMovement)  
  _Player_ disembarks the raft and resumes player movement.
- void `CheckClosestInteractable` ([Collider](#) target, float targetDist)  
  Check if an interactable is the closest interactable in view.
- void `DirFromAngle` (float angleInDegrees)  
  Gets the direction of the angle. Used for editor mode to see how big the field of view will be.
• Collider ClosestItem ()
  Returns the closest interactable item.

• void Resume ()
  Resume stat changes.

• void Pause ()
  Pause stat changes.

• void MenuPause ()
  Handles the pausing for menus.

• void StartWalkingSound ()
  Starts the walking sound.

• void StopWalkingSound ()
  Stops the walking sound.

• void SetIsOnLand ()
  Makes the player behave like they're on land. Called when the player teleports. Or by other things that'd move the player to be on land.

Public Attributes

• float ViewRadius
• float ViewAngle
• PlayerStatManager PlayerStatManager
• BipedIK PlayerIKSetUp
• LayerMask ClimbingRaycastMask

Properties

• bool IsGrounded [get]
• InteractableObject Interactable [get]
• Animator PlayerAnimator [get]
• bool IsOnLand [get]
  Returns true of the player is on solid ground
• bool IsInWater [get]
  Returns true if the player is swimming in water
• bool IsOnRaft [get]
  Returns true if the player is currently on a raft
• bool IsInShelter [get, set]
  Returns true if the player is currently in a shelter
• bool IsByFire [get, set]
  If the player is near a fire it returns true.
• bool IsWaterInView [get]
  Gets a value indicating whether player is by water.
• bool IsReading [get, set]
  If the player is reading returns true.

4.148.1 Detailed Description

Definition at line 8 of file PlayerController.cs.
4.148.2 Member Function Documentation

4.148.2.1 BoardRaft()

```csharp
void PlayerController.BoardRaft(
    RaftMovement raftMovement)
```

*Player* boards the raft and assumes raft controls until the player dismounts.

Definition at line 596 of file PlayerController.cs.

4.148.2.2 CheckClosestInteractable()

```csharp
void PlayerController.CheckClosestInteractable(
    Collider target,
    float targetDist)
```

Check if an interactable is the closest interactable in view.

**Parameters**

<table>
<thead>
<tr>
<th>interactable</th>
<th>distance</th>
</tr>
</thead>
</table>

Definition at line 720 of file PlayerController.cs.

4.148.2.3 ClosestItem()

```csharp
Collider PlayerController.ClosestItem()
```

Returns the closest interactable item.

Returns

Definition at line 753 of file PlayerController.cs.

4.148.2.4 DirFromAngle()

```csharp
Vector3 PlayerController.DirFromAngle(
    float angleInDegrees)
```

Gets the direction of the angle. Used for editor mode to see how big the field of view will be.
Parameters

angleInDegrees

Returns

Definition at line 742 of file PlayerController.cs.

4.148.2.5 DisembarkRaft()

void PlayerController.DisembarkRaft (
   RaftMovement raftMovement )

Player disembarks the raft and resumes player movement.

Parameters

raftMovement

Definition at line 619 of file PlayerController.cs.

4.148.2.6 MenuPause()

void PlayerController.MenuPause ( )

Handles the pausing for menus.

Definition at line 849 of file PlayerController.cs.

4.148.2.7 Pause()

void PlayerController.Pause ( )

Pause stat changes.

Definition at line 840 of file PlayerController.cs.
### 4.148.2.8 Resume()

```csharp
void PlayerController.Resume()
```

Resume stat changes.

Definition at line 827 of file PlayerController.cs.

### 4.148.2.9 SetIsOnLand()

```csharp
void PlayerController.SetIsOnLand()
```

Makes the player behave like they're on land. Called when the player teleports. Or by other things that'd move the player to be on land.

Definition at line 1064 of file PlayerController.cs.

### 4.148.2.10 StartWalkingSound()

```csharp
void PlayerController.StartWalkingSound()
```

Starts the walking sound.

Definition at line 873 of file PlayerController.cs.

### 4.148.2.11 StopWalkingSound()

```csharp
void PlayerController.StopWalkingSound()
```

Stops the walking sound.

Definition at line 890 of file PlayerController.cs.

### 4.148.3 Member Data Documentation

#### 4.148.3.1 ClimbingRaycastMask

```csharp
LayerMask PlayerController.ClimbingRaycastMask
```

Definition at line 151 of file PlayerController.cs.
4.148.3.2 PlayerIKSetUp

BipedIK PlayerController.PlayerIKSetUp

Definition at line 150 of file PlayerController.cs.

4.148.3.3 PlayerStatManager

PlayerStatManager PlayerController.PlayerStatManager

Definition at line 98 of file PlayerController.cs.

4.148.3.4 ViewAngle

float PlayerController.ViewAngle

Definition at line 58 of file PlayerController.cs.

4.148.3.5 ViewRadius

float PlayerController.ViewRadius

Definition at line 56 of file PlayerController.cs.

4.148.4 Property Documentation

4.148.4.1 Interactable

InteractableObject PlayerController.Interactable [get]

Definition at line 104 of file PlayerController.cs.

4.148.4.2 IsByFire

bool PlayerController.IsByFire [get], [set]

If the player is near a fire it returns true.

Definition at line 777 of file PlayerController.cs.
4.148.4.3 IsGrounded

bool PlayerController.IsGrounded [get]

Definition at line 84 of file PlayerController.cs.

4.148.4.4 IsInShelter

bool PlayerController.IsInShelter [get], [set]

Returns true if the player is currently in a shelter

Definition at line 762 of file PlayerController.cs.

4.148.4.5 IsInWater

bool PlayerController.IsInWater [get]

Returns true if the player is swimming in water

Definition at line 652 of file PlayerController.cs.

4.148.4.6 IsOnLand

bool PlayerController.IsOnLand [get]

Returns true if the player is on solid ground

Definition at line 644 of file PlayerController.cs.

4.148.4.7 IsOnRaft

bool PlayerController.IsOnRaft [get]

Returns true if the player is currently on a raft

Definition at line 660 of file PlayerController.cs.
4.148.8 IsReading

bool PlayerController.IsReading [get], [set]
If the player is reading returns true.
Definition at line 804 of file PlayerController.cs.

4.148.9 IsWaterInView

bool PlayerController.IsWaterInView [get]
Gets a value indicating whether player is by water.
true if this instance is by water; otherwise, false.
Definition at line 793 of file PlayerController.cs.

4.148.10 PlayerAnimator

Animator PlayerController.PlayerAnimator [get]
Definition at line 142 of file PlayerController.cs.
The documentation for this class was generated from the following file:

- Assets/Scripts/Player/PlayerController.cs

4.149 PlayerFocus Class Reference

Controls the player's head IK setup to have them look at the closest interactable if they can.

Inheritance diagram for PlayerFocus:

```
  MonoBehaviour
   |
   |
PlayerFocus
```
Collaboration diagram for PlayerFocus:

![Collaboration Diagram]

4.149.1 Detailed Description

Controls the player's head IK setup to have them look at the closest interactable if they can.

Definition at line 11 of file PlayerFocus.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/PlayerFocus.cs
4.150 PlayerInventory Class Reference

Inheritance diagram for PlayerInventory:

```
<table>
<thead>
<tr>
<th>Inventory</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ InventorySize</td>
</tr>
<tr>
<td># contents</td>
</tr>
<tr>
<td>+ Inventory()</td>
</tr>
<tr>
<td>+ Inventory()</td>
</tr>
<tr>
<td>+ LoadInventory()</td>
</tr>
<tr>
<td>+ GetInventory()</td>
</tr>
<tr>
<td>+ UseItem()</td>
</tr>
<tr>
<td>+ GetStacks()</td>
</tr>
<tr>
<td>+ GetInventoryBaseItem()</td>
</tr>
<tr>
<td>+ GetItemsByType()</td>
</tr>
<tr>
<td>+ AddItem()</td>
</tr>
<tr>
<td>+ RemoveStack()</td>
</tr>
<tr>
<td>and 6 more...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PlayerInventory</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ EquipedItem</td>
</tr>
<tr>
<td>+ PlayerInventory()</td>
</tr>
<tr>
<td>+ ItemAddedDelegate()</td>
</tr>
<tr>
<td>+ ItemRemovedDelegate()</td>
</tr>
<tr>
<td>+ ItemEquippedDelegate()</td>
</tr>
<tr>
<td>+ ItemUnequippedDelegate()</td>
</tr>
<tr>
<td>+ AddItem()</td>
</tr>
<tr>
<td>+ RemoveStack()</td>
</tr>
</tbody>
</table>
```
Collaboration diagram for PlayerInventory:

```
#include <iostream>
#include <vector>

class BaseItem {
public:
    virtual ~BaseItem() {}
private:
};

class ItemStack {
public:
    ItemStack();
    ~ItemStack();
    void UpdateStackAmountEvent();
private:
    BaseItem* item;
    int amount;
};

class Inventory {
public:
    Inventory();
    ~Inventory();
    void AddItem();
    void RemoveStack();
    void LoadInventory();
    void GetInventory();
    void UseItem();
    void GetStaks();
    void GetInventoryBaseItem();
    void GetItemsByType();
    void AddItem();
    void RemoveStack();
private:
    std::vector<ItemStack> stack;
};

class PlayerInventory {
public:
    PlayerInventory();
    ~PlayerInventory();
    void AddItemDelegate(BaseItem item);
    void ItemAddedDelegate(BaseItem item);
    void ItemRemovedDelegate(BaseItem item);
    void ItemEquippedDelegate(BaseItem item);
    void ItemUnequippedDelegate();
    void AddItem();
    void RemoveStack();
private:
    std::vector<ItemStack> stack;
};
```

Public Member Functions

- **PlayerInventory** (string name, string inventoryFile, int size)
  
  Creates an instance of **PlayerInventory**

- delegate void ItemAddedDelegate (BaseItem item)
- delegate void ItemRemovedDelegate (BaseItem item)
- delegate void ItemEquippedDelegate (BaseItem item)
- delegate void ItemUnequippedDelegate ()
• `ItemStack` `AddItem` (`BaseItem` `newItem`, `int` `amount`)  
  Add item to inventory and notify subscribers.

• `void` `RemoveStack` (`ItemStack` `stack`)  
  Removes the item from the inventory and notifies subscribers.

Properties

• `BaseItem` `EquipedItem`  
  Get or set the currently equipped item.

Events

• `ItemAddedDelegate` `ItemAddedSubscription`  
  Subscription triggered when an item is added to the inventory.

• `ItemRemovedDelegate` `ItemRemovedSubscription`  
  Subscription triggered when an item is removed from the inventory.

• `ItemEquippedDelegate` `ItemEquippedSubscription`  
  Subscription triggered when an item is equipped.

• `ItemUnequippedDelegate` `ItemUnequippedSubscription`  
  Subscription triggered when an equipped item is unequipped.

Additional Inherited Members

4.150.1 Detailed Description

Definition at line 4 of file PlayerInventory.cs.

4.150.2 Constructor & Destructor Documentation

4.150.2.1 PlayerInventory()

`PlayerInventory.PlayerInventory`  
`string` `name`,  
`string` `inventoryFile`,  
`int` `size`)

Creates an instance of `PlayerInventory`

Parameters

<table>
<thead>
<tr>
<th><code>name</code></th>
<th>Name of inventory.</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>inventoryFile</code></td>
<td>File to load inventory from.</td>
</tr>
</tbody>
</table>
Definition at line 13 of file PlayerInventory.cs.

4.150.3 Member Function Documentation

4.150.3.1 AddItem()

```csharp
ItemStack PlayerInventory.AddItem (
    BaseItem newItem,
    int amount )
```

Add item to inventory and notify subscribers.

Returns

The added item.

Parameters

<table>
<thead>
<tr>
<th>newItem</th>
<th>New item.</th>
</tr>
</thead>
<tbody>
<tr>
<td>amount</td>
<td>Amount.</td>
</tr>
</tbody>
</table>

Definition at line 49 of file PlayerInventory.cs.

4.150.3.2 ItemAddedDelegate()

```csharp
delegate void PlayerInventory.ItemAddedDelegate ( 
    BaseItem item )
```

4.150.3.3 ItemEquippedDelegate()

```csharp
delegate void PlayerInventory.ItemEquippedDelegate ( 
    BaseItem item )
```

4.150.3.4 ItemRemovedDelegate()

```csharp
delegate void PlayerInventory.ItemRemovedDelegate ( 
    BaseItem item )
```
4.150.3.5  ItemUnequippedDelegate()

delegate void PlayerInventory.ItemUnequippedDelegate ( )

4.150.3.6  RemoveStack()

void PlayerInventory.RemoveStack ( 
    ItemStack stack )

Removes the item from the inventory and notifies subscribers.

Parameters

stack  Item to remove.

Definition at line 65 of file PlayerInventory.cs.

4.150.4  Property Documentation

4.150.4.1  EquipedItem

BaseItem PlayerInventory.EquipedItem  [get],  [set]

Get or set the currently equipped item.

Null represents no currently equipped tool.

Definition at line 83 of file PlayerInventory.cs.

4.150.5  Event Documentation

4.150.5.1  ItemAddedSubscription

ItemAddedDelegate PlayerInventory.ItemAddedSubscription

Subscription triggered when an item is added to the inventory.

Definition at line 22 of file PlayerInventory.cs.
4.150.5.2 ItemEquippedSubscription

**ItemEquippedDelegate** PlayerInventory.ItemEquippedSubscription

Subscription triggered when an item is equipped.

Definition at line 34 of file PlayerInventory.cs.

4.150.5.3 ItemRemovedSubscription

**ItemRemovedDelegate** PlayerInventory.ItemRemovedSubscription

Subscription triggered when an item is removed from the inventory.

Definition at line 28 of file PlayerInventory.cs.

4.150.5.4 ItemUnequippedSubscription

**ItemUnequippedDelegate** PlayerInventory.ItemUnequippedSubscription

Subscription triggered when an equipped item is unequipped.

Definition at line 40 of file PlayerInventory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/PlayerInventory.cs

4.151 PlayerStatManager Class Reference

Collaboration diagram for PlayerStatManager:

```
+ PlayerStatManager()
+ ApplyCorrectHealthReductionRate()
+ StartStatUpdates()
+ StopStatUpdates()

+ WarthRate
+ HungerRate
+ HealthRate
+ StopStats
```

Generated by Doxygen
Public Member Functions

- **PlayerStatManager()**
  
  Initializes a new instance of the `PlayerStatManager` class.

- **void ApplyCorrectHealthReductionRate()**
  
  Applies the correct health reduction rate based on hunger and warmth stats.

- **void StartStatUpdates()**
  
  Starts the player's stat updates.

- **void StopStatUpdates()**
  
  Stops the player's stat updates.

Properties

- **WarmthRateManager WarmthRate [get]**
  
  Gets the warmth rate.

- **HungerRateManager HungerRate [get]**
  
  Gets the hunger rate.

- **HealthRateManager HealthRate [get]**
  
  Gets the health rate.

- **bool StopStats [get, set]**
  
  Gets or sets a value indicating whether this `PlayerStatManager` stops stats.

4.151.1 Detailed Description

Definition at line 3 of file `PlayerStatManager.cs`.

4.151.2 Constructor & Destructor Documentation

4.151.2.1 PlayerStatManager()

`PlayerStatManager.PlayerStatManager()`

Initializes a new instance of the `PlayerStatManager` class.

Definition at line 8 of file `PlayerStatManager.cs`.

4.151.3 Member Function Documentation
Class Documentation

4.151.3.1 ApplyCorrectHealthReductionRate()

```csharp
void PlayerStatManager.ApplyCorrectHealthReductionRate()
```

Applies the correct health reduction rate based on hunger and warmth stats.

Definition at line 19 of file PlayerStatManager.cs.

4.151.3.2 StartStatUpdates()

```csharp
void PlayerStatManager.StartStatUpdates()
```

Starts the player's stat updates.

Definition at line 34 of file PlayerStatManager.cs.

4.151.3.3 StopStatUpdates()

```csharp
void PlayerStatManager.StopStatUpdates()
```

 Stops the player's stat updates.

Definition at line 45 of file PlayerStatManager.cs.

4.151.4 Property Documentation

4.151.4.1 HealthRate

```csharp
HealthRateManager PlayerStatManager.HealthRate
```

Gets the health rate.

The health rate.

Definition at line 81 of file PlayerStatManager.cs.
4.151.4.2 HungerRate

**HungerRateManager** PlayerStatManager.HungerRate [get]

Gets the hunger rate.

The hunger rate.

Definition at line 71 of file PlayerStatManager.cs.

4.151.4.3 StopStats

**bool** PlayerStatManager.StopStats [get], [set]

Gets or sets a value indicating whether this **PlayerStatManager** stop stats.

true if stop stats; otherwise, false.

Definition at line 91 of file PlayerStatManager.cs.

4.151.4.4 WarmthRate

**WarmthRateManager** PlayerStatManager.WarmthRate [get]

Gets the warmth rate.

The warmth rate.

Definition at line 61 of file PlayerStatManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Player/PlayerStatManager.cs

4.152 PlayerTools Class Reference

Used for keeping track of implementation of tools in the game scene.

Collaboration diagram for PlayerTools:

```
<table>
<thead>
<tr>
<th>PlayerTools</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ HasEquipped</td>
</tr>
<tr>
<td>+ EquippedTool</td>
</tr>
<tr>
<td>+ PlayerTools()</td>
</tr>
<tr>
<td>+ GetToolByBaseItem()</td>
</tr>
</tbody>
</table>
```
Public Member Functions

- **PlayerTools (Tool[] possibleToolsList)**
  
  Create a **PlayerTools** object.

- **Tool GetToolByBaseItem (BaseItem item)**
  
  Iterates through tools and checks for an implementation of the **BaseItem**.

Properties

- **bool HasEquipped [get]**
  
  Returns true if the player has a tool equipped (EquippedTool is not null).

- **Tool EquippedTool [get]**
  
  The tool equipped by the player.

4.152.1 Detailed Description

Used for keeping track of implementation of tools in the game scene.

Definition at line 7 of file PlayerTools.cs.

4.152.2 Constructor & Destructor Documentation

4.152.2.1 PlayerTools()

```csharp
PlayerTools.PlayerTools ( Tool[] possibleToolsList )
```

Create a **PlayerTools** object.

Parameters

- **possibleToolsList** The list of Tools in the scene.

Definition at line 16 of file PlayerTools.cs.

4.152.3 Member Function Documentation

4.152.3.1 GetToolByBaseItem()

```csharp
Tool PlayerTools.GetToolByBaseItem ( BaseItem item )
```

Iterates through tools and checks for an implementation of the **BaseItem**.
Parameters

| item | The BaseItem to find. |

Returns

The implementation, or null if the item has not been implemented.

Definition at line 69 of file PlayerTools.cs.

4.152.4 Property Documentation

4.152.4.1 EquippedTool

Tool PlayerTools.EquippedTool [get]

The tool equipped by the player.

Definition at line 49 of file PlayerTools.cs.

4.152.4.2 HasEquipped

bool PlayerTools.HasEquipped [get]

Returns true if the player has a tool equipped (EquippedTool is not null).

Definition at line 38 of file PlayerTools.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Player/PlayerTools.cs

4.153 PressureSystem Class Reference

Collaboration diagram for PressureSystem:

```
<table>
<thead>
<tr>
<th>PressureSystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ Position</td>
</tr>
<tr>
<td>+ Pressure</td>
</tr>
<tr>
<td>+ IsHighPressure</td>
</tr>
</tbody>
</table>
```

Generated by Doxygen
Public Attributes

- Vector2 Position
- float Pressure
- bool IsHighPressure

4.153.1 Detailed Description

Definition at line 3 of file PressureSystem.cs.

4.153.2 Member Data Documentation

4.153.2.1 IsHighPressure

bool PressureSystem.IsHighPressure

Definition at line 7 of file PressureSystem.cs.

4.153.2.2 Position

Vector2 PressureSystem.Position

Definition at line 5 of file PressureSystem.cs.

4.153.2.3 Pressure

float PressureSystem.Pressure

Definition at line 6 of file PressureSystem.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/PressureSystem.cs
4.154 PressureSystems Class Reference

Collaboration diagram for PressureSystems:

```
PressureSystems
+ CityBounds
+ LocalPressureSystems
+ PressureSystems()
+ UpdatePressureSystem()
+ GetClosestPressureSystem()
+ LowPressureSystems()
```

Public Member Functions

- **PressureSystems (CityBoundaries bounds)**
  
  Initializes a new instance of the PressureSystems class.

- **void UpdatePressureSystem ()**
  
  Subscribed to clock delegate for every second to update. This method will move pressure systems around the world.

- **PressureSystem GetClosestPressureSystem (Vector2 position)**
  
  Gets the closest pressure system.

- **List< PressureSystem > LowPressureSystems ()**
  
  Return a list of low pressure systems.

Properties

- **Bounds CityBounds [get]**
  
  Get and set the city bounds that the pressure system can move in.

- **List< PressureSystem > LocalPressureSystems [get]**
  
  Gets the local pressure systems.

4.154.1 Detailed Description

Definition at line 19 of file PressureSystems.cs.

4.154.2 Constructor & Destructor Documentation

4.154.2.1 PressureSystems()

PressureSystems.PressureSystems ( 
  CityBoundaries bounds )

Initializes a new instance of the PressureSystems class.
Class Documentation

Parameters

- **bounds** : Bounds.

Definition at line 52 of file PressureSystems.cs.

### 4.154.3 Member Function Documentation

#### 4.154.3.1 GetClosestPressureSystem()

```csharp
PressureSystem PressureSystems.GetClosestPressureSystem (Vector2 position)
```

Gets the closest pressure system.

**Returns**

The closest pressure system.

**Parameters**

- **position** : Position.

Definition at line 144 of file PressureSystems.cs.

#### 4.154.3.2 LowPressureSystems()

```csharp
List<PressureSystem> PressureSystems.LowPressureSystems()
```

Return a list of low pressure systems.

**Returns**

The pressure systems.

Definition at line 169 of file PressureSystems.cs.

#### 4.154.3.3 UpdatePressureSystem()

```csharp
void PressureSystems.UpdatePressureSystem()
```

Subscribed to clock delegate for every second to update. This method will move pressure systems around the world.

Definition at line 75 of file PressureSystems.cs.
4.154.4 Property Documentation

4.154.4.1 CityBounds

    Bounds PressureSystems.CityBounds [get]

Get and set the city bounds that the pressure system can move in
The city bounds.
Definition at line 33 of file PressureSystems.cs.

4.154.4.2 LocalPressureSystems

    List<PressureSystem> PressureSystems.LocalPressureSystems [get]

Gets the local pressure systems.
The local pressure systems.
Definition at line 43 of file PressureSystems.cs.
The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/PressureSystems.cs

4.155 PressureSystemVisualization Class Reference

Inheritance diagram for PressureSystemVisualization:
4.155.1 Detailed Description

Definition at line 4 of file PressureSystemVisualization.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/GUI/PressureSystemVisualization.cs

4.156 ProceduralBuilding Class Reference

A building created procedurally.
Inheritance diagram for ProceduralBuilding:
Collaboration diagram for ProceduralBuilding:

```
Collaboration diagram for ProceduralBuilding:

- ProceduralBuilding
- Building
- ProceduralBuildingCreator

Public Member Functions
- ProceduralBuilding (Transform parent, Vector3 position, int seed, int numberOfFloors, DistrictConfiguration configuration)
  Creates an instance of the ProceduralBuilding class.
- override void Load()
  Loads the instance of the building into the scene.

Static Public Attributes
- static ProceduralBuildingCreator Generator
```

Generated by Doxygen
Procedural building generator.

- static Vector3 BuildingScale
  Scale of generated buildings.

Properties

- int Seed [get]
  The seed passed to the procedural generator. Should be unique to each building.

- int NumberOfFloors [get]
  Height to generate the building.

- DistrictConfiguration Configuration [get]
  The district configuration used to construct the building.

4.156.1 Detailed Description

A building created procedurally.

Definition at line 7 of file ProceduralBuilding.cs.

4.156.2 Constructor & Destructor Documentation

4.156.2.1 ProceduralBuilding()

ProceduralBuilding.ProceduralBuilding ( 
  Transform parent, 
  Vector3 position, 
  int seed, 
  int numberOfFloors, 
  DistrictConfiguration configuration )

Creates an instance of the ProceduralBuilding class.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>parent</td>
<td>The gameobject the building should be parented to.</td>
</tr>
<tr>
<td>position</td>
<td>The position of the building.</td>
</tr>
<tr>
<td>seed</td>
<td>The unique seed used to generate the building.</td>
</tr>
<tr>
<td>numberOfFloors</td>
<td>The number of floors to generate.</td>
</tr>
<tr>
<td>configuration</td>
<td>The district configuration used to construct the building.</td>
</tr>
</tbody>
</table>

Definition at line 27 of file ProceduralBuilding.cs.

4.156.3 Member Function Documentation

Generated by Doxygen
4.156.1 Load()

override void ProceduralBuilding.Load() [virtual]

Loads the instance of the building into the scene.

Implements Building.

Definition at line 69 of file ProceduralBuilding.cs.

4.156.4 Member Data Documentation

4.156.4.1 BuildingScale

Vector3 ProceduralBuilding.BuildingScale [static]

Scale of generated buildings.

Definition at line 17 of file ProceduralBuilding.cs.

4.156.4.2 Generator

ProceduralBuildingCreator ProceduralBuilding.Generator [static]

Procedural building generator.

Definition at line 12 of file ProceduralBuilding.cs.

4.156.5 Property Documentation

4.156.5.1 Configuration

DistrictConfiguration ProceduralBuilding.Configuration [get]

The district configuration used to construct the building.

Definition at line 60 of file ProceduralBuilding.cs.
4.156.2  NumberOfFloors

```csharp
int ProceduralBuilding.NumberOfFloors [get]
```

Height to generate the building.

Definition at line 51 of file ProceduralBuilding.cs.

4.156.3  Seed

```csharp
int ProceduralBuilding.Seed [get]
```

The seed passed to the procedural generator. Should be unique to each building.

Definition at line 42 of file ProceduralBuilding.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/ProceduralBuilding.cs

4.157  ProceduralBuildingAttachment Class Reference

Used to define important parts of building attachments that are used during building creation and item placement.

Inheritance diagram for ProceduralBuildingAttachment:
Collaboration diagram for ProceduralBuildingAttachment:

Public Attributes

- int AttachmentPoint
- ProceduralBuildingRoof HasRoof
  
  The roof the attachment comes with, if any.

Properties

- bool AddWindows [get]
  
  Does the attachment take windows

- Transform [] WindowPoints [get]
  
  The points on the attachment that windows can be placed at.

- Transform RoofLocation [get]
  
  The empty gameobject where a roof gets placed.

4.157.1 Detailed Description

Used to define important parts of building attachments that are used during building creation and item placement.

Definition at line 7 of file ProceduralBuildingAttachment.cs.
4.157.2 Member Data Documentation

4.157.2.1 AttachmentPoint

```cpp
int ProceduralBuildingAttachment.AttachmentPoint
```

Definition at line 10 of file ProceduralBuildingAttachment.cs.

4.157.2.2 HasRoof

```cpp
ProceduralBuildingRoof ProceduralBuildingAttachment.HasRoof
```

The roof the attachment comes with, if any.

Definition at line 44 of file ProceduralBuildingAttachment.cs.

4.157.3 Property Documentation

4.157.3.1 AddWindows

```cpp
bool ProceduralBuildingAttachment.AddWindows [get]
```

Does the attachment take windows

Definition at line 19 of file ProceduralBuildingAttachment.cs.

4.157.3.2 RoofLocation

```cpp
Transform ProceduralBuildingAttachment.RoofLocation [get]
```

The empty gameobject where a roof gets placed.

Definition at line 52 of file ProceduralBuildingAttachment.cs.
4.157.3.3 WindowPoints

Transform [[ProceduralBuildingAttachment.WindowPoints [get]]

The points on the attachment that windows can be placed at.

Definition at line 33 of file ProceduralBuildingAttachment.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingAttachment.cs

4.158 ProceduralBuildingBase Class Reference

Defines parts of the building base that are needed for procedural building generation.

Inheritance diagram for ProceduralBuildingBase:

![Inheritance diagram for ProceduralBuildingBase](image)
Collaboration diagram for ProceduralBuildingBase:

![Collaboration Diagram]

Properties

- **HeightType** `HeightType` [get]
  
  The way that this building handles growing vertically.

- **Transform[]** `AttachmentPoints` [get]
  
  The points on the base that can handle having an attachment added to them.

- **bool** `AddWindows` [get]
  
  Does the base need windows added to it.

- **Transform[]** `WindowPoints` [get]
  
  The points on the base that can take windows.

- **Transform** `RoofLocation` [get]
  
  The location of the empty gameobject where the roof is placed.

- **ProceduralBuildingRoof** `HasRoof` [get]
  
  If the building has a specific roof it needs, it's set here.

- **GameObject** `StackableObject` [get]
  
  If the object is of type stackable, we need to know which part of it to duplicate and stack.

4.158.1 Detailed Description

Defines parts of the building base that are needed for procedural building generation.

Definition at line 7 of file ProceduralBuildingBase.cs.
4.158.2  Property Documentation

4.158.2.1  AddWindows

    bool ProceduralBuildingBase.AddWindows  [get]

Does the base need windows added to it.

Definition at line 45 of file ProceduralBuildingBase.cs.

4.158.2.2  AttachmentPoints

    Transform [] ProceduralBuildingBase.AttachmentPoints  [get]

The points on the base that can handle having an attachment added to them.

Definition at line 32 of file ProceduralBuildingBase.cs.

4.158.2.3  HasRoof

    ProceduralBuildingRoof ProceduralBuildingBase.HasRoof  [get]

If the building has a specific roof it needs, it's set here.

Definition at line 86 of file ProceduralBuildingBase.cs.

4.158.2.4  HeightType

    HeightType ProceduralBuildingBase.HeightType  [get]

The way that this building handles growing vertically.

Definition at line 19 of file ProceduralBuildingBase.cs.

4.158.2.5  RoofLocation

    Transform ProceduralBuildingBase.RoofLocation  [get]

The location of the empty gameobject where the roof is placed.

Definition at line 72 of file ProceduralBuildingBase.cs.
4.158.2.6 StackableObject

GameObject ProceduralBuildingBase.StackableObject [get]

If the object is of type stackable, we need to know which part of it to duplicate and stack.

Definition at line 100 of file ProceduralBuildingBase.cs.

4.158.2.7 WindowPoints

Transform [] ProceduralBuildingBase.WindowPoints [get]

The points on the base that can take windows.

Definition at line 59 of file ProceduralBuildingBase.cs.

The documentation for this class was generated from the following file:

* Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingBase.cs

4.159 DistrictConfiguration.ProceduralBuildingBasesSizeXDefinition Struct Reference

Holds all of the bases and roofs for each size of building.

Collaboration diagram for DistrictConfiguration.ProceduralBuildingBasesSizeXDefinition:

```
DistrictConfiguration.Procedural
BuildingBasesSizeXDefinition
  + DistrictSizeXBases
  + DistrictSizeXRoofs
  + GetTypeLengthByType()
  + GetRandomEntryByType()
```

Public Member Functions

- int GetTypeLengthByType (BuildingParts partType)

  Gets the lengths of this district sizes bases or roofs.

- buildingIndex GetRandomEntryByType (BaseSize size, BuildingParts partType)

  Gets a random entry in the district's building array by its type. Returns a struct of that buildings size, and position in the enum.
Properties

- **ProceduralBuildingBase ** DistrictSizeXBases [get]
  
  The building bases associated with the size.

- **ProceduralBuildingRoof ** DistrictSizeXRoofs [get]
  
  The building roofs associated with the size.

### 4.1.59.1 Detailed Description

Holds all of the bases and roofs for each size of building.

Definition at line 110 of file DistrictDefinition.cs.

### 4.1.59.2 Member Function Documentation

#### 4.1.59.2.1 GetRandomEntryByType()

```csharp
buildingIndex DistrictConfiguration.ProceduralBuildingBasesSizeXDefinition.GetRandomEntryByType(BaseSize size, BuildingParts partType )
```

Gets a random entry in the district's building array by its type. Returns a struct of that buildings size, and position in the enum.

Definition at line 160 of file DistrictDefinition.cs.

#### 4.1.59.2.2 GetTypeLengthByType()

```csharp
int DistrictConfiguration.ProceduralBuildingBasesSizeXDefinition.GetTypeLengthByType(BuildingParts partType )
```

Gets the lengths of this district sizes bases or roofs.

Definition at line 141 of file DistrictDefinition.cs.

### 4.1.59.3 Property Documentation
4.159.3.1 DistrictSizeXBases

ProceduralBuildingBase [] DistrictConfiguration.ProceduralBuildingBasesSizeXDefinition
DistrictSizeXBases [get]

The building bases associated with the size.
Definition at line 118 of file DistrictDefinition.cs.

4.159.3.2 DistrictSizeXRoofs

ProceduralBuildingRoof [] DistrictConfiguration.ProceduralBuildingBasesSizeXDefinition
DistrictSizeXRoofs [get]

The building roofs associated with the size.
Definition at line 131 of file DistrictDefinition.cs.

The documentation for this struct was generated from the following file:

- Assets/Scripts/City/DistrictDefinition.cs

4.160 ProceduralBuildingCreator Class Reference

Defines all the buildings for each district and allows for the creation of buildings in those districts.

Inheritance diagram for ProceduralBuildingCreator:
Public Member Functions

- void CreateTestBuilding()
- ProceduralBuildingInstance CreateBuilding(DistrictConfiguration district, BaseSize size, float attatchmentPercentage, int storiesTall)

Populates a new proceduralbuilding class and spawns it in the world.

Public Attributes

- DistrictConfiguration TestDistrict
- BaseSize TestSize
- int TestStoriesTall
- float TestAttatchmentPercentage
4.160.1 Detailed Description

Defines all the buildings for each district and allows for the creation of buildings in those districts.

Definition at line 34 of file ProceduralBuildingCreator.cs.

4.160.2 Member Function Documentation

4.160.2.1 CreateBuilding()

```
ProceduralBuildingInstance ProceduralBuildingCreator.CreateBuilding (  
        DistrictConfiguration district,  
        BaseSize size,  
        float attachmentPercentage,  
        int storiesTall )
```

Populates a new procedural building class and spawns it in the world.

Definition at line 133 of file ProceduralBuildingCreator.cs.

4.160.2.2 CreateTestBuilding()

```
void ProceduralBuildingCreator.CreateTestBuilding ( )
```

Definition at line 83 of file ProceduralBuildingCreator.cs.

4.160.3 Member Data Documentation

4.160.3.1 TestAttachmentPercentage

```
float ProceduralBuildingCreator.TestAttachmentPercentage
```

Definition at line 80 of file ProceduralBuildingCreator.cs.

4.160.3.2 TestDistrict

```
DistrictConfiguration ProceduralBuildingCreator.TestDistrict
```

Definition at line 77 of file ProceduralBuildingCreator.cs.
4.160.3.3 TestSize

**BaseSize** ProceduralBuildingCreator.TestSize

Definition at line 78 of file ProceduralBuildingCreator.cs.

4.160.3.4 TestStoriesTall

```csharp
int ProceduralBuildingCreator.TestStoriesTall
```

Definition at line 79 of file ProceduralBuildingCreator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingCreator.cs

4.161 ProceduralBuildingInstance Class Reference

This class holds all the info that comes with making a procedural building

Inheritance diagram for ProceduralBuildingInstance:
Collaboration diagram for ProceduralBuildingInstance:

```
MonoBehaviour

ProceduralBuildingBase
+ HeightType
+ AttachmentPoints
+ AddWindows
+ WindowPoints
+ RoofLocation
+ HasRoof
+ StackableObject

ProceduralBuildingWindow
+ Width

ProceduralBuildingAttachment
+ AttachmentPoint
+ AddWindows
+ WindowPoints
+ RoofLocation

ProceduralBuildingRoof
+ CanRotate

ProceduralBuildingInstance
+ WindowSpacing
+ BuildingMaterial
+ GetRoofs()
+ GetRootMeshes()
+ BuildingBase
+ Windows
+ BuildingAttachments
+ BuildingRoof
+ HasRoof

Public Member Functions

- GameObject [] GetRoofs ()
  Used during item placement, returns all of the roof objects
- Mesh [] GetRootMeshes ()
  Used during item placement, returns all of the roof object meshes.

Public Attributes

- ProceduralBuildingBase BuildingBase
- ProceduralBuildingWindow Windows
- float WindowSpacing
- ProceduralBuildingRoof BuildingRoof
- ProceduralBuildingAttachment [] BuildingAttachments
- Material BuildingMaterial

4.161.1 Detailed Description

This class holds all the info that comes with making a procedural building

Definition at line 7 of file ProceduralBuildingInstance.cs.
```
4.161.2 Member Function Documentation

4.161.2.1 GetRoofMeshes()

Mesh [] ProceduralBuildingInstance.GetRoofMeshes()

Used during item placement, returns all of the roof object meshes.

Returns

the meshes of all the roofs.

Definition at line 40 of file ProceduralBuildingInstance.cs.

4.161.2.2 GetRoofs()

GameObject [] ProceduralBuildingInstance.GetRoofs()

Used during item placement, returns all of the roof objects

Returns

Returns all of the roofs as gameobjects.

Definition at line 20 of file ProceduralBuildingInstance.cs.

4.161.3 Member Data Documentation

4.161.3.1 BuildingAttachments

ProceduralBuildingAttachment [] ProceduralBuildingInstance.BuildingAttachments

Definition at line 13 of file ProceduralBuildingInstance.cs.

4.161.3.2 BuildingBase

ProceduralBuildingBase ProceduralBuildingInstance.BuildingBase

Definition at line 9 of file ProceduralBuildingInstance.cs.
4.161.3.3 BuildingMaterial

Material ProceduralBuildingInstance.BuildingMaterial

Definition at line 14 of file ProceduralBuildingInstance.cs.

4.161.3.4 BuildingRoof

ProceduralBuildingRoof ProceduralBuildingInstance.BuildingRoof

Definition at line 12 of file ProceduralBuildingInstance.cs.

4.161.3.5 Windows

ProceduralBuildingWindow ProceduralBuildingInstance.Windows

Definition at line 10 of file ProceduralBuildingInstance.cs.

4.161.3.6 WindowSpacing

float ProceduralBuildingInstance.WindowSpacing

Definition at line 11 of file ProceduralBuildingInstance.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingInstance.cs
4.162 ProceduralBuildingRoof Class Reference

Attached to the roof prefabs. Holds roof specific information needed when creating buildings.

Inheritance diagram for ProceduralBuildingRoof:

```
MonoBehaviour

ProceduralBuildingRoof
  + CanRotate
```

Collaboration diagram for ProceduralBuildingRoof:

```
MonoBehaviour

ProceduralBuildingRoof
  + CanRotate
```

Properties

- bool **CanRotate** [get]
  
  *If the roof can be rotated at 90 degree intervals when it's created.*
4.162.1 Detailed Description

Attached to the roof prefabs. Holds roof specific information needed when creating buildings.

Definition at line 7 of file ProceduralBuildingRoof.cs.

4.162.2 Property Documentation

4.162.2.1 CanRotate

bool ProceduralBuildingRoof.CanRotate [get]

If the roof can be rotated at 90 degree intervals when it's created.

Definition at line 15 of file ProceduralBuildingRoof.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingRoof.cs

4.163 ProceduralBuildingWindow Class Reference

Attached to the window prefabs. Holds window specific information needed when creating buildings.

Inheritance diagram for ProceduralBuildingWindow:
Collaboration diagram for ProceduralBuildingWindow:

```
MonoBehaviour

ProceduralBuildingWindow
+ Width
```

Properties

- float `Width` [get]
  
  The width of this window.

4.163.1 Detailed Description

Attached to the window prefabs. Holds window specific information needed when creating buildings.

Definition at line 7 of file ProceduralBuildingWindow.cs.

4.163.2 Property Documentation

4.163.2.1 Width

float `ProceduralBuildingWindow.Width` [get]

The width of this window.

Definition at line 15 of file ProceduralBuildingWindow.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingWindow.cs
Inheritance diagram for Radio:

```
MonoBehaviour

Radio
+ MusicDefaultPath
+ MysteryDefaultPath
+ StaticDefaultPath
+ lowMusic
+ highMusic
+ lowMystery
+ highMystery
+ lowWeather
+ highWeather
+ CurrentChannel
+ Power()
+ addToCarousel()
+ SetChannel()
+ ChangeChannel()
+ OnChannelClick()
+ GetWeatherAnnouncement()
+ GetMysteryAnnouncement()
```
Public Member Functions

- void **Power** ()
  *Turns radio on and off.*

- void **AddToCarousel** ([RadioChannel](#) channel, string soundEvent)
  *Adds sound event to carousel.*

- void **SetChannel** ([RadioChannel](#) channel)
  *Set new selected channel.*

- void **ChangeChannel** (float knobRotation)
  *Change the radio channel based on dial position.*

- void **OnChannelClick** ()
  *Flip through the channels.*

- string **GetWeatherAnnouncement** (float windSpeed, float temperature)
  *Creates string based on windSpeed and temperature (and eventually amount of rainfall)*

- string **GetMysteryAnnouncement** ()
  *Gets the mystery announcement.*
Public Attributes

- string MusicDefaultPath = "event://Radio/Music/Soulja_Boy"
- string MysteryDefaultPath = "event://Radio/Mystery/Mystery1"
- string StaticDefaultPath = "event://Radio/Static/Basic_Static"
- float lowMusic
- float highMusic
- float lowMystery
- float highMystery
- float lowWeather
- float highWeather

4.164.1 Detailed Description

Definition at line 10 of file Radio.cs.

4.164.2 Member Function Documentation

4.164.2.1 AddToCarousel()

void Radio.AddToCarousel (  
    RadioChannel channel,  
    string soundEvent )

Adds sound event to carousel.

Parameters

<table>
<thead>
<tr>
<th>channel</th>
<th>Channel.</th>
</tr>
</thead>
<tbody>
<tr>
<td>soundEvent</td>
<td>Sound event to be added.</td>
</tr>
</tbody>
</table>

Definition at line 268 of file Radio.cs.

4.164.2.2 ChangeChannel()

void Radio.ChangeChannel (  
    float knobRotation )

Change the radio channel based on dial position.

Parameters

<table>
<thead>
<tr>
<th>knobRotation</th>
</tr>
</thead>
</table>

Generated by Doxygen
4.164.2.3 GetMysteryAnnouncement()

string Radio.GetMysteryAnnouncement ( )

Gets the mystery announcement.

Definition at line 471 of file Radio.cs.

4.164.2.4 GetWeatherAnnouncement()

string Radio.GetWeatherAnnouncement ( 
    float windSpeed, 
    float temperature )

Creates string based on windSpeed and temperature (and eventually amount of rainfall)

Parameters

| windSpeed |
| temperature |

Returns

Definition at line 458 of file Radio.cs.

4.164.2.5 OnChannelClick()

void Radio.OnChannelClick ( )

Flip through the channels.

Definition at line 432 of file Radio.cs.

4.164.2.6 Power()

void Radio.Power ( )

Turns radio on and off.

Definition at line 242 of file Radio.cs.
4.164.2.7 SetChannel()

```csharp
void Radio.SetChannel (  
    RadioChannel channel )
```

Set new selected channel.

Parameters

- **channel**

Definition at line 357 of file Radio.cs.

4.164.3 Member Data Documentation

4.164.3.1 highMusic

```csharp
float Radio.highMusic
```

Definition at line 64 of file Radio.cs.

4.164.3.2 highMystery

```csharp
float Radio.highMystery
```

Definition at line 68 of file Radio.cs.

4.164.3.3 highWeather

```csharp
float Radio.highWeather
```

Definition at line 72 of file Radio.cs.

4.164.3.4 lowMusic

```csharp
float Radio.lowMusic
```

Definition at line 62 of file Radio.cs.
4.164.3.5 lowMystery

float Radio.lowMystery

Definition at line 66 of file Radio.cs.

4.164.3.6 lowWeather

float Radio.lowWeather

Definition at line 70 of file Radio.cs.

4.164.3.7 MusicDefaultPath

string Radio.MusicDefaultPath = "event:/Radio/Music/Soulja_Boy"

Definition at line 35 of file Radio.cs.

4.164.3.8 MysteryDefaultPath

string Radio.MysteryDefaultPath = "event:/Radio/Mystery/Mystery1"

Definition at line 37 of file Radio.cs.

4.164.3.9 StaticDefaultPath

string Radio.StaticDefaultPath = "event:/Radio/Static/Basic_Static"

Definition at line 39 of file Radio.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Radio/Radio.cs
4.165 RadioButton Class Reference

Inheritance diagram for RadioButton:

[Diagram showing inheritance structure]

Collaboration diagram for RadioButton:

[Diagram showing collaboration structure]

4.165.1 Detailed Description

Definition at line 7 of file RadioButton.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Radio/RadioButton.cs
4.166 RaftCategory Class Reference

Inheritance diagram for RaftCategory:
Collaboration diagram for RaftCategory:

Public Member Functions

- **override ItemCategory GetDuplicate()**
  
  Creates a copy of this raft category.

- **override void ReadyCategory()**
  
  Readies the item category by adding the attributes and actions it can complete.

- **void SetDown()**
  
  Sets down the raft in the world. Drops it where the player stands.
Properties

• float Speed [get, set]

  Gets or sets the speed of the item.

Additional Inherited Members

4.166.1 Detailed Description

Definition at line 6 of file RaftCategory.cs.

4.166.2 Member Function Documentation

4.166.2.1 GetDuplicate()

override ItemCategory RaftCategory.GetDuplicate ( ) [virtual]

Creates a copy of this raft category.

Returns

  The duplicate.

Reimplemented from ItemCategory.

Definition at line 31 of file RaftCategory.cs.

4.166.2.2 ReadyCategory()

override void RaftCategory.ReadyCategory ( ) [virtual]

Readies the item category by adding the attributes and actions it can complete.

Reimplemented from ItemCategory.

Definition at line 53 of file RaftCategory.cs.

4.166.2.3 SetDown()

void RaftCategory.SetDown ( )

Sets down the raft in the world. Drops it where the player stands.

Definition at line 64 of file RaftCategory.cs.
4.166.3 Property Documentation

4.166.3.1 Speed

float RaftCategory.Speed [get], [set]

Gets or sets the speed of the item.

The content of the water.

Definition at line 13 of file RaftCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/RaftCategory.cs

4.167 RaftInteractable Class Reference

Inheritance diagram for RaftInteractable:
Collaboration diagram for RaftInteractable:

Public Member Functions

- override void **SetUp** ()
  
  Sets action for raft as board raft

- void **BoardRaft** ()
  
  *Player* boards the raft and assumes raft controls until the player dismounts.

- void **DisembarkRaft** ()
  
  *Player* disembarks the raft and resumes player movement.

Properties

- **RaftCategory Raft** [get, set]
  
  Gets or sets the raft category object.

Additional Inherited Members

4.167.1 Detailed Description

Definition at line 4 of file RaftInteractable.cs.
4.167 RaftInteractable Class Reference

4.167.2 Member Function Documentation

4.167.2.1 BoardRaft()

```csharp
void RaftInteractable.BoardRaft ( )
```

**Player** boards the raft and assumes raft controls until the player dismounts.

Definition at line 45 of file RaftInteractable.cs.

4.167.2.2 DisembarkRaft()

```csharp
void RaftInteractable.DisembarkRaft ( )
```

**Player** disembarks the raft and resumes player movement.

**Parameters**

- `raftMovement`

Definition at line 64 of file RaftInteractable.cs.

4.167.2.3 SetUp()

```csharp
override void RaftInteractable.SetUp ( ) [virtual]
```

Sets action for raft as board raft

Reimplemented from **InteractableObject**.

Definition at line 21 of file RaftInteractable.cs.

4.167.3 Property Documentation

4.167.3.1 Raft

**RaftCategory** RaftInteractable.Raft [get], [set]

Gets or sets the raft category object.

Definition at line 13 of file RaftInteractable.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/RaftInteractable.cs

Generated by Doxygen
Inheritance diagram for RaftMovement:

```
MonoBehaviour

Movement
+ Speed
+ CurrentFallDammage
  # RigidBody
  # AccumulatedFallDamage
  # climbHeight
+ Idle()
+ Move()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ OnStateEnter()
+ OnStateExit()
+ GetRaycastHeight()

RaftMovement
+ IsActive
+ BoardRaftText
+ DisembarkRaftText
+ PlayerAnimator
+ PlayerStandHeight
+ Idle()
+ Move()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ OnStateEnter()
+ OnStateExit()
+ GetRaycastHeight()
+ SetMaxSpeed()
```
Public Member Functions

- override void Idle (Animator playerAnimator)
  
  *Runs the raft idle animation.*

- override void Move (Vector3 direction, bool sprinting, Animator playerAnimator)
  
  *Applies a force the raft in the specified direction.*

- override void Jump (Animator playerAnimator)
Doesn't do anything on boats for now.

- **override void Climb (Animator playerAnimator)**
  Doesn't do anything on boats for now

- **override float GetClimbHeight ()**
  The height the player can climb while in this movement state (which is 0!)

- **override void OnStateEnter ()**
  Called when the player enters the state.

- **override void OnStateExit ()**
  Called when the player exits the state.

- **override float GetRaycastHeight ()**
  UNUSED The height of the climbing raycast while in this movement state

- **void SetMaxSpeed (float speed)**

### Public Attributes

- **bool IsActive = false**
  Indicates whether the raft is active. Currently not in use.

- **string BoardRaftText**

- **string DisembarkRaftText**

- **Animator PlayerAnimator**

### Properties

- **float PlayerStandHeight [get]**

### Additional Inherited Members

#### 4.168.1 Detailed Description

Definition at line 5 of file RaftMovement.cs.

#### 4.168.2 Member Function Documentation

##### 4.168.2.1 Climb()

```csharp
override void RaftMovement.Climb (Animator playerAnimator) [virtual]
```

Doesn't do anything on boats for now

Implements **Movement**.

Definition at line 148 of file RaftMovement.cs.
4.168.2.2 GetClimbHeight()

```csharp
override float RaftMovement.GetClimbHeight () [virtual]
```
The height the player can climb while in this movement state (which is 0!)
Implements Movement.
Definition at line 156 of file RaftMovement.cs.

4.168.2.3 GetRaycastHeight()

```csharp
override float RaftMovement.GetRaycastHeight () [virtual]
```
UNUSED The height of the climbing raycast while in this movement state
Implements Movement.
Definition at line 188 of file RaftMovement.cs.

4.168.2.4 Idle()

```csharp
override void RaftMovement.Idle ( Animator playerAnimator ) [virtual]
```
Runs the raft idle animation.
Implements Movement.
Definition at line 98 of file RaftMovement.cs.

4.168.2.5 Jump()

```csharp
override void RaftMovement.Jump ( Animator playerAnimator ) [virtual]
```
Doesn't do anything on boats for now.
Implements Movement.
Definition at line 140 of file RaftMovement.cs.

4.168.2.6 Move()

```csharp
override void RaftMovement.Move ( Vector3 direction,
bool sprinting,
Animator playerAnimator ) [virtual]
```
Applies a force the raft in the specified direction.
Parameters

- direction

Implements Movement.

Definition at line 130 of file RaftMovement.cs.

4.168.2.7 OnStateEnter()

```csharp
override void RaftMovement.OnStateEnter () [virtual]
```

Called when the player enters the state.

Implements Movement.

Definition at line 164 of file RaftMovement.cs.

4.168.2.8 OnStateExit()

```csharp
override void RaftMovement.OnStateExit () [virtual]
```

Called when the player exits the state.

Implements Movement.

Definition at line 179 of file RaftMovement.cs.

4.168.2.9 SetMaxSpeed()

```csharp
void RaftMovement.SetMaxSpeed ( float speed )
```

Definition at line 193 of file RaftMovement.cs.

4.168.3 Member Data Documentation
4.168 RaftMovement Class Reference

4.168.3.1 BoardRaftText

string RaftMovement.BoardRaftText

Definition at line 33 of file RaftMovement.cs.

4.168.3.2 DisembarkRaftText

string RaftMovement.DisembarkRaftText

Definition at line 34 of file RaftMovement.cs.

4.168.3.3 IsActive

bool RaftMovement.IsActive = false

Indicates whether the raft is active. Currently not in use.

Definition at line 13 of file RaftMovement.cs.

4.168.3.4 PlayerAnimator

Animator RaftMovement.PlayerAnimator

Definition at line 37 of file RaftMovement.cs.

4.168.4 Property Documentation

4.168.4.1 PlayerStandHeight

float RaftMovement.PlayerStandHeight [get]

Definition at line 26 of file RaftMovement.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/RaftMovement.cs
4.169 RaftWake Class Reference

Inheritance diagram for RaftWake:

![Inheritance Diagram](image)

Collaboration diagram for RaftWake:

![Collaboration Diagram](image)

4.169.1 Detailed Description

Definition at line 4 of file RaftWake.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/RaftWake.cs
4.170 RainController Class Reference

This class handles the rain effect that is present throughout the entire game.

Inheritance diagram for RainController:
Public Attributes

- bool UseCustomValues
- ParticleSystem MainRain
- ParticleSystem RainFog
- float FogStartThreshold = 50f

Properties

- float RainLevel [get, set]
- Vector2 WindVectorXZ [get, set]

4.170.1 Detailed Description

This class handles the rain effect that is present throughout the entire game.

CURRENT PROBLEMS Currently there seems to be an issue with Unity running out of memory and crashing when editing the particle system. I have plenty of ram to spare so I think something else is going on. People online have said it may have to do with sub-emitters so I'm going to forgoe them for now. There is still a sub emitter on RainMain you can turn on but it may make the system unstable.

Definition at line 17 of file RainController.cs.

4.170.2 Member Data Documentation

4.170.2.1 FogStartThreshold

float RainController.FogStartThreshold = 50f

Definition at line 55 of file RainController.cs.

4.170.2.2 MainRain

ParticleSystem RainController.MainRain

Definition at line 24 of file RainController.cs.

4.170.2.3 RainFog

ParticleSystem RainController.RainFog

Definition at line 27 of file RainController.cs.
4.170.2.4 UseCustomValues

bool RainController.UseCustomValues

Definition at line 20 of file RainController.cs.

4.170.3 Property Documentation

4.170.3.1 RainLevel

float RainController.RainLevel [get], [set]

Definition at line 31 of file RainController.cs.

4.170.3.2 WindVectorXZ

Vector2 RainController.WindVectorXZ [get], [set]

Definition at line 63 of file RainController.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Effects/RainController.cs

4.171 RandomUtility Class Reference

Collaboration diagram for RandomUtility:
Static Public Member Functions

- static Vector2 RandomVector2d (Vector2 min, Vector2 max)
  Generates a random Vector2
- static Vector3 RandomVector3d (Vector3 min, Vector3 max)
  Generates a random Vector3

Properties

- static bool RandomBool [get]
  Gets a random true or false.
- static float RandomPercent [get]
  Get a random float between 0 and 1.
- static float RandomHundredPercent [get]
  Get a random float between 0 and 100.
- static float RandomBinomial [get]
  Randoms binomial with higher likelihood of 0

4.171.1 Detailed Description

Definition at line 3 of file RandomUtility.cs.

4.171.2 Member Function Documentation

4.171.2.1 RandomVector2d()

static Vector2 RandomUtility.RandomVector2d (Vector2 min, Vector2 max) [static]

Generates a random Vector2

Returns

The vector.

Parameters

<table>
<thead>
<tr>
<th>min</th>
<th>Minimum.</th>
</tr>
</thead>
<tbody>
<tr>
<td>max</td>
<td>Max.</td>
</tr>
</tbody>
</table>

Definition at line 57 of file RandomUtility.cs.
4.171.2.2 RandomVector3d()

```csharp
static Vector3 RandomUtility.RandomVector3d (  
    Vector3 min,  
    Vector3 max ) [static]
```

Generates a random Vector3

Returns

The vector3d.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>min</code></td>
<td>Minimum.</td>
</tr>
<tr>
<td><code>max</code></td>
<td>Max.</td>
</tr>
</tbody>
</table>

Definition at line 68 of file RandomUtility.cs.

4.171.3 Property Documentation

4.171.3.1 RandomBinomial

```csharp
float RandomUtility.RandomBinomial [static], [get]
```

Randoms binomial with higher likelihood of 0

Returns

The binomial.

Definition at line 44 of file RandomUtility.cs.

4.171.3.2 RandomBool

```csharp
bool RandomUtility.RandomBool [static], [get]
```

Gets a random true or false.

true if random bool; otherwise, false.

Definition at line 10 of file RandomUtility.cs.
4.171.3.3 RandomHundredPercent

float RandomUtility.RandomHundredPercent [static], [get]

Get a random float between 0 and 100.

Definition at line 32 of file RandomUtility.cs.

4.171.3.4 RandomPercent

float RandomUtility.RandomPercent [static], [get]

Get a random float between 0 and 1.

Definition at line 21 of file RandomUtility.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/RandomUtility.cs

### 4.172 Recipe Class Reference

Class that handles storing the contents of recipes and checking whether or not the requirements are met.

Collaboration diagram for Recipe:

```
Recipe
+ ResourceRequirements
+ ToolRequirements
+ Tiered
+ RecipeName
+ StatsToCheck
+ Recipe()
+ AddRequirement()
+ CheckCompleted()
```

**Public Member Functions**

- **Recipe ()**
  
  Initializes a new instance of the Recipe class. Used by the yaml deserializer.

- **void AddRequirement (Requirement req)**
  
  Adds a requirement to the list of requirements.

- **bool CheckCompleted ()**
  
  Checks to see if the recipe is completely fulfilled.
Properties

- List< Requirement > ResourceRequirements [get, set]
  list of resource requirements for a recipe. Items under this category are used up during crafting.

- List< Requirement > ToolRequirements [get, set]
  List of tool requirements for a recipe. Items under this category are not used up during crafting.

- bool Tiered [get, set]
  Gets or sets a value indicating whether this Recipe is given a tier when crafted.

- string RecipeName [get, set]
  name of the recipe, generally the name of the item that it will create

- List< CraftingStat > StatsToCheck [get, set]
  Gets or sets the stats to check when determining the quality of the crafted item

4.172.1 Detailed Description

Class that handles storing the contents of recipes and checking whether or not the requirements are met.

Definition at line 7 of file Recipe.cs.

4.172.2 Constructor & Destructor Documentation

4.172.2.1 Recipe()

Recipe.Recipe ()

Initializes a new instance of the Recipe class. Used by the yaml deserializer.

Definition at line 62 of file Recipe.cs.

4.172.3 Member Function Documentation

4.172.3.1 AddRequirement()

void Recipe.AddRequirement ( Requirement req )

Adds a requirement to the list of requirements.

Parameters

| req | Req. |
4.172.3.2 CheckCompleted()

bool Recipe.CheckCompleted();

Checks to see if the recipe is completely fulfilled.

Returns

true, if all requirements completed, false otherwise.

4.172.4 Property Documentation

4.172.4.1 RecipeName

string Recipe.RecipeName [get], [set]

name of the recipe, generally the name of the item that it will create

The name of the recipe.

4.172.4.2 ResourceRequirements

List<Requirement> Recipe.ResourceRequirements [get], [set]

list of resource requirements for a recipe. Items under this category are used up during crafting.

The requirements.

Definition at line 70 of file Recipe.cs.

Definition at line 79 of file Recipe.cs.

Definition at line 44 of file Recipe.cs.

Definition at line 14 of file Recipe.cs.
4.172.3 StatsToCheck

List<CraftingStat> Recipe.StatsToCheck [get], [set]

Gets or sets the stats to check when determining the quality of the crafted item.

The stats to check.

Definition at line 54 of file Recipe.cs.

4.172.4 Tiered

bool Recipe.Tiered [get], [set]

Gets or sets a value indicating whether this Recipe is given a tier when crafted.

true if tiered; otherwise, false.

Definition at line 34 of file Recipe.cs.

4.172.5 ToolRequirements

List<Requirement> Recipe.ToolRequirements [get], [set]

List of tool requirements for a recipe. Items under this category are not used up during crafting.

The tool requirement.

Definition at line 24 of file Recipe.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/Recipe.cs
4.173 RecipeBookBehavior Class Reference

Inheritance diagram for RecipeBookBehavior:

```
MonoBehaviour

RecipeBookBehavior
+ RecipeFileName
  + LoadRecipes()
  + ResetPanel()
  + RefreshRecipes()
  + UnlockRecipe()
  + InsertRecipeButton()
```

Collaboration diagram for RecipeBookBehavior:

```
MonoBehaviour

RecipeBookBehavior
+ RecipeFileName
  + LoadRecipes()
  + ResetPanel()
  + RefreshRecipes()
  + UnlockRecipe()
  + InsertRecipeButton()
```

Public Member Functions
- void LoadRecipes ()
4.173 RecipeBookBehavior Class Reference

Loads the recipes from the yaml file into the recipe select panel.

- void ResetPanel ()
  Resets the panel so that all active crafting attempts are ended.
- void RefreshRecipes ()
  Refreshs the recipes.
- void UnlockRecipe (Recipe newRecipe)
  Unlocks a recipe. Adds it to the crafting ui.
- void InsertRecipeButton (RecipeButtonGUIBehavior newButton, List< RecipeButtonGUIBehavior > listToInsertInto)
  Inserts the recipe button into either the uncraftable or craftable section.

Public Attributes

- string RecipeFileName

4.173.1 Detailed Description

Definition at line 7 of file RecipeBookBehavior.cs.

4.173.2 Member Function Documentation

4.173.2.1 InsertRecipeButton() 

void RecipeBookBehavior.InsertRecipeButton ( 
  RecipeButtonGUIBehavior newButton, 
  List< RecipeButtonGUIBehavior > listToInsertInto )

Inserts the recipe button into either the uncraftable or craftable section.

Parameters

| newButton | New button. |
| listToInsertInto | List to insert into. |

Definition at line 195 of file RecipeBookBehavior.cs.

4.173.2.2 LoadRecipes() 

void RecipeBookBehavior.LoadRecipes ()

Loads the recipes from the yaml file into the recipe select panel.

Definition at line 36 of file RecipeBookBehavior.cs.
4.173.2.3 RefreshRecipes()

void RecipeBookBehavior.RefreshRecipes ()

Refreshs the recipes.

Definition at line 105 of file RecipeBookBehavior.cs.

4.173.2.4 ResetPanel()

void RecipeBookBehavior.ResetPanel ()

Resets the panel so that all active crafting attempts are ended.

Definition at line 71 of file RecipeBookBehavior.cs.

4.173.2.5 UnlockRecipe()

void RecipeBookBehavior.UnlockRecipe ( Recipe newRecipe )

Unlocks a recipe. Adds it to the crafting ui.

Parameters

- **newRecipe** New recipe.

Definition at line 172 of file RecipeBookBehavior.cs.

4.173.3 Member Data Documentation

4.173.3.1 RecipeFileName

string RecipeBookBehavior.RecipeFileName

Definition at line 16 of file RecipeBookBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/CraftingGUI/RecipeBookBehavior.cs
Inheritance diagram for RecipeButtonGUIBehavior:
Collaboration diagram for RecipeButtonGUIBehavior:

```
MonoBehaviour

RecipeButtonGUIBehavior
+ EnabledColor
+ DisabledColor
+ HighlightColor
+ SelectedColor
+ SelectedBackColor
+ AssociatedRecipe
+ Craftable
+ Highlight()
+ UnHighlight()
+ ShowRecipe()
+ Unselect()
+ SetUpButton()
```

Public Member Functions

- void `Highlight()`
  
  *Set the color of the text to the highlight color.*

- void `UnHighlight()`
  
  *Sets the color of the text to its unhighlighted color.*

- void `ShowRecipe()`
  
  *Shows the recipe in the crafting panel.*

- void `Unselect()`
  
  *Unselect this instance.*

- void `SetUpButton(Recipe recipe, bool recipePossible)`
  
  *Sets up button's color and links a recipe to it.*

Public Attributes

- Color `EnabledColor`
- Color `DisabledColor`
- Color `HighlightColor`
- Color `SelectedColor`
- Color `SelectedBackColor`
Properties

- **Recipe AssociatedRecipe** [get]
  
  *Gets the associated recipe.*

- **bool Craftable** [get, set]
  
  *Gets or sets a value indicating whether this RecipeButtonGUIBehavior is craftable.*

### 4.174.1 Detailed Description

Definition at line 5 of file RecipeButtonGUIBehavior.cs.

### 4.174.2 Member Function Documentation

#### 4.174.2.1 Highlight()

```csharp
void RecipeButtonGUIBehavior.Highlight ()
```

Set the color of the text to the highlight color.

Definition at line 77 of file RecipeButtonGUIBehavior.cs.

#### 4.174.2.2 SetUpButton()

```csharp
void RecipeButtonGUIBehavior.SetUpButton (Recipe recipe, bool recipePossible )
```

Sets up button's color and links a recipe to it.

**Parameters**

- **recipe** [Recipe]

Definition at line 132 of file RecipeButtonGUIBehavior.cs.

#### 4.174.2.3 ShowRecipe()

```csharp
void RecipeButtonGUIBehavior.ShowRecipe ()
```

Shows the recipe in the crafting panel.

Definition at line 93 of file RecipeButtonGUIBehavior.cs.
4.174.2.4 UnHighlight()

```csharp
void RecipeButtonGUIBehavior.UnHighlight()
```
Sets the color of the text to it's unhighlighted color.
Definition at line 85 of file RecipeButtonGUIBehavior.cs.

4.174.2.5 Unselect()

```csharp
void RecipeButtonGUIBehavior.Unselect()
```
Unselect this instance.
Definition at line 109 of file RecipeButtonGUIBehavior.cs.

4.174.3 Member Data Documentation

4.174.3.1 DisabledColor

```csharp
Color RecipeButtonGUIBehavior.DisabledColor
```
Definition at line 11 of file RecipeButtonGUIBehavior.cs.

4.174.3.2 EnabledColor

```csharp
Color RecipeButtonGUIBehavior.EnabledColor
```
Definition at line 8 of file RecipeButtonGUIBehavior.cs.

4.174.3.3 HighlightColor

```csharp
Color RecipeButtonGUIBehavior.HighlightColor
```
Definition at line 14 of file RecipeButtonGUIBehavior.cs.
4.174.3.4  SelectedBackColor

Color RecipeButtonGUIBehavior.SelectedBackColor

Definition at line 18 of file RecipeButtonGUIBehavior.cs.

4.174.3.5  SelectedColor

Color RecipeButtonGUIBehavior.SelectedColor

Definition at line 16 of file RecipeButtonGUIBehavior.cs.

4.174.4  Property Documentation

4.174.4.1  AssociatedRecipe

Recipe RecipeButtonGUIBehavior.AssociatedRecipe [get]

Gets the associated recipe.

The associated recipe.

Definition at line 33 of file RecipeButtonGUIBehavior.cs.

4.174.4.2  Craftable

bool RecipeButtonGUIBehavior.Craftable [get], [set]

Gets or sets a value indicating whether this RecipeButtonGUIBehavior is craftable.

true if craftable; otherwise, false.

Definition at line 43 of file RecipeButtonGUIBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/CraftingGUI/RecipeButtonGUIBehavior.cs
4.175 RecipePageBehavior Class Reference

Inheritance diagram for RecipePageBehavior:

```
MonoBehaviour

RecipePageBehavior
  + SetUpRecipePage()
  + OnBeginCraftingClick()
  + ResetCraftingPanel()
  + NextStep()
  + DisplayPossibleItems()
  + UpdateSelection()
  + ToggleIngredientSelectionPanel()
  + AddSelectedIngredients()
  + Craft()
  + EndCraftingAttempt()
  + HidePanel()
```
Collaboration diagram for RecipePageBehavior:

![Collaboration Diagram](image)

### Public Member Functions

- **void** `SetUpRecipePage(Recipe pageRecipe, RecipeButtonGUIBehavior currentSelectedRecipeButton, bool craftable)`
  
  *Sets up recipe page.*

- **void** `OnBeginCraftingClick()`
  
  *Raises the begin crafting click event.*

- **void** `ResetCraftingPanel()`
  
  *Resets the crafting panel.*

- **void** `NextStep()`
  
  *In the item selection UI panel, advances the step that the recipe is on. Displays the next set of items that are allowed to be used in the recipe.*

- **void** `DisplayPossibleItems(int stepInCraftingProcess)`
  
  *Displays the possible items.*

- **void** `UpdateSelection(bool addSelected)`
  
  *Function is fired when an item in the item select UI panel is clicked. This will tentatively highlight that item as the item that should be used in the recipe. However this is not final until the continue/craft button is clicked.*

- **void** `ToggleIngredientSelectionPanel(bool on)`
  
  *Toggles the ingredient selection panel.*

- **void** `AddSelectedIngredients()`
  
  *Adds the selected ingredients to the itemSelected list.*

- **void** `Craft()`
Craft the item stated in the recipe.

- void **EndCraftingAttempt** ()
  
  Cancels the attempt on a recipe. Closes the item selection UI panel.

- void **HidePanel** ()
  
  Hides the panel.

### 4.175.1 Detailed Description

Definition at line 8 of file RecipePageBehavior.cs.

### 4.175.2 Member Function Documentation

#### 4.175.2.1 AddSelectedIngredients()

```csharp
void RecipePageBehavior.AddSelectedIngredients ()
```

Adds the selected ingredients to the itemSelected list.

Definition at line 373 of file RecipePageBehavior.cs.

#### 4.175.2.2 Craft()

```csharp
void RecipePageBehavior.Craft ()
```

Craft the item stated in the recipe.

Definition at line 386 of file RecipePageBehavior.cs.

#### 4.175.2.3 DisplayPossibleItems()

```csharp
void RecipePageBehavior.DisplayPossibleItems ( int stepInCraftingProcess )
```

Displays the possible items.

**Parameters**

- `stepInCraftingProcess` Step in crafting process.

Definition at line 227 of file RecipePageBehavior.cs.
4.175.2.4 EndCraftingAttempt()

```csharp
void RecipePageBehavior.EndCraftingAttempt()
```
Cancels the an attempt on a recipe. Closes the item selection UI panel.
Definition at line 407 of file RecipePageBehavior.cs.

4.175.2.5 HidePanel()

```csharp
void RecipePageBehavior.HidePanel()
```
Hides the panel.
Definition at line 422 of file RecipePageBehavior.cs.

4.175.2.6 NextStep()

```csharp
void RecipePageBehavior.NextStep()
```
In the item selection UI panel, advances the step that the recipe is on. Displays the next set of items that are allowed to be used in the recipe.
Definition at line 204 of file RecipePageBehavior.cs.

4.175.2.7 OnBeginCraftingClick()

```csharp
void RecipePageBehavior.OnBeginCraftingClick()
```
Raises the begin crafting click event.
Definition at line 167 of file RecipePageBehavior.cs.

4.175.2.8 ResetCraftingPanel()

```csharp
void RecipePageBehavior.ResetCraftingPanel()
```
Resets the crafting panel.
Definition at line 191 of file RecipePageBehavior.cs.

4.175.2.9 SetUpRecipePage()

```csharp
void RecipePageBehavior.SetUpRecipePage(
    Recipe pageRecipe,
    RecipeButtonGUIBehavior currentSelectedRecipeButton,
    bool craftable)
```
Sets up recipe page.
Parameters

<table>
<thead>
<tr>
<th>pageRecipe</th>
<th>Page recipe.</th>
</tr>
</thead>
<tbody>
<tr>
<td>currentSelectedRecipe</td>
<td>Button that is associated with the current recipe</td>
</tr>
<tr>
<td>craftable</td>
<td>Is the recipe craftable</td>
</tr>
</tbody>
</table>

Definition at line 96 of file RecipePageBehavior.cs.

4.175.2.10 ToggleIngredientSelectionPanel()

void RecipePageBehavior.ToggleIngredientSelectionPanel (bool on)

Toggles the ingredient selection panel.

Parameters

| on | If set to true on. |

Definition at line 360 of file RecipePageBehavior.cs.

4.175.2.11 UpdateSelection()

void RecipePageBehavior.UpdateSelection (bool addSelected)

Function is fired when an item in the item select ui panel is clicked. This will tentatively highlight that item as the item that should be used in the recipe. However this is not final until the continue/craft button is clicked.

Definition at line 318 of file RecipePageBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/CraftingGUI/RecipePageBehavior.cs
4.176 RecipeRequirementsUI Class Reference

Inheritance diagram for RecipeRequirementsUI:

```
MonoBehaviour

RecipeRequirementsUI
+ RequirementInstance
+ SetUpRequirement()
```

Collaboration diagram for RecipeRequirementsUI:

```
MonoBehaviour

Requirement
+ AmountRequired
+ ItemType
+ Requirement()
+ isFullfilled()
+ SubmitItem()

RecipeRequirementsUI
+ RequirementInstance
+ SetUpRequirement()
```

**Public Member Functions**

- void `SetUpRequirement (Requirement r)`

  Sets up requirement.
Public Attributes

- Requirement RequirementInstance

4.176.1 Detailed Description

Definition at line 5 of file RecipeRequirementsUI.cs.

4.176.2 Member Function Documentation

4.176.2.1 SetUpRequirement()

void RecipeRequirementsUI.SetUpRequirement (  
    Requirement r )

Sets up requirement.

Parameters

- r The red component.

Definition at line 25 of file RecipeRequirementsUI.cs.

4.176.3 Member Data Documentation

4.176.3.1 RequirementInstance

Requirement RecipeRequirementsUI.RequirementInstance

Definition at line 19 of file RecipeRequirementsUI.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/CraftingGUI/Recipe RequirementsUI.cs
4.177 RecipeYamlSerializer Class Reference

Inheritance diagram for RecipeYamlSerializer:

Collaboration diagram for RecipeYamlSerializer:

Public Member Functions

- `RecipeYamlSerializer` (string file)
Awake this instance.

- Dictionary<string, Recipe> LoadRecipes()

  Loads the recipes from the yaml file.

Additional Inherited Members

4.177.1 Detailed Description

Definition at line 11 of file RecipeYamlSerializer.cs.

4.177.2 Constructor & Destructor Documentation

4.177.2.1 RecipeYamlSerializer()

RecipeYamlSerializer.RecipeYamlSerializer ( string file )

Awake this instance.

Definition at line 16 of file RecipeYamlSerializer.cs.

4.177.3 Member Function Documentation

4.177.3.1 LoadRecipes()

Dictionary<string, Recipe> RecipeYamlSerializer.LoadRecipes ()

Loads the recipes from the yaml file.

Returns

  The recipes.

Definition at line 25 of file RecipeYamlSerializer.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/YamlParsing/RecipeYamlSerializer.cs
4.178 Regression Class Reference

Collaboration diagram for Regression:

```
+ Prediction()
```

Static Public Member Functions

- static float **Prediction** (float[] coefficients, int[,] powers, float[] inputs, float intercept)

  Uses linear regression from the coefficients, powers, and inputs to get a prediction.

4.178.1 Detailed Description

Definition at line 1 of file Regression.cs.

4.178.2 Member Function Documentation

4.178.2.1 Prediction()

```
static float Regression.Prediction {
    float [] coefficients,
    int powers[,] ,
    float [] inputs,
    float intercept ) [static]
```

Uses linear regression from the coefficients, powers, and inputs to get a prediction.

**Note:** the length of the powers on the inside should be size of the inputs array

Returns

The regression prediction.
Parameters

<table>
<thead>
<tr>
<th>coefficients</th>
<th>Coefficients.</th>
</tr>
</thead>
<tbody>
<tr>
<td>powers</td>
<td>Powers.</td>
</tr>
<tr>
<td>inputs</td>
<td>Inputs.</td>
</tr>
<tr>
<td>intercept</td>
<td>Intercept.</td>
</tr>
</tbody>
</table>

Definition at line 15 of file Regression.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/Regression.cs

### 4.179 Requirement Class Reference

A requirement includes a type of item (an item with a certain tag) and the amount of it.

Collaboration diagram for Requirement:

```
+ Requirement()
+ AmountRequired
+ ItemType
+ isFulfilled()
+ SubmitItem()
```

### Public Member Functions

- **Requirement ()**
  
  Initializes a new instance of the Requirement class. Used by the yaml deserializer.

- **bool isFulfilled ()**
  
  If the amountRequired is 0, then the requirement is fulfilled

- **void SubmitItem (BaseItem item)**
  
  Submits an item to see if it is something that can be used in the requirement.

### Properties

- **int AmountRequired [get, set]**
  
  How much of the item is needed for a recipe

- **string ItemType [get, set]**
  
  What type of item is accepted
4.179.1 Detailed Description

A requirement includes a type of item (an item with a certain tag) and the amount of it.

Definition at line 5 of file Requirement.cs.

4.179.2 Constructor & Destructor Documentation

4.179.2.1 Requirement()

Requirement.Requirement ( )

Initializes a new instance of the Requirement class. Used by the yaml deserializer.

Definition at line 30 of file Requirement.cs.

4.179.3 Member Function Documentation

4.179.3.1 isFullfilled()

bool Requirement.isFullfilled ( )

If the amountRequired is 0, then the requirement is fulfilled

Returns

true, if amountRequired is fullfilled, false otherwise.

Definition at line 38 of file Requirement.cs.

4.179.3.2 SubmitItem()

void Requirement.SubmitItem ( BaseItem item )

Submits an item to see if it is something that can be used in the requirement.

Parameters

| item | Item. |

Generated by Doxygen
Definition at line 47 of file Requirement.cs.

4.179.4  Property Documentation

4.179.4.1  AmountRequired

int Requirement.AmountRequired [get], [set]

how much of the item is needed for a recipe
The amount required.
Definition at line 12 of file Requirement.cs.

4.179.4.2  ItemType

string Requirement.ItemType [get], [set]

what type of item is accepted
The type of the item.
Definition at line 22 of file Requirement.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/Requirement.cs

4.180  RestartBehavior Class Reference

Inheritance diagram for RestartBehavior:
Collaboration diagram for RestartBehavior:

```
MonoBehaviour

RestartBehavior

+ Restart()
```

Public Member Functions

- void **Restart** ()

  Restarts the game. Goes back to main game scene and resets the game.

4.180.1 Detailed Description

Definition at line 5 of file RestartBehavior.cs.

4.180.2 Member Function Documentation

4.180.2.1 Restart()

```csharp
void RestartBehavior.Restart ()
```

Restarts the game. Goes back to main game scene and resets the game.

Definition at line 14 of file RestartBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/DeathScreen/RestartBehavior.cs
4.181 RooftopGeneration Class Reference

Inheritance diagram for RooftopGeneration:

```
MonoBehaviour

ItemGenerator
  # poolManager
  # districtItemInfo
  + GetItemExtents()
  + SetSeed()
  + AddTemplatesToItemPool()
  # getRarityInformation()
  # calculateBounds()

RooftopGeneration
  + AddDoorsToDistrict()
  + AddSheltersToDistrict()
  + PopulateRoof()
```
Collaboration diagram for RooftopGeneration:

Public Member Functions

- void AddDoorsToDistrict (List<GameObject> doors, List<float> doorExtents, string district)
  
  *Adds the doors to district.*

- void AddSheltersToDistrict (List<GameObject> shelters, List<float> shelterExtents, string district)
  
  *Adds the shelters to district.*

- void PopulateRoof (Building building, string district)
  
  *Populates the rooftop of a building.*
Additional Inherited Members

4.181.1 Detailed Description

Definition at line 5 of file RooftopGeneration.cs.

4.181.2 Member Function Documentation

4.181.2.1 AddDoorsToDistrict()

```csharp
void RooftopGeneration.AddDoorsToDistrict (  
    List< GameObject > doors,  
    List< float > doorExtents,  
    string district )
```

Adds the doors to district.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doors</td>
<td>Doors.</td>
</tr>
<tr>
<td>doorExtents</td>
<td>Door extents.</td>
</tr>
<tr>
<td>district</td>
<td>District.</td>
</tr>
</tbody>
</table>

Definition at line 52 of file RooftopGeneration.cs.

4.181.2.2 AddSheltersToDistrict()

```csharp
void RooftopGeneration.AddSheltersToDistrict (  
    List< GameObject > shelters,  
    List< float > shelterExtents,  
    string district )
```

Adds the shelters to district.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>shelters</td>
<td>Shelters.</td>
</tr>
<tr>
<td>shelterExtents</td>
<td>Shelter extents.</td>
</tr>
<tr>
<td>district</td>
<td>District.</td>
</tr>
</tbody>
</table>

Definition at line 64 of file RooftopGeneration.cs.
4.181.2.3 PopulateRoof()

```csharp
void RooftopGeneration.PopulateRoof(
    Building building,
    string district)
```

Populates the rooftop of a building.

**Parameters**

| Building  | Building information. |
| district  | District. |

Definition at line 75 of file RooftopGeneration.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/RooftopGeneration.cs

### 4.182 RooftopPointGenerator Class Reference

Inheritance diagram for RooftopPointGenerator:

```
SamplingPointGenerator
# cellSize
# grid
# waterTag
# defaultMinDistanceAway
# generateRandomPointAround()
# HasOverlappingNeighbors()
# verifyRaycastHit()

RooftopPointGenerator
+ GetValidPoints()
# PointToGrid()
```
Collaboration diagram for RooftopPointGenerator:

```plaintext
ItemPlacementSamplePoint
+ LocalTargetSurfaceLocation
+ MinDistance
+ GridPoint
+ WorldSpaceLocation
+ ItemIndex
+ Size
+ District
+ Type

#grid

SamplingPointGenerator
# cellSize
# waterTag
# defaultMinDistanceAway

# generateRandomPointAround()
# HasOverlappingNeighbors()
# verifyRaycastHit()

RooftopPointGenerator

+ GetValidPoints()
# PointToGrid()
```

**Public Member Functions**

- `List<ItemPlacementSamplePoint> GetValidPoints(Bounds targetBound, Vector3 targetCenter, District<ItemConfiguration> itemInfo, string district, bool hasDoor=false, bool hasShelter=false)`

  Takes the sampling points and checks to see if they are on locations that are not at too steep an incline on the building's surface. Then converts the sampling points to their world coordinates.

**Protected Member Functions**

- `Tuple<int, int> PointToGrid(Vector2 samplingPoint)`

  Gets the grid coordinates for a sampling point
Additional Inherited Members

4.182.1 Detailed Description

Definition at line 5 of file RooftopPointGenerator.cs.

4.182.2 Member Function Documentation

4.182.2.1 GetValidPoints()

```csharp
List<ItemPlacementSamplePoint> RooftopPointGenerator.GetValidPoints (  
    Bounds targetBound,  
    Vector3 targetCenter,  
    DistrictItemConfiguration itemInfo,  
    string district,  
    bool hasDoor = false,  
    bool hasShelter = false )
```

Takes the sampling points and checks to see if they are on locations that are not at too steep an incline on the building’s surface. Then converts the sampling points to their world coordinates.

Returns

The valid points.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetBound</td>
<td>Target bounds.</td>
</tr>
<tr>
<td>targetCenter</td>
<td>Target center.</td>
</tr>
<tr>
<td>itemInfo</td>
<td>Item info based on district.</td>
</tr>
<tr>
<td>district</td>
<td>Distric this belongs tot.</td>
</tr>
<tr>
<td>doorExtents</td>
<td>Door extents.</td>
</tr>
<tr>
<td>doorTemplates</td>
<td>Door templates.</td>
</tr>
<tr>
<td>hasDoor</td>
<td>If set to true has a door attachment.</td>
</tr>
<tr>
<td>hasShelter</td>
<td>If set to true has a shelter attachment.</td>
</tr>
</tbody>
</table>

Definition at line 39 of file RooftopPointGenerator.cs.

4.182.2.2 PointToGrid()

```csharp
Tuple<int, int> RooftopPointGenerator.PointToGrid (  
    Vector2 samplingPoint ) [protected]
```

Gets the grid coordinates for a sampling point.
Returns
The to grid.

Parameters

| samplingPoint | Sampling point |

Definition at line 371 of file RooftopPointGenerator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/RooftopPointGenerator.cs

### 4.183 SamplingPointGenerator Class Reference

Inheritance diagram for SamplingPointGenerator:
Collaboration diagram for SamplingPointGenerator:

```
ItemPlacementSamplePoint
  + LocalTargetSurfaceLocation
  + MinDistance
  + GridPoint
  + WorldSpaceLocation
  + ItemIndex
  + Size
  + District
  + Type

#grid

SamplingPointGenerator
  # cellSize
  # waterTag
  # defaultMinDistanceAway
  # generateRandomPointAround()
  # HasOverlappingNeighbors()
  # verifyRaycastHit()
```

Protected Member Functions

- Vector2 `generateRandomPointAround` (Vector2 targetPoint, float minDistance)
  
  Generates the random point around an existing targetPoint.
- bool `HasOverlappingNeighbors` (ItemPlacementSamplePoint samplePoint)
  
  Determines whether this point has neighbors that are too close.
- bool `verifyRaycastHit` (RaycastHit hit, bool generateInWater)
  
  Verifies that the raycastHit hits a point that is not water and not at an inclination greater than the maxAngle.

Protected Attributes

- float `cellSize`
  
  The size of the grid cells.
- ItemPlacementSamplePoint `[] grid`
  
  The grid that contains information about what space is occupied in the city.
- const string `waterTag = "Water"
  
  The water tag.
- const float `defaultMinDistanceAway = 0.5f`
  
  Minimum distance objects must be from each other by default, not accounting for the doorway which will increase it
4.183.1 Detailed Description

Definition at line 4 of file SamplingPointGenerator.cs.

4.183.2 Member Function Documentation

4.183.2.1 generateRandomPointAround()

Vector2 SamplingPointGenerator.generateRandomPointAround ( Vector2 targetPoint, float minDistance ) [protected]

Generates the random point around an existing targetPoint.

Returns
Random point relative to the building.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>targetPoint</td>
<td>Point to generate around</td>
</tr>
<tr>
<td>minDistance</td>
<td>Minimum distance from the current point that the new point must be.</td>
</tr>
</tbody>
</table>

Definition at line 37 of file SamplingPointGenerator.cs.

4.183.2.2 HasOverlappingNeighbors()

bool SamplingPointGenerator.HasOverlappingNeighbors ( ItemPlacementSamplePoint samplePoint ) [protected]

Determines whether this point has neighbors that are too close.

Returns
true if this instance has overlapping neighbors the specified samplePoint; otherwise, false.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>samplePoint</td>
<td>Sample point.</td>
</tr>
</tbody>
</table>

Definition at line 57 of file SamplingPointGenerator.cs.
4.183.2.3 verifyRaycastHit()

```csharp
bool SamplingPointGenerator.verifyRaycastHit (RaycastHit hit, bool generateInWater) [protected]
```

Verifies that the raycastHit hits a point that is not water and not at an inclination greater than the maxAngle.

**Returns**
- `true`, if raycast hits a point that meets specifications
- `false`, otherwise.

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>hit</code></td>
<td>Hit.</td>
</tr>
<tr>
<td><code>generateInWater</code></td>
<td>Whether or not it should be considered valid if generating in water.</td>
</tr>
</tbody>
</table>

Definition at line 133 of file SamplingPointGenerator.cs.

### 4.183.3 Member Data Documentation

#### 4.183.3.1 cellSize

```csharp
float SamplingPointGenerator.cellSize [protected]
```

The size of the grid cells.

Definition at line 9 of file SamplingPointGenerator.cs.

#### 4.183.3.2 defaultMinDistanceAway

```csharp
const float SamplingPointGenerator.defaultMinDistanceAway = 0.5f [protected]
```

Minimum distance objects must be from each other by default, not accounting for the doorway which will increase it.

Definition at line 24 of file SamplingPointGenerator.cs.

#### 4.183.3.3 grid

```csharp
ItemPlacementSamplePoint[,] SamplingPointGenerator.grid [protected]
```

The grid that contains information about what space is occupied in the city.

Definition at line 14 of file SamplingPointGenerator.cs.
4.183.4  waterTag

```csharp
const string SamplingPointGenerator.waterTag = "Water"  [protected]
```

The water tag.

Definition at line 19 of file SamplingPointGenerator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/SamplingPointGenerator.cs

### 4.184 SelectedIngredientButton Class Reference

Inheritance diagram for SelectedIngredientButton:
Collaboration diagram for SelectedIngredientButton:

Public Member Functions

- void **SetUpSelection** (IngredientButtonBehavior source, string name)
  
  Sets up class.
- void **Unsubscribe** ()
  
  Unsubscribe this instance from the events its subscribed to. Should be called prior to destruction of this class.
- void **HandleAddSelectEvent** ()
  
  Handles the add select event.
- void **RemoveSelection** ()
  
  Removes the selection.

Properties

- int **Amount** [get, set]
  
  Gets or sets the amount.
- string **ItemName** [get, set]
  
  Gets or sets the name of the item.
- string **AssociatedStackId** [get, set]
  
  Gets or sets the associated stack identifier.

4.184.1 Detailed Description

Definition at line 5 of file SelectedIngredientButton.cs.
4.184.2 Member Function Documentation

4.184.2.1 HandleAddSelectEvent()

```csharp
void SelectedIngredientButton.HandleAddSelectEvent();
```

Handles the add select event.

Definition at line 73 of file SelectedIngredientButton.cs.

4.184.2.2 RemoveSelection()

```csharp
void SelectedIngredientButton.RemoveSelection();
```

Removes the selection.

Definition at line 82 of file SelectedIngredientButton.cs.

4.184.2.3 SetUpSelection()

```csharp
void SelectedIngredientButton.SetUpSelection(
    IngredientButtonBehavior source,
    string name)
```

Sets up class.

Parameters

<table>
<thead>
<tr>
<th>source</th>
<th>Source.</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name.</td>
</tr>
</tbody>
</table>

Definition at line 50 of file SelectedIngredientButton.cs.

4.184.2.4 Unsubscribe()

```csharp
void SelectedIngredientButton.Unsubscribe();
```

Unsubscribe this instance from the events its subscribed to. Should be called prior to destruction of this class.

Definition at line 65 of file SelectedIngredientButton.cs.
4.184.3 Property Documentation

4.184.3.1 Amount

`int SelectedIngredientButton.Amount [get], [set]`

Gets or sets the amount.

The amount.

Definition at line 20 of file SelectedIngredientButton.cs.

4.184.3.2 AssociatedStackId

`string SelectedIngredientButton.AssociatedStackId [get], [set]`

Gets or sets the associated stack identifier.

The associated stack identifier.

Definition at line 40 of file SelectedIngredientButton.cs.

4.184.3.3 ItemName

`string SelectedIngredientButton.ItemName [get], [set]`

Gets or sets the name of the item.

The name of the item.

Definition at line 30 of file SelectedIngredientButton.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/CraftingGUI/SelectedIngredientButton.cs
4.185 SetCreatureGUI Class Reference

Inheritance diagram for SetCreatureGUI:

```
MonoBehaviour

SetCreatureGUI
```

Collaboration diagram for SetCreatureGUI:

```
MonoBehaviour

SetCreatureGUI
```

4.185.1 Detailed Description

Definition at line 5 of file SetCreatureGUI.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Creatures/UI/SetCreatureGUI.cs
4.186 SetEnvironmentGUI Class Reference

Inheritance diagram for SetEnvironmentGUI:

Collaboration diagram for SetEnvironmentGUI:

4.186.1 Detailed Description

Definition at line 6 of file SetEnvironmentGUI.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/GUI/SetEnvironmentGUI.cs
4.187 SettingsMenuBehavior Class Reference

Inheritance diagram for SettingsMenuBehavior:

```
MonoBehaviour

SettingsMenuBehavior
+ turnSoundOnOff()
+ changeVolume()
+ OnSeedValueChanged()
+ OnConfigKeysClick()
```

Collaboration diagram for SettingsMenuBehavior:

```
MonoBehaviour

SettingsMenuBehavior
+ turnSoundOnOff()
+ changeVolume()
+ OnSeedValueChanged()
+ OnConfigKeysClick()
```

Public Member Functions

- void turnSoundOnOff ()
• void changeVolume(float value)

Changes the volume.

• void OnSeedValueChanged(string inputSeed)

Validates seed input filed and sets the procedural city generation seed.

• void OnConfigKeysClick()

Opens the user input configuration panel.

4.187.1 Detailed Description

Definition at line 6 of file SettingsMenuBehavior.cs.

4.187.2 Member Function Documentation

4.187.2.1 changeVolume()

definition at line 51 of file SettingsMenuBehavior.cs.

Parameters

| value | Value |

4.187.2.2 OnConfigKeysClick()

definition at line 72 of file SettingsMenuBehavior.cs.

4.187.2.3 OnSeedValueChanged()

void SettingsMenuBehavior.OnSeedValueChanged ( string inputSeed )

Validates seed input field and sets the procedural city generation seed.

Definition at line 59 of file SettingsMenuBehavior.cs.
4.187.2.4 turnSoundOnOff()

void SettingsMenuBehavior.turnSoundOnOff ( )

Turns the sound on or off based on toggle value.

Parameters

- **isChecked**: If set to true is sound on.

Definition at line 35 of file SettingsMenuBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/MainMenuScene/SettingsMenuBehavior.cs

4.188 ShelterCategory Class Reference

Inheritance diagram for ShelterCategory:
Collaboration diagram for ShelterCategory:

Public Member Functions

- **override ItemCategory GetDuplicate ()**
  
  Creates a copy of this shelter category.

- **override void ReadyCategory ()**
  
  Readies the item category by adding the attributes and actions it can complete.

- **void SetDown ()**
  
  Sets down the shelter in the world. Drops it where the player stands.
Properties

- **int WarmthRate [get, set]**
  
  Gets and sets the warmth rate.

Additional Inherited Members

4.188.1 Detailed Description

Definition at line 6 of file ShelterCategory.cs.

4.188.2 Member Function Documentation

4.188.2.1 GetDuplicate()

override ItemCategory ShelterCategory.GetDuplicate () [virtual]

Creates a copy of this shelter category.

Returns

The duplicate.

Reimplemented from ItemCategory.

Definition at line 25 of file ShelterCategory.cs.

4.188.2.2 ReadyCategory()

override void ShelterCategory.ReadyCategory () [virtual]

Readies the item category by adding the attributes and actions it can complete.

Reimplemented from ItemCategory.

Definition at line 45 of file ShelterCategory.cs.

4.188.2.3 SetDown()

void ShelterCategory.SetDown ()

Sets down the shelter in the world. Drops it where the player stands.

Definition at line 57 of file ShelterCategory.cs.
4.188.3 Property Documentation

4.188.3.1 WarmthRate

```csharp
int ShelterCategory.WarmthRate [get], [set]
```

Gets and sets the warmth rate.

Definition at line 12 of file ShelterCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/ShelterCategory.cs

4.189 ShelterInteractable Class Reference

Inheritance diagram for ShelterInteractable:

![Inheritance Diagram](image_url)
Collaboration diagram for ShelterInteractable:

```
MonoBehaviour

InteractableObject
+ Show
+ Text
# setupComplete
+ SetUp()
+ PerformAction()
+ SetAction()

ShelterInteractable
+ Shelter
+ SetUp()
+ OnTriggerEnter()
+ OnTriggerExit()
```

**Public Member Functions**

- override void **SetUp ()**
  
  Sets up shelter.

- void **OnTriggerEnter** (Collider other)
  
  Enter the shelter.

- void **OnTriggerExit** (Collider other)
  
  Exit the shelter.

**Properties**

- **ShelterCategory Shelter** [get, set]
  
  Gets or sets the shelter category object.

**Additional Inherited Members**

4.189.1 Detailed Description

Definition at line 4 of file ShelterInteractable.cs.
4.189.2  Member Function Documentation

4.189.2.1  OnTriggerEnter()

```csharp
void ShelterInteractable.OnTriggerEnter ( Collider other )
```

Enter the shelter.

Definition at line 28 of file ShelterInteractable.cs.

4.189.2.2  OnTriggerExit()

```csharp
void ShelterInteractable.OnTriggerExit ( Collider other )
```

Exit the shelter.

Definition at line 40 of file ShelterInteractable.cs.

4.189.2.3  SetUp()

```csharp
override void ShelterInteractable.SetUp ( ) [virtual]
```

Sets up shelter.

Reimplemented from InteractableObject.

Definition at line 20 of file ShelterInteractable.cs.

4.189.3  Property Documentation

4.189.3.1  Shelter

```csharp
ShelterCategory ShelterInteractable.Shelter [get], [set]
```

Gets or sets the shelter category object.

Definition at line 12 of file ShelterInteractable.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/ShelterInteractable.cs
4.190  SicknessNotification Class Reference

Inheritance diagram for SicknessNotification:

```
MonoBehaviour

SicknessNotification
+ CloseNotification()
```

Collaboration diagram for SicknessNotification:

```
MonoBehaviour

SicknessNotification
+ CloseNotification()
```

Public Member Functions

- void CloseNotification ()
  
  Closes out notification bar.

4.190.1  Detailed Description

Definition at line 6 of file SicknessNotification.cs.
4.190.2 Member Function Documentation

4.190.2.1 CloseNotification()

```csharp
void SicknessNotification.CloseNotification();
```

Closes out notification bar.

Definition at line 106 of file SicknessNotification.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/GameViewScene/SicknessNotification.cs

4.191 SolidCategory Class Reference

The solid item category. Contains attributes and actions that befit a solid category item.
Inheritance diagram for SolidCategory:
Collaboration diagram for SolidCategory:

Public Member Functions

- **override ItemCategory GetDuplicate()**
  
  *Creates a copy of the ItemCategory.*

- **override void ReadyCategory()**
  
  *Preps the category for use by loading attributes.*

- **void WeaveRope()**
A subcategory action of "Weave". Makes the item into an item of rope type. If the rope is thinner, then the name will be thread, otherwise it will be rope.

- void **WeaveBasket** ()
  A subcategory action of "Weave". Makes the item into a "Container" type object. Removes all other item categories from it afterwards.
- void **Sharpen** ()
  Sharpens the item. If it is sharp enough, it becomes a blade.

**Properties**

- float **Flexibility** [get, set]
  Gets or sets the flexibility.
- float **Durability** [get, set]
  Gets or sets the durability.
- float **Elasticity** [get, set]
  Gets or sets the elasticity.
- float **Stickiness** [get, set]
  Gets or sets the stickiness.
- float **Thickness** [get, set]
  Gets or sets the thickness.
- float **Sharpness** [get, set]
  Gets or sets the sharpness.

**Additional Inherited Members**

4.191.1 Detailed Description

The solid item category. Contains attributes and actions that befit a solid category item.

Definition at line 10 of file SolidCategory.cs.

4.191.2 Member Function Documentation

4.191.2.1 **GetDuplicate()**

```csharp
override ItemCategory SolidCategory.GetDuplicate () [virtual]
```

Creates a copy of the ItemCategory.

Returns

The duplicate.

Reimplemented from ItemCategory.

Definition at line 144 of file SolidCategory.cs.
### 4.191.2.2 ReadyCategory()

```csharp
override void SolidCategory.ReadyCategory() [virtual]
```

Preps the category for use by loading attributes.

Reimplemented from `ItemCategory`.

Definition at line 175 of file `SolidCategory.cs`.

### 4.191.2.3 Sharpen()

```csharp
void SolidCategory.Sharpen()
```

Sharpens the item. If it is sharp enough, it becomes a blade.

Definition at line 286 of file `SolidCategory.cs`.

### 4.191.2.4 WeaveBasket()

```csharp
void SolidCategory.WeaveBasket()
```

A subcategory action of "Weave". Makes the item into a "Container" type object. Removes all other item categories from it afterwards.

Definition at line 249 of file `SolidCategory.cs`.

### 4.191.2.5 WeaveRope()

```csharp
void SolidCategory.WeaveRope()
```

A subcategory action of "Weave". Makes the item into an item of rope type. If the rope is thinner, then the name will be thread, otherwise it will be rope.

Definition at line 215 of file `SolidCategory.cs`.

### 4.191.3 Property Documentation

Generated by Doxygen
4.191.3.1 Durability

float SolidCategory.Durability [get], [set]

Gets or sets the durability.

The durability.

Definition at line 28 of file SolidCategory.cs.

4.191.3.2 Elasticity

float SolidCategory.Elasticity [get], [set]

Gets or sets the elasticity.

The elasticity.

Definition at line 38 of file SolidCategory.cs.

4.191.3.3 Flexibility

float SolidCategory.Flexibility [get], [set]

Gets or sets the flexibility.

The flexibility.

Definition at line 18 of file SolidCategory.cs.

4.191.3.4 Sharpness

float SolidCategory.Sharpness [get], [set]

Gets or sets the sharpness.

The sharpness.

Definition at line 68 of file SolidCategory.cs.
### 4.191.3.5 Stickiness

```csharp
float SolidCategory.Stickiness [get], [set]
```

Gets or sets the stickiness.

The stickiness.

Definition at line 48 of file SolidCategory.cs.

### 4.191.3.6 Thickness

```csharp
float SolidCategory.Thickness [get], [set]
```

Gets or sets the thickness.

The thickness.

Definition at line 58 of file SolidCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/SolidCategory.cs

### 4.192 SplashScreenBehavior Class Reference

Inheritance diagram for SplashScreenBehavior:
Collaboration diagram for SplashScreenBehavior:

```
MonoBehaviour

+ SplashScreenBehavior
```

### 4.192.1 Detailed Description

Definition at line 5 of file SplashScreenBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/MainMenuScene/SplashScreenBehavior.cs

### 4.193 SpriteManager Class Reference

Collaboration diagram for SpriteManager:

```
SpriteManager

+ SpriteManager()
+ GetSprite()
```

**Public Member Functions**

- **SpriteManager** (string atlasPath)
  
  Initializes a new instance of the SpriteManager class.

- Sprite **GetSprite** (string name)
  
  Gets the sprite from the sprite sheet by name.
4.193 SpriteManager Class Reference

4.193.1 Detailed Description

Definition at line 5 of file SpriteManager.cs.

4.193.2 Constructor & Destructor Documentation

4.193.2.1 SpriteManager()

SpriteManager(SpriteManager (  
    string atlasPath )

Initializes a new instance of the SpriteManager class.

Parameters

- **atlasPath** Atlas path.

Definition at line 13 of file SpriteManager.cs.

4.193.3 Member Function Documentation

4.193.3.1 GetSprite()

Sprite SpriteManager.GetSprite (  
    string name )

Gets the sprite from the sprite sheet by name.

Returns

The sprite.

Parameters

- **name** Name.

Definition at line 30 of file SpriteManager.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/Utility/SpriteManager.cs

Generated by Doxygen
4.194 StatRate Class Reference

Inheritance diagram for StatRate:

```
+ StatRate
  + Units
  + PerSeconds
  + CurrentStat
  + MaxStat

+ HealthRateManager
  + HealthRateManager()
  + UseDefaultHealthRate()
  + TakeFallDamage()
  + UseHungerZeroHealthReductionRate()
  + UseHeatSourceHealthIncreaseRate()
  + UseHealthEnergy()

+ HungerRateManager
  + HungerRateManager()
  + UseDefaultHungerReductionRate()
  + UseFoodEnergy()

+ WarmthRateManager
  + WarmthRateManager()
  + UseDefaultWarmthReductionRate()
  + UseWaterWarmthReductionRate()
  + UseHeatSourceWarmthIncreaseRate()
  + UseHeatSource()
  + SetUnitsInShelter()
```

Collaboration diagram for StatRate:

```
StatRate
+ Units
+ PerSeconds
+ CurrentStat
+ MaxStat

+ StatRate()
+ ChangeRateValues()
+ ApplyRateToStat()
```

Public Member Functions

- **StatRate** (int units, int perSeconds)
  *Initializes a new instance of the StatRate class.*

- void **ChangeRateValues** (int newUnits, int newSeconds)
  *Changes the rate values.*

- void **ApplyRateToStat** ()
  *Applies the rate to stat.*

Protected Member Functions

- **StatRate** ()
  *Initializes a new instance of the StatRate class.*
4.194 StatRate Class Reference

Properties

- int Units [get, protected set]
  Gets the number of units that should decrease or increase.
- int PerSeconds [get, protected set]
  Gets the interval of seconds that the units should change after.
- int CurrentStat [get, protected set]
  Gets the current health stat.
- int MaxStat [get, set]
  Gets or sets the max stat.

4.194.1 Detailed Description

Definition at line 3 of file StatRate.cs.

4.194.2 Constructor & Destructor Documentation

4.194.2.1 StatRate() [1/2]

StatRate.StatRate () [protected]

Initializes a new instance of the StatRate class.

Definition at line 8 of file StatRate.cs.

4.194.2.2 StatRate() [2/2]

StatRate.StatRate ( int units,
                   int perSeconds )

Initializes a new instance of the StatRate class.

Parameters

<table>
<thead>
<tr>
<th>units</th>
<th>Units.</th>
</tr>
</thead>
<tbody>
<tr>
<td>perSeconds</td>
<td>Per seconds.</td>
</tr>
</tbody>
</table>

Definition at line 15 of file StatRate.cs.

4.194.3 Member Function Documentation
4.194.3.1 ApplyRateToStat()

```csharp
void StatRate.ApplyRateToStat();
```
Applies the rate to stat.

Definition at line 34 of file StatRate.cs.

4.194.3.2 ChangeRateValues()

```csharp
void StatRate.ChangeRateValues(int newUnits, int newSeconds);
```
Changes the rate values.

Parameters

- **newUnits**: New units.
- **newSeconds**: New seconds.

Definition at line 25 of file StatRate.cs.

4.194.4 Property Documentation

4.194.4.1 CurrentStat

```csharp
int StatRate.CurrentStat [get], [protected set];
```
Gets the current health stat.

The current health stat.

Definition at line 64 of file StatRate.cs.

4.194.4.2 MaxStat

```csharp
int StatRate.MaxStat [get], [set], [protected];
```
Gets or sets the max stat.

The max stat.

Definition at line 74 of file StatRate.cs.
4.194.3  PerSeconds

    int StatRate.PerSeconds  [get], [protected set]

    Gets the interval of seconds that the units should change after.
    The seconds.

    Definition at line 54 of file StatRate.cs.

4.194.4  Units

    int StatRate.Units  [get], [protected set]

    Gets the number of units that should decrease or increase.
    The units.

    Definition at line 44 of file StatRate.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Player/StatRate.cs

4.195  TeleportPlayer.TeleportLocation Struct Reference

Collaboration diagram for TeleportPlayer.TeleportLocation:

```
TeleportPlayer.TeleportLocation
+ CanTeleportTo
+ Location
```

Public Attributes

- bool CanTeleportTo
Properties

- **Transform Location** [get]
  
  A location the player can teleport to.

### 4.195.1 Detailed Description

Definition at line 12 of file TeleportPlayer.cs.

### 4.195.2 Member Data Documentation

#### 4.195.2.1 CanTeleportTo

bool TeleportPlayer.TeleportLocation.CanTeleportTo

Definition at line 27 of file TeleportPlayer.cs.

### 4.195.3 Property Documentation

#### 4.195.3.1 Location

Transform TeleportPlayer.TeleportLocation.Location [get]

A location the player can teleport to.

Definition at line 20 of file TeleportPlayer.cs.

The documentation for this struct was generated from the following file:

- Assets/Scripts/Teleporters/TeleportPlayer.cs
4.196 TeleportPlayer Class Reference

A class that moves the player when they press a key. Used by fire escapes, certain doors, and the window washers.

Inheritance diagram for TeleportPlayer:

Collaboration diagram for TeleportPlayer:

Classes

- struct TeleportLocation
Public Member Functions

- void TeleportToA()
  
  Teleports the player to the A position.

- void TeleportToB()
  
  Teleports the player to the B position.

4.196.1 Detailed Description

A class that moves the player when they press a key. Used by fire escapes, certain doors, and the window washers.

TODO: Make the class work with fire escapes.

Definition at line 9 of file TeleportPlayer.cs.

4.196.2 Member Function Documentation

4.196.2.1 TeleportToA()

void TeleportPlayer.TeleportToA ( )

Teleports the player to the A position.

Definition at line 39 of file TeleportPlayer.cs.

4.196.2.2 TeleportToB()

void TeleportPlayer.TeleportToB ( )

Teleports the player to the B position.

Definition at line 51 of file TeleportPlayer.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Teleporters/TeleportPlayer.cs
A building based on a prefab.

Inheritance diagram for TemplateBuilding:
Collaboration diagram for TemplateBuilding:

```
Building
+ IsLoaded
+ Position
+ Instance
+ Parent
+ AttachmentInformation
+ Attachments
+ BoundingBox
+ Load()
+ Unload()

TemplateBuilding
+ Template
+ TemplateBuilding()
+ Load()
```

Public Member Functions

- **TemplateBuilding (Transform parent, Vector3 position, GameObject template)**
  
  Creates an instance of a TemplateBuilding class.

- **override void Load ()**
  
  Loads the instance of the building into the scene.

Properties

- **GameObject Template [get]**
  
 Prefab to instantiate.

4.197.1 Detailed Description

A building based on a prefab.

Definition at line 7 of file TemplateBuilding.cs.

4.197.2 Constructor & Destructor Documentation
4.197.1 TemplateBuilding()

TemplateBuilding.TemplateBuilding (  
    Transform parent,  
    Vector3 position,  
    GameObject template )

Creates an instance of a TemplateBuilding class.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>template</td>
<td>The prefab gameobject.</td>
</tr>
<tr>
<td>parent</td>
<td>The gameobject the building should be parented to.</td>
</tr>
<tr>
<td>position</td>
<td>The position of the building.</td>
</tr>
</tbody>
</table>

Definition at line 15 of file TemplateBuilding.cs.

4.197.3 Member Function Documentation

4.197.3.1 Load()

override void TemplateBuilding.Load ( ) [virtual]

Loads the instance of the building into the scene.

Implements Building.

Definition at line 35 of file TemplateBuilding.cs.

4.197.4 Property Documentation

4.197.4.1 Template

GameObject TemplateBuilding.Template [get]

Prefab to instantiate.

Definition at line 27 of file TemplateBuilding.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/TemplateBuilding.cs
Inheritance diagram for TestGUI:

```
Monobehaviour

TestGUI
+ buttons
+ radio
+ audios
+ soundA
+ soundB
+ newClip

+ ShowRadioButtons()
+ Weather_Clicked()
+ Music_Clicked()
```
Collaboration diagram for TestGUI:

```
Collaboration diagram for TestGUI:

MonoBehaviour

Radio
+ MusicDefaultPath
+ MysteryDefaultPath
+ StaticDefaultPath
+ lowMusic
+ highMusic
+ lowMystery
+ highMystery
+ lowWeather
+ highWeather
+ CurrentChannel
+ Power() 
+ AddToCarousel()
+ SetChannel()
+ ChangeChannel()
+ OnChannelClick()
+ GetWeatherAnnouncement()
+ GetMysteryAnnouncement()

TestGUI
+ buttons
+ audios
+ soundA
+ soundB
+ newClip
+ ShowRadioButtons()
+ Weather_Clicked()
+ Music_Clicked()
```

Public Member Functions

- void `ShowRadioButtons` ()
- void `Weather_Clicked` ()
- void `Music_Clicked` ()
Public Attributes

- GameObject [] buttons
- Radio radio
- AudioSource [] audios
- AudioSource soundA
- AudioSource soundB
- AudioClip newClip

4.198.1 Detailed Description

Definition at line 4 of file TestGUI.cs.

4.198.2 Member Function Documentation

4.198.2.1 Music_Clicked()

void TestGUI.Music_Clicked ( )

Definition at line 43 of file TestGUI.cs.

4.198.2.2 ShowRadioButtons()

void TestGUI.ShowRadioButtons ( )

Definition at line 23 of file TestGUI.cs.

4.198.2.3 Weather_Clicked()

void TestGUI.Weather_Clicked ( )

Definition at line 38 of file TestGUI.cs.

4.198.3 Member Data Documentation
4.198.3.1 audios

AudioSource [] TestGUI.audios

Definition at line 7 of file TestGUI.cs.

4.198.3.2 buttons

GameObject [] TestGUI.buttons

Definition at line 5 of file TestGUI.cs.

4.198.3.3 newClip

AudioClip TestGUI.newClip

Definition at line 10 of file TestGUI.cs.

4.198.3.4 radio

Radio TestGUI.radio

Definition at line 6 of file TestGUI.cs.

4.198.3.5 soundA

AudioSource TestGUI.soundA

Definition at line 8 of file TestGUI.cs.

4.198.3.6 soundB

AudioSource TestGUI.soundB

Definition at line 9 of file TestGUI.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Radio/TestGUI.cs
Inheritance diagram for Thermometer:

```
MonoBehaviour
    
    Tool
    + ToolName
    + InUse
    # attachJoint
    # toolName
    # unequipActName

    + Use()
    + Equip()
    + Unequip()
    + SetUpTool()

Thermometer
    + CheckThermometer

    + SetUpTool()
    + Use()
    + Equip()
    + Unequip()
```

Generated by Doxygen
Collaboration diagram for Thermometer:

```
MonoBehaviour

Tool
+ ToolName
+ InUse
# attachJoint
# toolName
# unequipActName
+ Use()
+ Equip()
+ Unequip()
+ SetUpTool()

Thermometer
+ CheckThermometer
+ SetUpTool()
+ Use()
+ Equip()
+ Unequip()
```

Public Member Functions

- **override void** `SetUpTool (BaseItem itemForTool)`
  
  *Sets up the tool so that it is linked to the proper item in the inventory.*

- **override void** `Use ()`
  
  *Uses the thermometer.*

- **override void** `Equip ()`
  
  *Equip the thermometer.*

- **override void** `Unequip ()`
  
  *Unequip the thermometer.*

Properties

- **bool** `CheckThermometer` [get]
  
  *Returns true if the thermometer is in use.*
Additional Inherited Members

4.199.1 Detailed Description

Definition at line 7 of file Thermometer.cs.

4.199.2 Member Function Documentation

4.199.2.1 Equip()

```csharp
override void Thermometer.Equip () [virtual]
```

Equip the thermometer.

Implements `Tool`.

Definition at line 66 of file Thermometer.cs.

4.199.2.2 SetUpTool()

```csharp
override void Thermometer.SetUpTool ( BaseItem itemForTool ) [virtual]
```

Sets up the tool so that it is linked to the proper item in the inventory.

Parameters

- `itemForTool` Item for tool.

Reimplemented from `Tool`.

Definition at line 32 of file Thermometer.cs.

4.199.2.3 Unequip()

```csharp
override void Thermometer.Unequip () [virtual]
```

Unequip the thermometer.

Implements `Tool`.

Definition at line 75 of file Thermometer.cs.
4.199.2.4 Use()

```csharp
override void Thermometer.Use() [virtual]
```

Uses the thermometer.

Implements Tool.

Definition at line 57 of file Thermometer.cs.

4.199.3 Property Documentation

4.199.3.1 CheckThermometer

```csharp
bool Thermometer.CheckThermometer [get]
```

Returns true if the thermometer is in use.

Definition at line 85 of file Thermometer.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/Thermometer.cs

4.200 Tool Class Reference

Inheritance diagram for Tool:

![Inheritance Diagram](image-url)
Collaboration diagram for Tool:

```
MonoBehaviour

Tool
+ ToolName
+ InUse
# attachJoint
# toolName
# unequipActName

+ Use()
+ Equip()
+ Unequip()
+ SetUpTool()
```

Public Member Functions

- abstract void `Use()`
- abstract void `Equip()`
- abstract void `Unequip()`
- virtual void `SetUpTool(BaseItem itemForTool)`
  
  *Sets up the tool so that it is linked to the proper item in the inventory.*

Protected Attributes

- GameObject `attachJoint`
- string `toolName`
- const string `unequipActName = "Unequip"`

Properties

- string `ToolName [get, protected set]`
  
  *The name of the tool, eg. "fishing rod"*
- bool `InUse [get, protected set]`
  
  *Return true if the tool is being used.*

4.200.1 Detailed Description

Definition at line 4 of file Tool.cs.
4.200 Tool Class Reference

4.200.2 Member Function Documentation

4.200.2.1 Equip()

abstract void Tool.Equip ( ) [pure virtual]

Implemented in FishingRod, LightItem, Barometer, Thermometer, and Idol.

4.200.2.2 SetUpTool()

virtual void Tool.SetUpTool ( BaseItem itemForTool ) [virtual]

Sets up the tool so that it is linked to the proper item in the inventory.

Parameters

itemForTool Item for tool.

Reimplemented in LightItem, FishingRod, Barometer, Thermometer, and Idol.

Definition at line 44 of file Tool.cs.

4.200.2.3 Unequip()

abstract void Tool.Unequip ( ) [pure virtual]

Implemented in FishingRod, LightItem, Barometer, Thermometer, and Idol.

4.200.2.4 Use()

abstract void Tool.Use ( ) [pure virtual]

Implemented in LightItem, FishingRod, Idol, Barometer, and Thermometer.

4.200.3 Member Data Documentation

Generated by Doxygen
4.200.3.1 attachJoint

GameObject Tool.attachJoint [protected]

Definition at line 8 of file Tool.cs.

4.200.3.2 toolName

string Tool.toolName [protected]

Definition at line 12 of file Tool.cs.

4.200.3.3 unequipActName

const string Tool.unequipActName = "Unequip" [protected]

Definition at line 14 of file Tool.cs.

4.200.4 Property Documentation

4.200.4.1 InUse

bool Tool.InUse [get], [protected set]

Return true if the tools is being used.

Definition at line 35 of file Tool.cs.

4.200.4.2 ToolName

string Tool.ToolName [get], [protected set]

The name of the tool, eg. "fishing rod"

Definition at line 26 of file Tool.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Items/Tool.cs
4.201 TransformParent Class Reference

Can Parent one transform to another while ignoring specific axis.

Inheritance diagram for TransformParent:

Collaboration diagram for TransformParent:
Public Attributes

- bool UsePositionX
- bool UsePositionY
- bool UsePositionZ
- Transform Parent

4.201.1 Detailed Description

Can Parent one transform to another while ignoring specific axis.

Definition at line 8 of file TransformParent.cs.

4.201.2 Member Data Documentation

4.201.2.1 Parent

Transform TransformParent.Parent

Definition at line 14 of file TransformParent.cs.

4.201.2.2 UsePositionX

bool TransformParent.UsePositionX

Definition at line 10 of file TransformParent.cs.

4.201.2.3 UsePositionY

bool TransformParent.UsePositionY

Definition at line 11 of file TransformParent.cs.

4.201.2.4 UsePositionZ

bool TransformParent.UsePositionZ

Definition at line 12 of file TransformParent.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/TransformParent.cs
4.202 Tuple<T1, T2> Class Template Reference

Literally here so I can have a way of having int pairs without converting back and forth from Vector2.

Collaboration diagram for Tuple<T1, T2>:

<table>
<thead>
<tr>
<th>Tuple&lt;T1, T2&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ X</td>
</tr>
<tr>
<td>+ Y</td>
</tr>
<tr>
<td>+ New&lt;T1, T2&gt;()</td>
</tr>
<tr>
<td>~ Tuple()</td>
</tr>
</tbody>
</table>

Static Public Member Functions

- static Tuple<T1, T2> New<T1, T2>(T1 xVal, T2 yVal)

Properties

- T1 X [get, set]
- T2 Y [get, set]

4.202.1 Detailed Description

Literally here so I can have a way of having int pairs without converting back and forth from Vector2.

Definition at line 4 of file Tuple.cs.

4.202.2 Member Function Documentation

4.202.2.1 New<T1, T2>()

```csharp
static Tuple<T1, T2> Tuple<T1, T2>.New<T1, T2>(
    T1 xVal,
    T2 yVal ) [static]
```

Definition at line 32 of file Tuple.cs.
4.202.3 Property Documentation

4.202.3.1 X

\texttt{T1 } \texttt{Tuple< T1, T2 >.X \ [get], \ [set]}

Definition at line 7 of file Tuple.cs.

4.202.3.2 Y

\texttt{T2 } \texttt{Tuple< T1, T2 >.Y \ [get], \ [set]}

Definition at line 13 of file Tuple.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/Tuple.cs

4.203 TutorialGraffiti Class Reference

Inheritance diagram for TutorialGraffiti:
Collaboration diagram for TutorialGraffiti:

```
MonoBehaviour

TutorialGraffiti
+ setGraffitiText()
```

**Public Member Functions**

- void `setGraffitiText ()`
  
  *Set the text on the graffiti based on the current control scheme*

### 4.203.1 Detailed Description

Definition at line 6 of file TutorialGraffiti.cs.

### 4.203.2 Member Function Documentation

#### 4.203.2.1 setGraffitiText()

```c
void TutorialGraffiti.setGraffitiText ()
```

*Set the text on the graffiti based on the current control scheme*

Definition at line 35 of file TutorialGraffiti.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Graffiti/TutorialGraffiti.cs
4.204  uGUITools Class Reference

Inheritance diagram for uGUITools:

```
MonoBehaviour

uGUITools
```

Collaboration diagram for uGUITools:

```
MonoBehaviour

uGUITools
```

4.204.1 Detailed Description

Definition at line 4 of file uGUITools.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Editor/uGUITools.cs
Collaboration diagram for VectorUtility:

```
VectorUtility
+ GetSlope()
+ GetPerpendicularSlope()
+ GetAngle()
+ XZ()
+ twoDimensional3d()
```

**Static Public Member Functions**

- static float **GetSlope** (Vector2 posOne, Vector2 posTwo)
  
  *Gets the slope.*

- static float **GetPerpendicularSlope** (Vector2 posOne, Vector2 posTwo)
  
  *Gets the perpendicular slope.*

- static float **GetAngle** (Vector2 posOne, Vector2 posTwo)
  
  *Gets the tangent angle between two points*

- static Vector2 **XZ** (Vector3 vector)
  
  *Convert a vector3 to vector2 using the z axis instead of the y*

- static Vector3 **twoDimensional3d** (Vector2 vector, float yValue=0)
  
  *Convert a 2d vector to 3d with a y value*

**4.205.1 Detailed Description**

Definition at line 3 of file VectorUtility.cs.

**4.205.2 Member Function Documentation**

**4.205.2.1 GetAngle()**

```
static float VectorUtility.GetAngle (  
    Vector2 posOne,  
    Vector2 posTwo  
) [static]
```

*Gets the tangent angle between two points*

*Returns*

*The wind angle.*
4.205.2.2 GetPerpendicularSlope()

static float VectorUtility.GetPerpendicularSlope (  
  Vector2 posOne,  
  Vector2 posTwo ) [static]

Gets the perpendicular slope.

Returns

The perpendicular slope.

Parameters

<table>
<thead>
<tr>
<th>posOne</th>
<th>Position one.</th>
</tr>
</thead>
<tbody>
<tr>
<td>posTwo</td>
<td>Position two.</td>
</tr>
</tbody>
</table>

Definition at line 29 of file VectorUtility.cs.

4.205.3 GetSlope()

static float VectorUtility.GetSlope ( 
  Vector2 posOne,  
  Vector2 posTwo ) [static]

Gets the slope.

Returns

The slope.

Parameters

<table>
<thead>
<tr>
<th>posOne</th>
<th>Position one.</th>
</tr>
</thead>
<tbody>
<tr>
<td>posTwo</td>
<td>Position two.</td>
</tr>
</tbody>
</table>

Definition at line 11 of file VectorUtility.cs.
4.205.2.4  twoDimensional3d()

```csharp
static Vector3 VectorUtility.twoDimensional3d ( Vector2 vector,
    float yValue = 0 ) [static]
```

Convert a 2d vector to 3d with a y value

Returns
The dimensional3d.

Parameters

<table>
<thead>
<tr>
<th>vector</th>
<th>Vector.</th>
</tr>
</thead>
<tbody>
<tr>
<td>y_val</td>
<td>Y value.</td>
</tr>
</tbody>
</table>

Definition at line 71 of file VectorUtility.cs.

4.205.2.5  XZ()

```csharp
static Vector2 VectorUtility.XZ ( Vector3 vector ) [static]
```

Convert a vector3 to vector2 using the z axis instead of the y

Parameters

| vector | Vector. |

Definition at line 60 of file VectorUtility.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Utility/VectorUtility.cs
4.206 Voroni Class Reference

Collaboration diagram for Voroni:

```
+ Voroni
+ Center
+ Voroni()
  + GetEdges()
```

Public Member Functions

- **Voroni** (Vector2[] seeds, int max, int min)
  
  *Initializes a new instance of the Voroni class.*

- **Vector2[] GetEdges()**
  
  *Gets the edges based on the initial seeds and max/min span for this instance.*

Properties

- **Vector2 Center** [get, set]
  
  *Gets or sets the center of the Voroni diagram.*

4.206.1 Detailed Description

Definition at line 5 of file Voroni.cs.

4.206.2 Constructor & Destructor Documentation

4.206.2.1 Voroni()

```
Voroni.Voroni( 
    Vector2[] seeds, 
    int max, 
    int min )
```

*Initializes a new instance of the Voroni class.*
Parameters

<table>
<thead>
<tr>
<th>seeds</th>
<th>Seeds.</th>
</tr>
</thead>
<tbody>
<tr>
<td>max</td>
<td>Max.</td>
</tr>
<tr>
<td>min</td>
<td>Minimum.</td>
</tr>
</tbody>
</table>

Definition at line 18 of file Voroni.cs.

### 4.206.3 Member Function Documentation

#### 4.206.3.1 GetEdges()

```csharp
Vector2[] Voroni.GetEdges();
```

Gets the edges based on the initial seeds and max/min span for this instance.

Returns

The edges.

Definition at line 45 of file Voroni.cs.

### 4.206.4 Property Documentation

#### 4.206.4.1 Center

```csharp
Vector2 Voroni.Center [get], [set]
```

Gets or sets the center of the Voroni diagram.

The center.

Definition at line 30 of file Voroni.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/City/Voroni.cs
Inheritance diagram for WarmthIdolCategory:
Collaboration diagram for WarmthIdolCategory:

Public Member Functions

- override void `ReadyCategory()`
  
  Prep the category for use by loading attributes and actions into lists.

- override ItemCategory `GetDuplicate()`
  
  Gets a copy of the ItemCategory.

- override void `ApplyBenefit()`
Applies the warmth benefit.
* override void `RemoveBenefit()`
  Removes the warmth benefit.

Properties

* int `WarmthBenefit` [get, set]
  Gets or sets the warmth benefit.

Additional Inherited Members

4.207.1 Detailed Description

Definition at line 6 of file `WarmthIdolCategory.cs`.

4.207.2 Member Function Documentation

4.207.2.1 ApplyBenefit()

override void `WarmthIdolCategory.ApplyBenefit()` [virtual]

Applies the warmth benefit.

Implements `IdolCategory`.

Definition at line 58 of file `WarmthIdolCategory.cs`.

4.207.2.2 GetDuplicate()

override `ItemCategory` `WarmthIdolCategory.GetDuplicate()` [virtual]

Gets a copy of the `ItemCategory`.

Returns

  The duplicate.

Reimplemented from `EquipableCategory`.

Definition at line 34 of file `WarmthIdolCategory.cs`. 
4.207.2.3 ReadyCategory()

override void WarmthIdolCategory.ReadyCategory ( ) [virtual]

Preps the category for use by loading attributes and actions into lists.

Reimplemented from IdolCategory.

Definition at line 23 of file WarmthIdolCategory.cs.

4.207.2.4 RemoveBenefit()

override void WarmthIdolCategory.RemoveBenefit ( ) [virtual]

Removes the warmth benefit.

Implements IdolCategory.

Definition at line 66 of file WarmthIdolCategory.cs.

4.207.3 Property Documentation

4.207.3.1 WarmthBenefit

int WarmthIdolCategory.WarmthBenefit [get], [set]

Gets or sets the warmth benefit.

The warmth benefit.

Definition at line 13 of file WarmthIdolCategory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Item Crafting/Backend/WarmthIdolCategory.cs
Inheritance diagram for WarmthRateManager:
Collaboration diagram for WarmthRateManager:

Public Member Functions

- **WarmthRateManager** ()
  
  Initializes a new instance of the *WarmthRateManager* class.

- void **UseDefaultWarmthReductionRate** ()
  
  Uses the default warmth reduction rate.

- void **UseWaterWarmthReductionRate** ()
  
  Uses the water warmth reduction rate.

- void **UseHeatSourceWarmthIncreaseRate** ()
  
  Uses the heat source warmth increase rate.

- void **UseClothRate** (int clothUnits)
  
  Uses the cloth rate.

- void **SetUnitsInShelter** (int unit)

Additional Inherited Members

4.208.1 Detailed Description

Definition at line 3 of file *WarmthRateManager.cs.*
4.208.2 Constructor & Destructor Documentation

4.208.2.1 WarmthRateManager()

WarmthRateManager.WarmthRateManager ( )

Initializes a new instance of the WarmthRateManager class.
Definition at line 43 of file WarmthRateManager.cs.

4.208.3 Member Function Documentation

4.208.3.1 SetUnitsInShelter()

void WarmthRateManager.SetUnitsInShelter ( int unit )

Definition at line 169 of file WarmthRateManager.cs.

4.208.3.2 UseClothRate()

void WarmthRateManager.UseClothRate ( int clothUnits )

Uses the cloth rate.
Parameters

| units | Units |

Definition at line 104 of file WarmthRateManager.cs.

4.208.3.3 UseDefaultWarmthReductionRate()

void WarmthRateManager.UseDefaultWarmthReductionRate ( )

Uses the default warmth reduction rate.
Definition at line 59 of file WarmthRateManager.cs.
4.208.3.4 UseHeatSourceWarmthIncreaseRate()

```csharp
public void UseHeatSourceWarmthIncreaseRate()
```

Uses the heat source warmth increase rate.
Definition at line 84 of file `WarmthRateManager.cs`.

4.208.3.5 UseWaterWarmthReductionRate()

```csharp
public void UseWaterWarmthReductionRate()
```

Uses the water warmth reduction rate.
Definition at line 70 of file `WarmthRateManager.cs`.

The documentation for this class was generated from the following file:

- Assets/Scripts/Player/WarmthRateManager.cs

### 4.209 WaterItemGeneration Class Reference

Inheritance diagram for WaterItemGeneration:

```
MonoBehaviour

ItemGenerator
  # poolManager
  # districtItemInfo
  + GetItemExtents()
  + SetSeed()
  + AddTemplatesToItemPool()
```

```
WaterItemGeneration
  + SetCityInformation()
  + GenerateInWater()
  + AddBuildingToWaterGenerationMap()

```
Public Member Functions

- **void** `SetCityInformation` (float `cityWidth`, float `cityDepth`, Vector3 `cityCenter`, District[ ] `districts`)
  *Sets the city information. Must be called before points are generated.*

- **void** `GenerateInWater` ()
  *Generates the items in water.*

- **void** `AddBuildingToWaterGenerationMap` (Bounds `buildingBound`)
  *Adds the building to water point generation’s map of blocked off space.*
Additional Inherited Members

4.209.1 Detailed Description

Definition at line 5 of file WaterItemGeneration.cs.

4.209.2 Member Function Documentation

4.209.2.1 AddBuildingToWaterGenerationMap()

```csharp
void WaterItemGeneration.AddBuildingToWaterGenerationMap ( Bounds buildingBound )
```

Adds the building to water point generation's map of blocked off space.

Parameters

- `buildingBound`: Building bound.

Definition at line 94 of file WaterItemGeneration.cs.

4.209.2.2 GenerateInWater()

```csharp
void WaterItemGeneration.GenerateInWater ( )
```

Generates the items in water.

Definition at line 70 of file WaterItemGeneration.cs.

4.209.2.3 SetCityInformation()

```csharp
void WaterItemGeneration.SetCityInformation ( float cityWidth,
                                          float cityDepth,
                                          Vector3 cityCenter,
                                          District [] districts )
```

Sets the city information. Must be called before points are generated.

Parameters

- `cityWidth`: City width.
- `cityDepth`: City depth.
- `districts`: Districts.
Definition at line 56 of file WaterItemGeneration.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/WaterItemGeneration.cs

### 4.210 WaterMovement Class Reference

Inheritance diagram for WaterMovement:
Collaboration diagram for WaterMovement:

```
MonoBehaviour

Movement
+ Speed
+ CurrentFallDamage
# Rigidbody
# AccumulatedFallDamage
# climbHeight
+ Idle()
+ Move()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ OnStateEnter()
+ OnStateExit()
+ GetRaycastHeight()

FXSplashManager
+ SplashPoolSize
  + FillPool()
  + CreateSplash()
  + CreateSplash()

+ SplashManager

WaterMovement
+ SwimmingHeight
+ Idle()
+ Move()
+ Jump()
+ Climb()
+ GetClimbHeight()
+ GetRaycastHeight()
+ OnStateEnter()
+ OnStateExit()
```

Public Member Functions

- override void **Idle** (Animator playerAnimator)
  
  *Will run the idle swim animation in the future*

- override void **Move** (Vector3 direction, bool sprinting, Animator playerAnimator)
  
  *Player swims in the specified direction.*

- override void **Jump** (Animator playerAnimator)
  
  *Player dives into water.*

- override void **Climb** (Animator playerAnimator)
  
  *The player's climb animation plays*


- `override float GetClimbHeight ()`
  The height the player can climb while in this movement state

- `override float GetRaycastHeight ()`
  The height of the climbing raycast while in this movement state

- `override void OnStateEnter ()`
  Called when the player enters the state.

- `override void OnStateExit ()`
  Called when the player exits the state.

**Public Attributes**

- `FXSplashManager SplashManager`

**Properties**

- `float SwimmingHeight [get]`

**Additional Inherited Members**

### 4.210.1 Detailed Description

Definition at line 6 of file WaterMovement.cs.

### 4.210.2 Member Function Documentation

#### 4.210.2.1 Climb()

```cpp
override void WaterMovement.Climb ( Animator playerAnimator ) [virtual]
```

The player's climb animation plays

**Parameters**

- `playerAnimator` | The player's animator

Implements `Movement`.

Definition at line 98 of file WaterMovement.cs.
4.210.2.2 GetClimbHeight()

```csharp
override float WaterMovement.GetClimbHeight () [virtual]
```

The height the player can climb while in this movement state

Implements Movement.

Definition at line 106 of file WaterMovement.cs.

4.210.2.3 GetRaycastHeight()

```csharp
override float WaterMovement.GetRaycastHeight () [virtual]
```

The height of the climbing raycast while in this movement state

Implements Movement.

Definition at line 114 of file WaterMovement.cs.

4.210.2.4 Idle()

```csharp
override void WaterMovement.Idle ( Animator playerAnimator ) [virtual]
```

Will run the idle swim animation in the future

Parameters

- **playerAnimator**: The player's animator

Implements Movement.

Definition at line 59 of file WaterMovement.cs.

4.210.2.5 Jump()

```csharp
override void WaterMovement.Jump ( Animator playerAnimator ) [virtual]
```

**Player** dives into water.
Parameters

| playerAnimator | The player's animator |

Implements Movement.

Definition at line 89 of file WaterMovement.cs.

4.210.2.6 Move()

```csharp
override void WaterMovement.Move ( 
    Vector3 direction, 
    bool sprinting, 
    Animator playerAnimator ) [virtual]
```

Player swims in the specified direction.

Parameters

| direction | The player's move direction |
| sprinting | Is the player sprinting |
| playerAnimator | The player's animator |

Implements Movement.

Definition at line 72 of file WaterMovement.cs.

4.210.2.7 OnStateEnter()

```csharp
override void WaterMovement.OnStateEnter ( ) [virtual]
```

Called when the player enters the state.

Implements Movement.

Definition at line 123 of file WaterMovement.cs.

4.210.2.8 OnStateExit()

```csharp
override void WaterMovement.OnStateExit ( ) [virtual]
```

Called when the player exits the state.

Implements Movement.

Definition at line 134 of file WaterMovement.cs.
4.210.3 Member Data Documentation

4.210.3.1 SplashManager

FXSplashManager WaterMovement.SplashManager

Definition at line 19 of file WaterMovement.cs.

4.210.4 Property Documentation

4.210.4.1 SwimmingHeight

float WaterMovement.SwimmingHeight [get]

Definition at line 29 of file WaterMovement.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/WaterMovement.cs
4.211  WaterPointGenerator Class Reference

Inheritance diagram for WaterPointGenerator:
Collaboration diagram for WaterPointGenerator:

```
ItemPlacementSamplePoint
+ LocalTargetSurfaceLocation
+ MinDistance
+ GridPoint
+ WorldSpaceLocation
+ ItemIndex
+ Size
+ District
+ Type

#grid

SamplingPointGenerator
# cellSize
# waterTag
# defaultMinDistanceAway

# generateRandomPointAround()
# HasOverlappingNeighbors()
# verifyRaycastHit()

WaterPointGenerator
+ MaxNumberOfItems
+ NumberOfInitialPoints
+ NewPointsPerSamplingPoint
+ MaxAttempts
+ DistrictCheck
+ Districts

+ WaterPointGenerator()
+ AddBuildingBounds()
+ GetPointsInWater()
+ generateInitialPoints()
# PointToGrid()
```

Public Member Functions

- **WaterPointGenerator** (float width, float depth, Vector3 center)
  
  Initializes a new instance of the WaterPointGenerator class.

- **void AddBuildingBounds** (Bounds bound)
  
  Adds the building bounds to the grid of occupied space.

- **List<ItemPlacementSamplePoint> GetPointsInWater** (Dictionary<string, List<float>> generatable
  
  ItemExtents)
Class Documentation

Gets the sampling points in water.

- List<ItemPlacementSamplePoint> generateInitialPoints(Dictionary<string, List<float>> generatable, ObjectExtents)

  Generates the initial sampling points.

Protected Member Functions

- Tuple<int, int> PointToGrid(Vector2 samplingPoint)

  Gets the grid coordinates for a sampling point. Since the point locations are not created with the minimum bounds being (0, 0) in mind, it needs to be offset such that the minimum bound is at (0, 0). This can be done by adding half the dimension of the city and the center of the city.

Properties

- int MaxNumberOfItems [get, set]
  Gets or sets the max number of items.
- int NumberOfInitialPoints [get, set]
  Generally, the more initial points generated, the more evenly spread out the items will be.
- int NewPointsPerSamplingPoint [get, set]
  Generally, the more points per previous point and less number of initial points, the more clustered item generation will be.
- int MaxAttempts [get, set]
  The number of attempts to place a point. Generally, the higher the number, the closer to the maximum number of items will be generated.
- int DistrictCheck [get, set]
  How many times it will generate an item for a point without checking for district before checking for district again.
- District[] Districts [get, set]
  Gets or sets the districts.

Additional Inherited Members

4.211.1 Detailed Description

Definition at line 5 of file WaterPointGenerator.cs.

4.211.2 Constructor & Destructor Documentation

4.211.2.1 WaterPointGenerator()

WaterPointGenerator.WaterPointGenerator (float width, float depth, Vector3 center)

Initializes a new instance of the WaterPointGenerator class.
Parameters

<table>
<thead>
<tr>
<th>width</th>
<th>City width.</th>
</tr>
</thead>
<tbody>
<tr>
<td>depth</td>
<td>City depth.</td>
</tr>
<tr>
<td>center</td>
<td>City center.</td>
</tr>
</tbody>
</table>

Definition at line 90 of file WaterPointGenerator.cs.

4.211.3 Member Function Documentation

4.211.3.1 AddBuildingBounds()

```csharp
void WaterPointGenerator.AddBuildingBounds (Bounds bound)
```

Adds the building bounds to the grid of occupied space.

Parameters

| bound | Bound. |

Definition at line 107 of file WaterPointGenerator.cs.

4.211.3.2 generateInitialPoints()

```csharp
List<ItemPlacementSamplePoint> WaterPointGenerator.generateInitialPoints (Dictionary<string, List<float>> generatableObjectExtents)
```

Generates the initial sampling points.

Returns

The initial points.

Parameters

| generatableObjectExtents | Generatable object extents. |

Definition at line 261 of file WaterPointGenerator.cs.
4.211.3.3 GetPointsInWater()

```csharp
List<ItemPlacementSamplePoint> WaterPointGenerator.GetPointsInWater(
    Dictionary<string, List<float>> generatableItemExtents)
```

Gets the sampling points in water.

Returns

The points in water.

Parameters

- `generatableItemExtents` Generatable item extents.

Definition at line 142 of file WaterPointGenerator.cs.

4.211.3.4 PointToGrid()

```csharp
Tuple<int, int> WaterPointGenerator.PointToGrid(
    Vector2 samplingPoint) [protected]
```

Gets the grid coordinates for a sampling point Since the point locations are not created with the minimum bounds being (0, 0) in mind it needs to be offset such that the minimum bound is at (0, 0) Which can be done by adding half the dimension of the city and the center of the city

Returns

The to grid.

Parameters

- `samplingPoint` Sampling point.

Definition at line 359 of file WaterPointGenerator.cs.

4.211.4 Property Documentation

4.211.4.1 DistrictCheck

```csharp
int WaterPointGenerator.DistrictCheck [get], [set]
```

How many times it will generate an item for a point without checking for district before checking for district again

Definition at line 50 of file WaterPointGenerator.cs.
4.211.2 Districts

District [] WaterPointGenerator.Districts [get], [set]

Gets or sets the districts.

The districts.

Definition at line 60 of file WaterPointGenerator.cs.

4.211.3 MaxAttempts

int WaterPointGenerator.MaxAttempts [get], [set]

The number of attempts to place a point. Generally, the higher the number, the closer to the maximum number of items will be generated.

Definition at line 41 of file WaterPointGenerator.cs.

4.211.4 MaxNumberOfItems

int WaterPointGenerator.MaxNumberOfItems [get], [set]

Gets or sets the max number of items.

The max number of items.

Definition at line 12 of file WaterPointGenerator.cs.

4.211.5 NewPointsPerSamplingPoint

int WaterPointGenerator.NewPointsPerSamplingPoint [get], [set]

Generally, the more points per previous point and less number of initial points, the more clustered item generation will be.

The new points per sampling point.

Definition at line 32 of file WaterPointGenerator.cs.
4.211.4.6 NumberofInitialPoints

```csharp
int WaterPointGenerator.NumberOfInitialPoints [get], [set]
```

Generally, the more initial points generated, the more evenly spread out the items will be

The number of initial points.

Definition at line 22 of file WaterPointGenerator.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts(ItemProceduralGeneration/WaterPointGenerator.cs

4.212 WeatherSoundSystem Class Reference

Inheritance diagram for WeatherSoundSystem:

```
MonoBehaviour

WeatherSoundSystem
+ DefaultWeatherEventPath
```
Collaboration diagram for WeatherSoundSystem:

Public Attributes

- string DefaultWeatherEventPath = "event:/Ambient/Weather/Basic_Rain"
  The default weather event path.

4.212.1 Detailed Description

Definition at line 4 of file WeatherSoundSystem.cs.

4.212.2 Member Data Documentation

4.212.2.1 DefaultWeatherEventPath

string WeatherSoundSystem.DefaultWeatherEventPath = "event:/Ambient/Weather/Basic_Rain"

The default weather event path.

Definition at line 9 of file WeatherSoundSystem.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/WeatherSoundSystem.cs

Generated by Doxygen
4.213 WeatherSystem Class Reference

Collaboration diagram for WeatherSystem:

```
<table>
<thead>
<tr>
<th>WeatherSystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ WindDirection2d</td>
</tr>
<tr>
<td>+ WindDirection3d</td>
</tr>
<tr>
<td>+ NormalizedOctantWindDirection2d</td>
</tr>
<tr>
<td>+ WeatherPressureSystems</td>
</tr>
<tr>
<td>+ WeatherInformation</td>
</tr>
<tr>
<td>+ UpdateWeather()</td>
</tr>
<tr>
<td>+ EnableWeather()</td>
</tr>
<tr>
<td>+ DisableWeather()</td>
</tr>
<tr>
<td>+ ToString()</td>
</tr>
<tr>
<td>+ UpdateSystem()</td>
</tr>
<tr>
<td>+ WeatherSystem()</td>
</tr>
</tbody>
</table>
```

Public Member Functions

- void `UpdateWeather(Vector3 positionVector)`
  Updates the weather array variable
- void `EnableWeather()`
  Enables the weather updates.
- void `DisableWeather()`
  Disables the weather updates.
- override string `ToString()`
  Returns a System.String that represents the current WeatherSystem.
- void `UpdateSystem()`
  Updates the weather and internal pressure systems.
- `WeatherSystem(CityBoundaries bounds, PauseSystem pauseController)`
  Initializes a new instance of the WeatherSystem class.

Properties

- Vector2 `WindDirection2d [get]`
  Gets the wind direction in 2d.
- Vector3 `WindDirection3d [get]`
  Gets the wind direction in 3d.
- Vector2 `NormalizedOctantWindDirection2d [get]`
  Gets the cartesian wind direction2d.
- `PressureSystems WeatherPressureSystems [get]`
- `float [] WeatherInformation [get]`
4.213 WeatherSystem Class Reference

4.213.1 Detailed Description

Definition at line 21 of file WeatherSystem.cs.

4.213.2 Constructor & Destructor Documentation

4.213.2.1 WeatherSystem()

WeatherSystem.WeatherSystem ( 
    CityBoundaries bounds,
    PauseSystem pauseController )

Initializes a new instance of the WeatherSystem class.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bounds</td>
<td>Bounds of the city.</td>
</tr>
<tr>
<td>pauseController</td>
<td>Pause system instance.</td>
</tr>
</tbody>
</table>

Definition at line 470 of file WeatherSystem.cs.

4.213.3 Member Function Documentation

4.213.3.1 DisableWeather()

void WeatherSystem.DisableWeather ( )

Disables the weather updates.

Definition at line 428 of file WeatherSystem.cs.

4.213.3.2 EnableWeather()

void WeatherSystem.EnableWeather ( )

Enables the weather updates.

Definition at line 420 of file WeatherSystem.cs.
4.213.3 ToString()

```csharp
override string WeatherSystem.ToString()
```

Returns a System.String that represents the current WeatherSystem.

Definition at line 437 of file WeatherSystem.cs.

4.213.4 UpdateSystem()

```csharp
void WeatherSystem.UpdateSystem()
```

Updates the weather and internal pressure systems.

Definition at line 456 of file WeatherSystem.cs.

4.213.5 UpdateWeather()

```csharp
void WeatherSystem.UpdateWeather(
    Vector3 positionVector)
```

Updates the weather array variable

Parameters

| position | Position |

Definition at line 397 of file WeatherSystem.cs.

4.213.4 Property Documentation

4.213.4.1 NormalizedOctantWindDirection2d

```csharp
Vector2 WeatherSystem.NormalizedOctantWindDirection2d [get]
```

Gets the cartesian wind direction2d.
The cartesian wind direction2d.
Definition at line 108 of file WeatherSystem.cs.
4.213.4.2 WeatherInformation

float [] WeatherSystem.WeatherInformation [get]

Definition at line 133 of file WeatherSystem.cs.

4.213.4.3 WeatherPressureSystems

PressureSystems WeatherSystem.WeatherPressureSystems [get]

Definition at line 127 of file WeatherSystem.cs.

4.213.4.4 WindDirection2d

Vector2 WeatherSystem.WindDirection2d [get]

Gets the wind direction in 2d.

The wind direction.

Definition at line 81 of file WeatherSystem.cs.

4.213.4.5 WindDirection3d

Vector3 WeatherSystem.WindDirection3d [get]

Gets the wind direction in 3d.

The wind direction3d.

Definition at line 94 of file WeatherSystem.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Weather/WeatherSystem.cs

Generated by Doxygen
4.214  WeightedPair Class Reference

Collaboration diagram for WeightedPair:

```
WeightedPair
+ First
+ Second
+ Threshold
+ WeightedPair()
```

Public Member Functions

- **WeightedPair** (int first, int second, float threshold)
  
  Initializes a new instance of the WeightedPair class.

Properties

- int **First** [get, set]
  
  Gets or sets the first value of the weighted pair.
- int **Second** [get, set]
  
  Gets or sets the second value of the weighted pair.
- float **Threshold** [get, set]
  
  Gets or sets the threshold value that determines whether the first or second value is selected. The higher the threshold, the more likely the first value is selected.

4.214.1  Detailed Description

Definition at line 4 of file WeightedPair.cs.

4.214.2  Constructor & Destructor Documentation

4.214.2.1  WeightedPair()

WeightedPair.WeightedPair (
    int first,
    int second,
    float threshold )

Initializes a new instance of the WeightedPair class.
4.214 WeightedPair Class Reference

Parameters

<table>
<thead>
<tr>
<th>first</th>
<th>First index value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>second</td>
<td>Second index value.</td>
</tr>
<tr>
<td>threshold</td>
<td>A higher threshold value will result in a lower chance of the second value being selected.</td>
</tr>
</tbody>
</table>

Definition at line 12 of file WeightedPair.cs.

4.214.3 Property Documentation

4.214.3.1 First

```csharp
int WeightedPair.First [get], [set]
```

Gets or sets the first value of the weighted pair.

The first.

Definition at line 24 of file WeightedPair.cs.

4.214.3.2 Second

```csharp
int WeightedPair.Second [get], [set]
```

Gets or sets the second value of the weighted pair.

The second.

Definition at line 34 of file WeightedPair.cs.

4.214.3.3 Threshold

```csharp
float WeightedPair.Threshold [get], [set]
```

Gets or sets the threshold value that determines whether the first or second value is selected. The higher the threshold, the more likely the first value is selected.

The threshold.

Definition at line 45 of file WeightedPair.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/WeightedPair.cs

Generated by Doxygen
Inheritance diagram for WhaleSpawner:
Collaboration diagram for WhaleSpawner:

Protected Member Functions

- override Vector3 findSpawnLocation ()
  
  Get a location where the unit can be spawned

Additional Inherited Members

4.215.1 Detailed Description

Definition at line 4 of file WhaleSpawner.cs.

4.215.2 Member Function Documentation
4.215.2.1 findSpawnLocation()

```csharp
override Vector3 WhaleSpawner.findSpawnLocation() [protected], [virtual]
```

Get a location where the unit can be spawned

Returns

The location.

Reimplemented from `CreatureSpawner`.

Definition at line 10 of file `WhaleSpawner.cs`.

The documentation for this class was generated from the following file:

- `Assets/Scripts/Creatures/WhaleSpawner.cs`

### 4.216 WindMovement Class Reference

Inheritance diagram for `WindMovement`:

```
MonoBehaviour

WindMovement
```

Generated by Doxygen
Collaboration diagram for WindMovement:

```
MonoBehaviour

WindMovement
```

4.216.1 Detailed Description

Definition at line 6 of file WindMovement.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Movement/Creatures/WindMovement.cs

4.217 WindowWasher Class Reference

Goes on the window washer prefab, handles the ropes on the washer, moving it up with the water, and teleporting the player.
Inheritance diagram for WindowWasher:

```
MonoBehaviour

WindowWasher
+ MaxLowerDistance
+ StartUp
+ WasherBaseLength
+ PlacementCenter
+ Start()
+ Update()
+ TeleportPlayer()
+ UpdateRopePosition()
```

Collaboration diagram for WindowWasher:

```
MonoBehaviour

WindowWasher
+ MaxLowerDistance
+ StartUp
+ WasherBaseLength
+ PlacementCenter
+ Start()
+ Update()
+ TeleportPlayer()
+ UpdateRopePosition()
```
Public Member Functions

- void Start ()
  
  Set up the rope material on start.

- void Update ()

- void TeleportPlayer ()
  
  Moves the player object and the window washer either up or down.

- void UpdateRopePosition ()
  
  When the window washer moves the rope needs to update to reflect this.

Public Attributes

- float MaxLowerDistance

- bool StartUp

Properties

- float WasherBaseLength [get]

- GameObject PlacementCenter [get]

4.217.1 Detailed Description

Goes on the window washer prefab, handles the ropes on the washer, moving it up with the water, and teleporting the player.

TODO: Change window washer to not teleport and jump up and down, but to travel up and down over time carrying the player with it.

Definition at line 9 of file WindowWasher.cs.

4.217.2 Member Function Documentation

4.217.2.1 Start()

void WindowWasher.Start ()

Set up the rope material on start.

Definition at line 70 of file WindowWasher.cs.

Generated by Doxygen
4.217.2.2  TeleportPlayer()

void WindowWasher.TeleportPlayer ( )

Moves the player object and the window washer either up or down.

Definition at line 143 of file WindowWasher.cs.

4.217.2.3  Update()

void WindowWasher.Update ( )

Definition at line 95 of file WindowWasher.cs.

4.217.2.4  UpdateRopePosition()

void WindowWasher.UpdateRopePosition ( )

When the window washer moves the rope needs to update to reflect this.

Definition at line 179 of file WindowWasher.cs.

4.217.3  Member Data Documentation

4.217.3.1  MaxLowerDistance

float WindowWasher.MaxLowerDistance

Definition at line 11 of file WindowWasher.cs.

4.217.3.2  StartUp

bool WindowWasher.StartUp

Definition at line 64 of file WindowWasher.cs.

4.217.4  Property Documentation
4.217.1 PlacementCenter

GameObject WindowWasher.PlacementCenter [get]

Definition at line 57 of file WindowWasher.cs.

4.217.2 WasherBaseLength

float WindowWasher.WasherBaseLength [get]

Definition at line 47 of file WindowWasher.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/Teleporters/WindowWasher.cs

4.218 WorldItemFactory Class Reference

Collaboration diagram for WorldItemFactory:

```
+ WorldItemFactory()
+ LoadTemplates()
+ CreatePickUpInteractableItem()
+ CreateRandomPickupInteractableItem()
+ CreateGenericInteractableItem()
+ GetAllInteractableItemsByDistrict()
+ GetRandomItemIndex()
```

Public Member Functions

- **WorldItemFactory ()**
  
  Initializes a new instance of the WorldItemFactory class.

- **void LoadTemplates ()**
  
  Loads the item templates.

- **GameObject CreatePickUpInteractableItem (BaseItem itemToCreate, int amount)**
  
  Creates the interactable item that is ready to be placed in the world.

- **GameObject CreateRandomPickupInteractableItem (string district)**
Creates a random interactable item that is ready to be placed in the world.

- **GameObject** CreateGenericInteractableItem (BaseItem itemToCreate)
  
  Creates a generic interactable item. Does not add on the interactable script, but sets up item with assumption that player will attach one.

- **Dictionary<string, List<GameObject>>** GetAllInteractableItemsByDistrict (bool setActive, bool water)
  
  Gets all interactable items by district. GetRandomItemIndex can be used to get a random item from the Dictionary returned by this function.

- **int** GetRandomItemIndex (string district, bool onWater)
  
  Gets the random index that can be used to access items in a district. Order is by the Dictionary returned by GetAllInteractableItemsByDistrict.

### 4.218.1 Detailed Description

Definition at line 5 of file WorldItemFactory.cs.

### 4.218.2 Constructor & Destructor Documentation

#### 4.218.2.1 WorldItemFactory()

WorldItemFactory.WorldItemFactory ( )

Initializes a new instance of the WorldItemFactory class.

Definition at line 39 of file WorldItemFactory.cs.

### 4.218.3 Member Function Documentation

#### 4.218.3.1 CreateGenericInteractableItem()

GameObject WorldItemFactory.CreateGenericInteractableItem (BaseItem itemToCreate)

Creates a generic interactable item. Does not add on the interactable script, but sets up item with assumption that player will attach one.

**Returns**

The generic interactable item.

**Parameters**

- **itemToCreate** Item to create.
4.218.3.2 CreatePickUpInteractableItem()

GameObject WorldItemFactory.CreatePickUpInteractableItem ( BaseItem itemToCreate, int amount )

Creates the interactable item that is ready to be placed in the world.

Returns
The interactable item.

Parameters

<table>
<thead>
<tr>
<th>itemToCreate</th>
<th>Item to create.</th>
</tr>
</thead>
<tbody>
<tr>
<td>amount</td>
<td>Amount.</td>
</tr>
</tbody>
</table>

4.218.3.3 CreateRandomPickupInteractableItem()

GameObject WorldItemFactory.CreateRandomPickupInteractableItem ( string district )

Creates a random interactable item that is ready to be placed in the world.

Returns
A random interactable item.

4.218.3.4 GetAllInteractableItemsByDistrict()

Dictionary<string, List<GameObject>> WorldItemFactory.GetAllInteractableItemsByDistrict ( bool setActive, bool water )

Gets all interactable items by district. GetRandomItemIndex can be used to get a random item from the Dictionary returned by this function.

Returns
The all interactable items by district.
Parameters

| setActive | If set to true, gameobjects are active when created. |
| water     | If set to true, gets objects for water. |

Definition at line 147 of file WorldItemFactory.cs.

### 4.218.3.5 GetRandomItemIndex()

```csharp
int WorldItemFactory.GetRandomItemIndex (
    string district,
    bool onWater )
```

Gets the random index that can be used to access items in a district. Order is by the Dictionary returned by GetAllInteractableItemsByDistrict.

**Returns**

The random item index.

Parameters

| district | District. |
| onWater  | Whether or not items are generating from water. |

Definition at line 182 of file WorldItemFactory.cs.

### 4.218.3.6 LoadTemplates()

```csharp
void WorldItemFactory.LoadTemplates ( )
```

Loads the item templates.

Definition at line 49 of file WorldItemFactory.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/ItemProceduralGeneration/WorldItemFactory.cs
**Public Member Functions**

- void **SetAction** (UnityAction action)
  
  *Sets the action to the button's onclick listener.*

**Properties**

- string **ButtonName** [get, set]
  
  *Gets or sets the name of the item.*
4.219.1 Detailed Description

Definition at line 6 of file WorldSelectionButtonBehavior.cs.

4.219.2 Member Function Documentation

4.219.2.1 `SetAction()`

```csharp
void WorldSelectionButtonBehavior.SetAction (UnityAction action)
```

Sets the action to the button’s onclick listener.

Parameters

- `action` Action.

Definition at line 36 of file WorldSelectionButtonBehavior.cs.

4.219.3 Property Documentation

4.219.3.1 `ButtonName`

```csharp
string WorldSelectionButtonBehavior.ButtonName [get], [set]
```

Gets or sets the name of the item.

The name of the item.

Definition at line 21 of file WorldSelectionButtonBehavior.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/WorldInteraction/WorldSelectionButtonBehavior.cs
Public Member Functions

- void **DisplayItemOptions** (List<string> itemTypes, **OverworldItemOptionSelection** requestingItem, Game<Object targetPanel=null)
Displays the item options.

- void DisplayActions (List<ItemAction> actions, OverworldItemOptionSelection requestingItem, GameObject targetPanel=null)

  Displays the items possible actions.

- void FinishSelection (string selectedItem)

  Finishes the selection.

- void CloseSelection (GameObject targetContent, GameObject targetHolder)

  Closes the selection.

- void ClearOptions (GameObject targetPanel)

  Clears the options from the target panel.

4.220.1 Detailed Description

Definition at line 7 of file WorldSelectionGUIDirector.cs.

4.220.2 Member Function Documentation

4.220.2.1 ClearOptions()

 void WorldSelectionGUIDirector.ClearOptions ( GameObject targetPanel )

Clears the options from the target panel.

Parameters

- **targetPanel** Target panel.

Definition at line 169 of file WorldSelectionGUIDirector.cs.

4.220.2.2 CloseSelection()

 void WorldSelectionGUIDirector.CloseSelection ( GameObject targetContent,
                                               GameObject targetHolder )

Closes the selection.

Parameters

- **targetContent** Target content panel that contains options.
- **targetHolder** Target holder panel that contains the display panel.
4.220 WorldSelectionGUIDirector Class Reference

Definition at line 150 of file WorldSelectionGUIDirector.cs.

4.220.2.3 DisplayActions()

void WorldSelectionGUIDirector.DisplayActions (  
    List<ItemAction> actions,  
    OverworldItemOptionSelection requestingItem,  
    GameObject targetPanel = null )

Displays the items possible actions.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>actions</td>
<td>Actions that should be displayed.</td>
</tr>
<tr>
<td>requestingItem</td>
<td>Requesting item that stores the selected options.</td>
</tr>
<tr>
<td>targetPanel</td>
<td>Target panel where the options will appear.</td>
</tr>
</tbody>
</table>

Definition at line 78 of file WorldSelectionGUIDirector.cs.

4.220.2.4 DisplayItemOptions()

void WorldSelectionGUIDirector.DisplayItemOptions (  
    List<string> itemTypes,  
    OverworldItemOptionSelection requestingItem,  
    GameObject targetPanel = null )

Displays the item options.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>itemTypes</td>
<td>Items of types that should be displayed.</td>
</tr>
<tr>
<td>requestingItem</td>
<td>Requesting item that stores selected options.</td>
</tr>
<tr>
<td>targetPanel</td>
<td>Target panel where the options will appear.</td>
</tr>
</tbody>
</table>

Definition at line 45 of file WorldSelectionGUIDirector.cs.

4.220.2.5 FinishSelection()

void WorldSelectionGUIDirector.FinishSelection (  
    string selectedItem )

Finishes the selection.
Parameters

| selectedItem | Selected. |

Definition at line 140 of file WorldSelectionGUIDirector.cs.

The documentation for this class was generated from the following file:

- Assets/Scripts/GUI/WorldInteraction/WorldSelectionGUIDirector.cs
Chapter 5

File Documentation

5.1 Assets/Scripts/AmbientSoundManager.cs File Reference

Classes

• class AmbientSoundManager

5.2 Assets/Scripts/Camera/CameraController.cs File Reference

Classes

• class CameraController

5.3 Assets/Scripts/City/Block.cs File Reference

Classes

• class Block

5.4 Assets/Scripts/City/BlockGenerator.cs File Reference

Classes

• class BlockGenerator

5.5 Assets/Scripts/City/Building.cs File Reference

Classes

• class Building
5.6 Assets/Scripts/City/BuildingGenerator.cs File Reference

Classes

- class BuildingGenerator

5.7 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingAttachment.cs File Reference

Classes

- class ProceduralBuildingAttachment
  
  Used to define important parts of building attachments that are used during building creation and item placement.

5.8 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingBase.cs File Reference

Classes

- class ProceduralBuildingBase
  
  Defines parts of the building base that are needed for procedural building generation.

5.9 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingCreator.cs File Reference

Classes

- struct buildingIndex
- class ProceduralBuildingCreator
  
  Defines all the buildings for each district and allows for the creation of buildings in those districts.

Enumerations

- enum HeightType { HeightType.Stretch, HeightType.Stack }
- enum BaseSize { BaseSize.OneByOne, BaseSize.TwoByTwo, BaseSize.ThreeByThree }
- enum BuildingParts { BuildingParts.Base, BuildingParts.Roof }

5.9.1 Enumeration Type Documentation

5.9.1.1 BaseSize

enum BaseSize [strong]
5.10 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingInstance.cs File Reference

### Enumerator

<table>
<thead>
<tr>
<th>Enumerator</th>
</tr>
</thead>
<tbody>
<tr>
<td>OneByOne</td>
</tr>
<tr>
<td>TwoByTwo</td>
</tr>
<tr>
<td>ThreeByThree</td>
</tr>
</tbody>
</table>

Definition at line 12 of file ProceduralBuildingCreator.cs.

#### 5.9.1.2 BuildingParts

```csharp
enum BuildingParts
{
    Base,
    Roof
};
```

Definition at line 19 of file ProceduralBuildingCreator.cs.

#### 5.9.1.3 HeightType

```csharp
enum HeightType
{
    Stretch,
    Stack
};
```

Definition at line 6 of file ProceduralBuildingCreator.cs.

5.10 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingInstance.cs File Reference

### Classes

- class ProceduralBuildingInstance

  This class holds all the info that comes with making a procedural building

Generated by Doxygen
5.11 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingRoof.cs File Reference

Classes

- class ProceduralBuildingRoof
  
  Attached to the roof prefabs. Holds roof specific information needed when creating buildings.

5.12 Assets/Scripts/City/BuildingProceduralGeneration/ProceduralBuildingWindow.cs File Reference

Classes

- class ProceduralBuildingWindow
  
  Attached to the window prefabs. Holds window specific information needed when creating buildings.

5.13 Assets/Scripts/City/City.cs File Reference

Classes

- class City
- class City.BuildingEnumerator
  
  Class for creating an iterator to traverse the tree-structure of buildings in the City.

5.14 Assets/Scripts/City/CityBoundaries.cs File Reference

Classes

- class CityBoundaries

5.15 Assets/Scripts/City/CityController.cs File Reference

Classes

- class CityController

5.16 Assets/Scripts/City/DebugCity.cs File Reference

Classes

- class DebugCity
5.17 Assets/Scripts/City/District.cs File Reference

Classes

- class District

5.18 Assets/Scripts/City/DistrictDefinition.cs File Reference

Classes

- class DistrictConfiguration
  
  This class holds all the relevant info needed to define a district. It holds things like the building pieces the district contains, that districts materials, window types, etc.

- class DistrictConfiguration.BuildingTemplatePlacement
  
  Structure for accessing additional building placement configuration.

- struct DistrictConfiguration.ProceduralBuildingBasesSizeXDefinition
  
  Holds all of the bases and roofs for each size of building.

5.19 Assets/Scripts/City/DistrictGenerator.cs File Reference

Classes

- class DistrictGenerator

5.20 Assets/Scripts/City/DistrictPopupAttribute.cs File Reference

Classes

- class DistrictPopupAttribute

5.21 Assets/Scripts/City/GenerationUtility.cs File Reference

Classes

- class GenerationUtility

5.22 Assets/Scripts/City/Management/CityChunk.cs File Reference

Classes

- class CityChunk
  
  Manages loading and unloading a chunk of buildings
5.23 Assets/Scripts/City/Management/CityChunkManager.cs File Reference

Classes

- class CityChunkManager
  
  *Used to manage loading and unloading city building chunks according to player location in world.*

5.24 Assets/Scripts/City/ProceduralBuilding.cs File Reference

Classes

- class ProceduralBuilding
  
  *A building created procedurally.*

5.25 Assets/Scripts/City/TemplateBuilding.cs File Reference

Classes

- class TemplateBuilding
  
  *A building based on a prefab.*

5.26 Assets/Scripts/City/Voroni.cs File Reference

Classes

- class Voroni

5.27 Assets/Scripts/Creatures/CreatureManager.cs File Reference

Classes

- class CreatureManager

5.28 Assets/Scripts/Creatures/CreatureSpawner.cs File Reference

Classes

- class CreatureSpawner
5.29 Assets/Scripts/Creatures/CreatureTracker.cs File Reference

Classes

- class CreatureTracker

5.30 Assets/Scripts/Creatures/FishSpawner.cs File Reference

Classes

- class FishSpawner

5.31 Assets/Scripts/Creatures/JellyFishSpawner.cs File Reference

Classes

- class JellyFishSpawner

5.32 Assets/Scripts/Creatures/UI/SetCreatureGUI.cs File Reference

Classes

- class SetCreatureGUI

5.33 Assets/Scripts/Creatures/WhaleSpawner.cs File Reference

Classes

- class WhaleSpawner

5.34 Assets/Scripts/DeathScreen/DeathFlavorText.cs File Reference

Classes

- class DeathFlavorText

5.35 Assets/Scripts/DeathScreen/DeathManager.cs File Reference

Classes

- class DeathManager
### 5.36 Assets/Scripts/DeathScreen/RestartBehavior.cs File Reference

Classes

- class `RestartBehavior`

### 5.37 Assets/Scripts/Editor/BaseItemPopup.cs File Reference

Classes

- class `BaseItemPopup`

### 5.38 Assets/Scripts/Editor/DistrictPopup.cs File Reference

Classes

- class `DistrictPopup`

### 5.39 Assets/Scripts/Editor/uGUITools.cs File Reference

Classes

- class `uGUITools`

### 5.40 Assets/Scripts/Effects/DayNightSkybox.cs File Reference

Classes

- class `DayNightSkybox`

  This script modifies the `_Blend` variable in the custom shader, `Skybox/Blended` based on the `Game.Insance`'s clock.

### 5.41 Assets/Scripts/Effects/DebrisController.cs File Reference

Classes

- class `DebrisController`

  This class handles particle effects for miscellaneous debris, such as sand, leaves, dirt, and paper, in relation to them blowing around in the wind during the game. This is structured very similarly to the `RainController` class, except that this only handles one particle system at a time, without blending.
Classes
   • class FXSplash

5.43 Assets/Scripts/Effects/FXSplashManager.cs File Reference

Classes
   • class FXSplashManager

5.44 Assets/Scripts/Effects/RainController.cs File Reference

Classes
   • class RainController
    
    This class handles the rain effect that is present throughout the entire game.

5.45 Assets/Scripts/Ending/BeaconInteractable.cs File Reference

Classes
   • class BeaconInteractable

5.46 Assets/Scripts/Ending/EndingController.cs File Reference

Classes
   • class EndingController

5.47 Assets/Scripts/Ending/EndingTrigger.cs File Reference

Classes
   • class EndingTrigger

5.48 Assets/Scripts/Events/EventManager.cs File Reference

Classes
   • class EventManager
5.49  Assets/Scripts/Game.cs File Reference

Classes
  • class Game

5.50  Assets/Scripts/Graffiti/TutorialGraffiti.cs File Reference

Classes
  • class TutorialGraffiti

5.51  Assets/Scripts/GUI/CraftingGUI/IngredientButtonBehavior.cs File Reference

Classes
  • class IngredientButtonBehavior

5.52  Assets/Scripts/GUI/CraftingGUI/RecipeBookBehavior.cs File Reference

Classes
  • class RecipeBookBehavior

5.53  Assets/Scripts/GUI/CraftingGUI/RecipeButtonGUIBehavior.cs File Reference

Classes
  • class RecipeButtonGUIBehavior

5.54  Assets/Scripts/GUI/CraftingGUI/RecipePageBehavior.cs File Reference

Classes
  • class RecipePageBehavior

5.55  Assets/Scripts/GUI/CraftingGUI/RecipeRequirementsUI.cs File Reference

Classes
  • class RecipeRequirementsUI
5.56 Assets/Scripts/GUI/CraftingGUI/SelectedIngredientButton.cs File Reference

Classes

- class `SelectedIngredientButton`

5.57 Assets/Scripts/GUI/GameViewScene/GameViewBehavior.cs File Reference

Classes

- class `GameViewBehavior`

5.58 Assets/Scripts/GUI/GameViewScene/HUDBehavior.cs File Reference

Classes

- class `HUDBehavior`

5.59 Assets/Scripts/GUI/GameViewScene/SicknessNotification.cs File Reference

Classes

- class `SicknessNotification`

5.60 Assets/Scripts/GUI/InstanceManagerGUI.cs File Reference

Classes

- class `GuiInstanceManager`

5.61 Assets/Scripts/GUI/InventoryGUI/ChooseItemAmountPanelBehavior.cs File Reference

Classes

- class `ChooseItemAmountPanelBehavior`

  Item amount affect panel behavior.
5.62 Assets/Scripts/GUI/InventoryGUI/InventoryUI.cs File Reference

Classes

• class InventoryUI

5.63 Assets/Scripts/GUI/InventoryGUI/ItemActionButtonUI.cs File Reference

Classes

• class ItemActionButtonUI

5.64 Assets/Scripts/GUI/InventoryGUI/ItemAttributeUI.cs File Reference

Classes

• class ItemAttributeUI

5.65 Assets/Scripts/GUI/InventoryGUI/ItemStackDetailPanelBehavior.cs File Reference

Classes

• class ItemStackDetailPanelBehavior

Item stack detail panel behavior.

5.66 Assets/Scripts/GUI/InventoryGUI/ItemStackUI.cs File Reference

Classes

• class ItemStack

5.67 Assets/Scripts/GUI/MainMenuScene/EndCreditsBehavior.cs File Reference

Classes

• class EndCreditsBehavior

5.68 Assets/Scripts/GUI/MainMenuScene/KeyCodeConfigMenuBehavior.cs File Reference

Classes

• class KeyCodeConfigMenuBehavior
Classes

• class MainMenuBehavior

5.70 Assets/Scripts/GUI/MainMenuScene/SettingsMenuBehavior.cs File Reference

Classes

• class SettingsMenuBehavior

5.71 Assets/Scripts/GUI/MainMenuScene/SplashScreenBehavior.cs File Reference

Classes

• class SplashScreenBehavior

5.72 Assets/Scripts/GUI/Utility/FaderManager.cs File Reference

Classes

• class FaderManager
  Can be called to fade the screen in or out.

5.73 Assets/Scripts/GUI/Utility/GridLayoutManager.cs File Reference

Classes

• class GridLayoutManager

5.74 Assets/Scripts/GUI/Utility/SpriteManager.cs File Reference

Classes

• class SpriteManager

5.75 Assets/Scripts/GUI/WorldInteraction/EquippedItemDropdown.cs File Reference

Classes

• class EquippedItemDropdown
Classes

- class `WorldSelectionButtonBehavior`

Classes

- class `WorldSelectionGUIDirector`

Classes

- class `Attribute`

Classes

- class `BaseItem`
  
  *Class attached to all items. Handles getting information from the various Item Categories.*

Classes

- class `ClothCategory`

Classes

- class `CollectableItem`
  
  *Abstract class that is inherited by all BaseItem classes and Item Categories*
5.82  Assets/Scripts/Item Crafting/Backend/ContainerCategory.cs File Reference

Classes

• class ContainerCategory

5.83  Assets/Scripts/Item Crafting/Backend/CraftingStat.cs File Reference

Classes

• class CraftingStat

5.84  Assets/Scripts/Item Crafting/Backend/EquipableCategory.cs File Reference

Classes

• class EquipableCategory

5.85  Assets/Scripts/Item Crafting/Backend/FireBaseCategory.cs File Reference

Classes

• class FireBaseCategory

5.86  Assets/Scripts/Item Crafting/Backend/FishingRodCategory.cs File Reference

Classes

• class FishingRodCategory

5.87  Assets/Scripts/Item Crafting/Backend/FleshCategory.cs File Reference

Classes

• class FleshCategory

5.88  Assets/Scripts/Item Crafting/Backend/FuelCategory.cs File Reference

Classes

• class FuelCategory
5.89  Assets/Scripts/Item Crafting/Backend/IdolCategory.cs File Reference

Classes

• class IdolCategory

5.90  Assets/Scripts/Item Crafting/Backend/Ingredient.cs File Reference

Classes

• struct Ingredient

5.91  Assets/Scripts/Item Crafting/Backend/Inventory.cs File Reference

Classes

• class Inventory

Defines the inventory of items the player can access. Currently just considers all the gameobjects parented under
item parent as part of the inventory. TODO: Enable this to be used for on hand and raft inventories. Add inventory
size limit and ability to adjust the size. Make the inventory save strings of objects then use YAML file to define those
items as needed.

5.92  Assets/Scripts/Item Crafting/Backend/ItemAction.cs File Reference

Classes

• class ItemAction

Struct that defines an action being the name of the action and the UnityAction. The action may also have an id that
specifies which subcategory of action it falls under.

5.93  Assets/Scripts/Item Crafting/Backend/ItemCategory.cs File Reference

Classes

• class ItemCategory

Abstract class for the classes which contain suites of actions and attributes which can be used to define a category
of items. Examples include solid, liquid, and plant.

5.94  Assets/Scripts/Item Crafting/Backend/ItemCondition.cs File Reference

Classes

• class ItemCondition
5.95 Assets/Scripts/Item Crafting/Backend/ItemFactory.cs File Reference

Classes

- class ItemFactory

  Factory that creates an base items given its name and complex crafted items given recipe and ingredients. Currently just a placeholder class only able to craft a fishing rod.

5.96 Assets/Scripts/Item Crafting/Backend/ItemRarity.cs File Reference

Classes

- class ItemRarity

5.97 Assets/Scripts/Item Crafting/Backend/ItemStack.cs File Reference

Classes

- class ItemStack

5.98 Assets/Scripts/Item Crafting/Backend/ItemTypes.cs File Reference

Classes

- class ItemTypes

5.99 Assets/Scripts/Item Crafting/Backend/LightCategory.cs File Reference

Classes

- class LightCategory

5.100 Assets/Scripts/Item Crafting/Backend/MedicineCategory.cs File Reference

Classes

- class MedicineCategory
5.101 Assets/Scripts/Item Crafting/Backend/PlantCategory.cs File Reference

Classes

- class PlantCategory

  Contains attributes and actions that befit a plant category item.

5.102 Assets/Scripts/Item Crafting/Backend/PlayerInventory.cs File Reference

Classes

- class PlayerInventory

5.103 Assets/Scripts/Item Crafting/Backend/RaftCategory.cs File Reference

Classes

- class RaftCategory

5.104 Assets/Scripts/Item Crafting/Backend/Recipe.cs File Reference

Classes

- class Recipe

  Class that handles storing the contents of recipes and checking whether or not the requirements are met.

5.105 Assets/Scripts/Item Crafting/Backend/Requirement.cs File Reference

Classes

- class Requirement

  A requirement includes a type of item (an item with a certain tag) and the amount of it.

5.106 Assets/Scripts/Item Crafting/Backend/ShelterCategory.cs File Reference

Classes

- class ShelterCategory
5.107 Assets/Scripts/Item Crafting/Backend/SolidCategory.cs File Reference

Classes

- class SolidCategory
  
  The solid item category. Contains attributes and actions that befit a solid category item.

5.108 Assets/Scripts/Item Crafting/Backend/WarmthIdolCategory.cs File Reference

Classes

- class WarmthIdolCategory

5.109 Assets/Scripts/Item Crafting/BaseItemPopupAttribute.cs File Reference

Classes

- class BaseItemPopupAttribute

5.110 Assets/Scripts/ItemProceduralGeneration/DistrictItemConfiguration.cs File Reference

Classes

- class DistrictItemConfiguration

5.111 Assets/Scripts/ItemProceduralGeneration/DistrictItemRarityConfiguration.cs File Reference

Classes

- class DistrictItemRarityConfiguration

5.112 Assets/Scripts/ItemProceduralGeneration/DummyGenerator.cs File Reference

Classes

- class DummyGenerator
5.113  Assets/Scripts/ItemProceduralGeneration/ItemDiscarder.cs File Reference

Classes
  • class ItemDiscarder

5.114  Assets/Scripts/ItemProceduralGeneration/ItemGenerator.cs File Reference

Classes
  • class ItemGenerator

5.115  Assets/Scripts/ItemProceduralGeneration/ItemPlacementSamplePoint.cs File Reference

Classes
  • class ItemPlacementSamplePoint

5.116  Assets/Scripts/ItemProceduralGeneration/ItemPoolInfo.cs File Reference

Classes
  • class ItemPoolInfo

5.117  Assets/Scripts/ItemProceduralGeneration/ItemPoolManager.cs File Reference

Classes
  • class ItemPoolManager

5.118  Assets/Scripts/ItemProceduralGeneration/ItemSpawner.cs File Reference

Classes
  • class ItemSpawner

5.119  Assets/Scripts/ItemProceduralGeneration/RoofTopGeneration.cs File Reference

Classes
  • class RooftopGeneration
5.120 Assets/Scripts/ItemProceduralGeneration/RooftopPointGenerator.cs File Reference

Classes

• class RooftopPointGenerator

5.121 Assets/Scripts/ItemProceduralGeneration/SamplingPointGenerator.cs File Reference

Classes

• class SamplingPointGenerator

5.122 Assets/Scripts/ItemProceduralGeneration/WaterItemGeneration.cs File Reference

Classes

• class WaterItemGeneration

5.123 Assets/Scripts/ItemProceduralGeneration/WaterPointGenerator.cs File Reference

Classes

• class WaterPointGenerator

5.124 Assets/Scripts/ItemProceduralGeneration/WeightedPair.cs File Reference

Classes

• class WeightedPair

5.125 Assets/Scripts/ItemProceduralGeneration/WorldItemFactory.cs File Reference

Classes

• class WorldItemFactory

Generated by Doxygen
5.126 Assets/Scripts/Items/Barometer.cs File Reference

Classes

- class Barometer

5.127 Assets/Scripts/Items/FireInteractable.cs File Reference

Classes

- class FireInteractable

5.128 Assets/Scripts/Items/FishingLure.cs File Reference

Classes

- class FishingLure

5.129 Assets/Scripts/Items/FishingRod.cs File Reference

Classes

- class FishingRod

5.130 Assets/Scripts/Items/FloatBehavior.cs File Reference

Classes

- class FloatBehavior

5.131 Assets/Scripts/Items/HasSelectionInterface.cs File Reference

Classes

- interface HasSelectionInterface

5.132 Assets/Scripts/Items/Idol.cs File Reference

Classes

- class Idol
5.133 Assets/Scripts/Items/InteractableObject.cs File Reference

Classes
- class InteractableObject

5.134 Assets/Scripts/Items/InteractableSetUp.cs File Reference

Classes
- class InteractableSetUp

5.135 Assets/Scripts/Items/InventoryInteractable.cs File Reference

Classes
- class InventoryInteractable

5.136 Assets/Scripts/Items/LightItem.cs File Reference

Classes
- class LightItem

5.137 Assets/Scripts/Items/OverworldItemOptionSelection.cs File Reference

Classes
- class OverworldItemOptionSelection

5.138 Assets/Scripts/Items/PickUpItem.cs File Reference

Classes
- class PickUpItem

5.139 Assets/Scripts/Items/RaftInteractable.cs File Reference

Classes
- class RaftInteractable
5.140 Assets/Scripts/Items/ShelterInteractable.cs File Reference

Classes

• class ShelterInteractable

5.141 Assets/Scripts/Items/Thermometer.cs File Reference

Classes

• class Thermometer

5.142 Assets/Scripts/Items/Tool.cs File Reference

Classes

• class Tool

5.143 Assets/Scripts/Loading/GameLoader.cs File Reference

Classes

• class GameLoader
  
  Tracks all loading tasks and determines overall loading percentage.

5.144 Assets/Scripts/Loading/GameLoaderTask.cs File Reference

Classes

• class GameLoaderTask
  
  Tracks information for a loading task.

5.145 Assets/Scripts/Loading/LoadingScreen.cs File Reference

Classes

• class LoadingScreen
5.146 Assets/Scripts/Loading/LoadingScreenFlavorText.cs File Reference

Classes

• class LoadingScreenFlavorText

5.147 Assets/Scripts/Loading/LoadingScreenFlavorTextManager.cs File Reference

Classes

• class LoadingScreenFlavorTextManager

5.148 Assets/Scripts/Movement/Creatures/FishAgent.cs File Reference

Classes

• class FishAgent

5.149 Assets/Scripts/Movement/Creatures/FishAgentConfig.cs File Reference

Classes

• class FishAgentConfig

5.150 Assets/Scripts/Movement/Creatures/WindMovement.cs File Reference

Classes

• class WindMovement

5.151 Assets/Scripts/Movement/LandMovement.cs File Reference

Classes

• class LandMovement

5.152 Assets/Scripts/Movement/Movement.cs File Reference

Classes

• class Movement
5.153  Assets/Scripts/Movement/PlayerAnimationEvents.cs File Reference

**Classes**

- class `PlayerAnimationEvents`

  *This script has functions called by the player’s animator.*

5.154  Assets/Scripts/Movement/PlayerFocus.cs File Reference

**Classes**

- class `PlayerFocus`

  *Controls the player’s head IK setup to have them look at the closest interactable if they can.*

5.155  Assets/Scripts/Movement/RaftMovement.cs File Reference

**Classes**

- class `RaftMovement`

5.156  Assets/Scripts/Movement/RaftWake.cs File Reference

**Classes**

- class `RaftWake`

5.157  Assets/Scripts/Movement/WaterMovement.cs File Reference

**Classes**

- class `WaterMovement`

5.158  Assets/Scripts/Notes/DummyNoteGenerator.cs File Reference

**Classes**

- class `DummyNoteGenerator`
5.159 Assets/Scripts/Notes/Note.cs File Reference

Classes

• class Note

   Base interactable note class for all notes.

5.160 Assets/Scripts/Notes/NoteData.cs File Reference

Classes

• class NoteData

   Base class for all notes.

5.161 Assets/Scripts/Notes/NoteFactory.cs File Reference

Classes

• class NoteFactory

5.162 Assets/Scripts/Notes/NoteUIController.cs File Reference

Classes

• class NoteUIController

   Note user interface controller. TODO: Make this a child of game UI

5.163 Assets/Scripts/Player/ControlScheme.cs File Reference

Classes

• class ControlScheme

5.164 Assets/Scripts/Player/HealthRateManager.cs File Reference

Classes

• class HealthRateManager

Generated by Doxygen
5.165 Assets/Scripts/Player/HungerRateManager.cs File Reference

Classes

- class HungerRateManager

5.166 Assets/Scripts/Player/Player.cs File Reference

Classes

- class Player

Enumerations

- enum PlayerHealthStatus { PlayerHealthStatus.None, PlayerHealthStatus.FoodPoisoning, PlayerHealthStatus.Pneumonia }

Types of possible player health statuses

5.166.1 Enumeration Type Documentation

5.166.1.1 PlayerHealthStatus

enum PlayerHealthStatus [strong]

Types of possible player health statuses

<table>
<thead>
<tr>
<th>Enumerator</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
</tr>
<tr>
<td>FoodPoisoning</td>
</tr>
<tr>
<td>Pneumonia</td>
</tr>
</tbody>
</table>

Definition at line 7 of file Player.cs.

5.167 Assets/Scripts/Player/PlayerController.cs File Reference

Classes

- class PlayerController
5.168 Assets/Scripts/Player/PlayerStatManager.cs File Reference

Classes
  • class PlayerStatManager

5.169 Assets/Scripts/Player/PlayerTools.cs File Reference

Classes
  • class PlayerTools
    Used for keeping track of implementation of tools in the game scene.

5.170 Assets/Scripts/Player/StatRate.cs File Reference

Classes
  • class StatRate

5.171 Assets/Scripts/Player/WarmthRateManager.cs File Reference

Classes
  • class WarmthRateManager

5.172 Assets/Scripts/Radio/Announcements/AnnouncementCategory.cs File Reference

Enumerations

  • enum AnnouncementCategory
    { AnnouncementCategory.Location, AnnouncementCategory.Player, AnnouncementCategory.Weather, AnnouncementCategory.Time }
    Announcement category. Used in MysteryAnnouncement class.

5.172.1 Enumeration Type Documentation

5.172.1.1 AnnouncementCategory

enum AnnouncementCategory
  { AnnouncementCategory.Location, AnnouncementCategory.Player, AnnouncementCategory.Weather, AnnouncementCategory.Time }

Announcement category. Used in MysteryAnnouncement class.
## Enumerator

<table>
<thead>
<tr>
<th>Location</th>
<th>Player</th>
<th>Weather</th>
<th>Time</th>
</tr>
</thead>
</table>

Definition at line 4 of file AnnouncementCategory.cs.

### 5.173 Assets/Scripts/Radio/Announcements/AnnouncementFactory.cs File Reference

**Classes**

- class AnnouncementFactory

### 5.174 Assets/Scripts/Radio/Announcements/MysteryAnnouncement.cs File Reference

**Classes**

- class MysteryAnnouncement
  
  Mystery announcement base class

### 5.175 Assets/Scripts/Radio/Dial.cs File Reference

**Classes**

- class Dial

### 5.176 Assets/Scripts/Radio/InteractableRadioModel.cs File Reference

**Classes**

- class InteractableRadioModel

### 5.177 Assets/Scripts/Radio/Radio.cs File Reference

**Classes**

- class Radio
Enumerations

- enum RadiChannel { RadiChannel.Music, RadiChannel.Weather, RadiChannel.Mystery, RadiChannel.Null }

5.177.1 Enumeration Type Documentation

5.177.1.1 RadiChannel

enum RadiChannel [strong]

Enumerator

| Music | Weather | Mystery | Null |

Definition at line 8 of file Radio.cs.

5.178 Assets/Scripts/Radio/RadioButton.cs File Reference

Classes

- class RadioButton

Enumerations

- enum ButtonType { ButtonType.VolumeUp, ButtonType.VolumeDown, ButtonType.Power }

5.178.1 Enumeration Type Documentation

5.178.1.1 ButtonType

enum ButtonType [strong]

Enumerator

| VolumeUp | VolumeDown | Power |

Generated by Doxygen
Definition at line 5 of file RadioButton.cs.

5.179 Assets/Scripts/Radio/TestGUI.cs File Reference

Classes

• class TestGUI

5.180 Assets/Scripts/Teleporters/TeleportPlayer.cs File Reference

Classes

• class TeleportPlayer
  A class that moves the player when they press a key. Used by fire escapes, certain doors, and the window washers.
  • struct TeleportPlayer.TeleportLocation

5.181 Assets/Scripts/Teleporters/WindowWasher.cs File Reference

Classes

• class WindowWasher
  Goes on the window washer prefab, handles the ropes on the washer, moving it up with the water, and teleporting the player.

5.182 Assets/Scripts/Utility/BezierLine.cs File Reference

Classes

• class BezierLine

5.183 Assets/Scripts/Utility/BooleanOperator.cs File Reference

Classes

• class BooleanOperator

5.184 Assets/Scripts/Utility/ConsoleCommandRouter.cs File Reference

Classes

• class ConsoleCommandRouter
5.185  Assets/Scripts/Utility/File/FileManager.cs File Reference

Classes

• class FileManager

5.186  Assets/Scripts/Utility/File/GoogleDrive.cs File Reference

Classes

• class GoogleDrive

5.187  Assets/Scripts/Utility/GameSettings.cs File Reference

Classes

• class GameSettings

5.188  Assets/Scripts/Utility/NullParent.cs File Reference

Classes

• class NullParent

  Sometimes you want something to be in a prefab, but not have a parent in the hierarchy. This script helps with that.

5.189  Assets/Scripts/Utility/PauseSystem.cs File Reference

Classes

• class PauseSystem

5.190  Assets/Scripts/Utility/RandomUtility.cs File Reference

Classes

• class RandomUtility

5.191  Assets/Scripts/Utility/Regression.cs File Reference

Classes

• class Regression
5.192  Assets/Scripts/Utility/SystemLogger.cs File Reference

Classes

• class **SystemLogger**

5.193  Assets/Scripts/Utility/TransformParent.cs File Reference

Classes

• class **TransformParent**

  *Can Parent one transform to another while ignoring specific axis.*

5.194  Assets/Scripts/Utility/Tuple.cs File Reference

Classes

• class **Tuple<T1, T2>**

  *Literally here so I can have a way of having int pairs without converting back and forth from Vector2.*

• class **Tuple<T1, T2>**

  *Literally here so I can have a way of having int pairs without converting back and forth from Vector2.*

5.195  Assets/Scripts/Utility/VectorUtility.cs File Reference

Classes

• class **VectorUtility**

5.196  Assets/Scripts/Weather/Clock.cs File Reference

Classes

• class **Clock**

5.197  Assets/Scripts/Weather/DayNight.cs File Reference

Classes

• class **DayNight**
5.198 Assets/Scripts/Weather/DiurnalTemperatureVariance.cs File Reference

Classes

- class DiurnalTemperatureVariance
  A gamified version of: https://en.wikipedia.org/wiki/Diurnal_temperature_variation

5.199 Assets/Scripts/Weather/Effects/LightningEffect.cs File Reference

Classes

- class LightningEffect

5.200 Assets/Scripts/Weather/FloodWater.cs File Reference

Classes

- class FloodWater

5.201 Assets/Scripts/Weather/GUI/PressureSystemVisualization.cs File Reference

Classes

- class PressureSystemVisualization

5.202 Assets/Scripts/Weather/GUI/SetEnvironmentGUI.cs File Reference

Classes

- class SetEnvironmentGUI

5.203 Assets/Scripts/Weather/Lightning.cs File Reference

Classes

- class Lightning

5.204 Assets/Scripts/Weather/PressureSystem.cs File Reference

Classes

- class PressureSystem

Generated by Doxygen
5.205  Assets/Scripts/Weather/PressureSystems.cs File Reference

Classes

• class PressureSystems

Enumerations

• enum PressureConstants {
  PressureConstants.PressureSiberia, PressureConstants.PressureHighMax, PressureConstants.PressureHighMin, PressureConstants.PressureLowMax,
  PressureConstants.PressureLowMin, PressureConstants.PressureHurricane
}  

  Pressure constants that access readonly array pressureConstants in PressureSystem.

5.205.1  Enumeration Type Documentation

5.205.1.1  PressureConstants

definition at line 9 of file PressureSystems.cs.

5.206  Assets/Scripts/Weather/WeatherSoundSystem.cs File Reference

Classes

• class WeatherSoundSystem

5.207  Assets/Scripts/Weather/WeatherSystem.cs File Reference

Classes

• class WeatherSystem
Enumerations

- enum Weather {
  WeatherPressure, WeatherTemperature, WeatherWindSpeedMagnitude, WeatherWindSpeedX,
}

Enumeration for accessing weather variables in the array that contains the weather data in class WeatherSystem.

5.207.1 Enumeration Type Documentation

5.207.1.1 Weather

```csharp
enum Weather [strong]

Enumeration for accessing weather variables in the array that contains the weather data in class WeatherSystem.

<table>
<thead>
<tr>
<th>Enumerator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pressure</td>
</tr>
<tr>
<td>Temperature</td>
</tr>
<tr>
<td>WindSpeedMagnitude</td>
</tr>
<tr>
<td>WindSpeedX</td>
</tr>
<tr>
<td>WindSpeedY</td>
</tr>
<tr>
<td>RelativeHumidity</td>
</tr>
<tr>
<td>RelativeDewPoint</td>
</tr>
<tr>
<td>Precipitation</td>
</tr>
</tbody>
</table>
```

Definition at line 9 of file WeatherSystem.cs.

5.208 Assets/Scripts/YamlParsing/AnnouncementYAMLParser.cs File Reference

Classes

- class AnnouncementYAMLParser

5.209 Assets/Scripts/YamlParsing/CraftingSystemSerializer.cs File Reference

Classes

- class CraftingSystemSerializer
5.210  Assets/Scripts/YamlParsing/InventoryItemYAMLModel.cs File Reference

Classes
  • class InventoryItemYAMLModel

5.211  Assets/Scripts/YamlParsing/InventoryYAMLModel.cs File Reference

Classes
  • class InventoryYAMLModel

5.212  Assets/Scripts/YamlParsing/InventoryYamlParser.cs File Reference

Classes
  • class InventoryYamlParser

5.213  Assets/Scripts/YamlParsing/ItemDistrictModel.cs File Reference

Classes
  • class ItemDistrictModel

5.214  Assets/Scripts/YamlParsing/ItemSerializer.cs File Reference

Classes
  • class ItemSerializer

5.215  Assets/Scripts/YamlParsing/ItemYAMLMap.cs File Reference

Classes
  • class ItemYAMLMap

5.216  Assets/Scripts/YamlParsing/LoadingScreenFlavorTextYAMLParser.cs File Reference

Classes
  • class LoadingScreenFlavorTextYAMLParser
5.217  Assets/Scripts/YamlParsing/NoteYAMLParser.cs File Reference

Classes

• class NoteYAMLParser

5.218  Assets/Scripts/YamlParsing/RecipeYamlSerializer.cs File Reference

Classes

• class RecipeYamlSerializer
Index

AccumulatedFallDamage
  Movement, 418
Action
  ControlScheme, 110
  ItemStackUI, 373
ActionComplete
  ItemAction, 308
ActionModifiedModels
  BaseItem, 41
ActionModifiedSprites
  BaseItem, 41
ActionName
  ItemAction, 308
Actions
  ItemCategory, 321
ActivatePressureSystem
  ConsoleCommandRouter, 103
ActivateRadio
  InteractableRadioModel, 281
Activated
  ItemPoolInfo, 343
AddBuildingBounds
  WaterPointGenerator, 641
AddBuildingToWaterGenerationMap
  WaterItemGeneration, 631
AddDoorsToDistrict
  RooftopGeneration, 558
AddFuel
  FireBaseCategory, 184
  LightCategory, 393
  LightItem, 398
AddItem
  Inventory, 287
  PlayerInventory, 470
AddItemCategory
  BaseItem, 34
AddItemFromWorld
  ItemPoolManager, 346
AddItemInfo
  ItemPoolInfo, 342
AddItemToPool
  ItemPoolManager, 346
AddNewInventory
  InventoryYamlParser, 305
AddNonDuplicateVertex
  GenerationUtility, 246
AddPossibleAction
  OverworldItemOptionSelection, 436
AddRequirement
  Recipe, 531
  AddSelectedIngredients
    RecipePageBehavior, 544
  AddSheltersToDistrict
    RooftopGeneration, 558
  AddTemplatesToItemPool
    ItemGenerator, 335
  AddToCarousel
    Radio, 507
  AddToGrid
    ItemPoolManager, 346
  AddWindows
    ProceduralBuildingAttachment, 489
    ProceduralBuildingBase, 492
AdvancedTool
  ItemTypes, 375
agentLayer
  FishAgent, 195
Alignment
  FishAgent, 193
AlignmentRadius
  FishAgentConfig, 199
AlignmentWeight
  FishAgentConfig, 199
AmbientSoundManager, 19
Amount
  Ingredient, 272
  ItemStack, 359
  PickUpItem, 443
  SelectedIngredientButton, 569
AmountRequired
  Requirement, 554
AnglePerSecond
  Clock, 92
AnnouncementCategory
  AnnouncementCategory.cs, 697
  AnnouncementCategory.cs
  AnnouncementFactory, 697
AnnouncementDatabase
  AnnouncementFactory, 21
  AnnouncementDatabase, 21
  AnnouncementFactory, 21
  AnnouncementFactory, 21
  AnnouncementDatabase, 22
  PlayerDatabase, 22
  TimeDatabase, 22
  WeatherDatabase, 22
AnnouncementYAMLParser
  AnnouncementYAMLParser, 23
  AnnouncementYAMLParser, 23
<table>
<thead>
<tr>
<th>Project Path</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assets/Scripts/Item Crafting/Backend/Collectable-Item.cs,</td>
<td>Assets/Scripts/ItemProceduralGeneration/ItemDiscarder.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/ItemProceduralGeneration/ItemGenerator.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/ItemProceduralGeneration/ItemPlacementSamplePoint.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/ItemProceduralGeneration/ItemPoolInfo.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/ItemProceduralGeneration/ItemPoolManager.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/ItemProceduralGeneration/ItemSpawner.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/ItemProceduralGeneration/RooftopPointGenerator.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/ItemProceduralGeneration/WorldItemFactory.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/Barometer.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/FireInteractable.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/FishingLure.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/FishingRod.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/FloatBehavior.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/Idol.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/InteractableObject.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/InteractableSetUp.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/InventoryInteractable.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/LightItem.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/OverworldItemOptionSelection.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/PickUpItem.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/RaftInteractable.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/ShelterInteractable.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/Thermometer.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Items/Tool.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Loading/GameLoader.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Loading/GameLoaderTask.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Loading/LoadingScreen.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Loading/LoadingScreenFlavorText.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Loading/LoadingScreenFlavorTextManager.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Movement/Creatures/FishAgent.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Movement/Creatures/FishAgentConfig.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Movement/Creatures/WindMovement.cs,</td>
</tr>
<tr>
<td></td>
<td>Assets/Scripts/Movement/LandMovement.cs,</td>
</tr>
<tr>
<td></td>
<td>Generated by Doxygen</td>
</tr>
</tbody>
</table>
DistrictConfiguration::BuildingTemplatePlacement, 67
Instance, 60
IsLoaded, 60
Load, 59
Parent, 61
Position, 61
Unload, 59
BuildingAttachments
  ProceduralBuildingInstance, 500
BuildingBase
  ProceduralBuildingInstance, 500
BuildingGenerator, 64
  CityCenterBuilding, 65
  Generate, 65
buildingIndex, 66
  BuildingSize, 66
  PartNumber, 66
BuildingMaterial
  ProceduralBuildingInstance, 500
BuildingParts
  ProceduralBuildingCreator.cs, 671
BuildingRoof
  ProceduralBuildingInstance, 501
BuildingScale
  ProceduralBuilding, 486
BuildingSize
  buildingIndex, 66
Buildings
  Block, 52
  CityChunk, 83
buildings
  DummyGenerator, 158
BurnItem
  FireInteractable, 189
BurnRate
  LightCategory, 394
  BurnRateMultiplier
    FirebaseCategory, 185
BurnTime
  FuelCategory, 223
ButtonName
  WorldSelectionButtonBehavior, 664
ButtonType
  RadioButton.cs, 699
buttons
  TestGUI, 603
calculateBounds
  ItemGenerator, 335
CalculateRemainingFuel
  FirebaseCategory, 184
  LightCategory, 393
CallJumpForce
  PlayerAnimationEvents, 456
CallJumpLand
  PlayerAnimationEvents, 456
CameraController, 68
  CameraPositions, 70
CanCast
  FishingRod, 208
CanRotate
  ProceduralBuildingRoof, 503
CanTeleportTo
  TeleportPlayer::TeleportLocation, 594
CancelChosenAmount
  ChooseItemAmountPanelBehavior, 73
Cast
  FishingLure, 203
Category
  MysteryAnnouncement, 420
categoryNames
  CraftingSystemSerializer, 117
categoryTypes
  CraftingSystemSerializer, 117
cellSize
  SamplingPointGenerator, 565
Center
  Block, 52
  City, 77
  Voroni, 621
CertificateValidationCallback
  GoogleDrive, 250
Change
  ChooseItemAmountPanelBehavior, 73
ChangeChannel
  Radio, 507
ChangeName
  BaseItem, 34
ChangeRateValues
  StatRate, 592
changeVolume
  SettingsMenuBehavior, 573
CheckBarometer
  Barometer, 29
CheckClosestInteractable
  PlayerController, 460
CheckCompleted
  Recipe, 532
CheckCondition
    ItemCondition, 323
CheckForModification
    ItemStackUI, 369
CheckGridSize
    GridLayoutManager, 253
CheckRecipePossible
    Inventory, 288
CheckRequirementMet
    Inventory, 288
CheckThermometer
    Thermometer, 607
ChooseItemAmountPanelBehavior
    CancelChosenAmount, 71
    Change, 73
    ChooseNumOfItemsToAffect, 74
    CloseEntirePanel, 74
    FinalizeAction, 74
    ItemNameDisplay, 75
    OpenItemAmountPanel, 74
    OpenItemDetailPanel, 74
    SelectedItem, 75
ChooseNumOfItemsToAffect
    ChooseItemAmountPanelBehavior, 74
ChunkSize
    CityChunkManager, 87
Chunks
    CityChunkManager, 86
ChunksAcross
    CityChunkManager, 86
ChunksAwayToLoad
    CityChunkManager, 86
ChunksDown
    CityChunkManager, 86
City
    BoundingBox, 77
    Center, 77
    City, 76
    Districts, 77
    GetEnumerator, 77
    TallestBuilding, 77
City.BuildingEnumerator, 61
City::BuildingEnumerator
    Current, 63
    Dispose, 62
    MoveNext, 63
    Reset, 63
CityBoundaries
    BoundingBox2d, 79
    BoundingBox3d, 79
    CityBoundaries, 79
    CityBounds, 80
    RandomVector2d, 80
    RandomVector3d, 80
CityBounds
    CityBoundaries, 80
    Game, 231
    PressureSystems, 481
CityCenterBuilding
    BuildingGenerator, 65
CityChunk
    BoundingBox, 83
    Buildings, 83
    CityChunk, 82
    CurrentCoroutine, 83
    IsLoaded, 83
    Load, 82
    Location, 84
    Unload, 83
CityChunkManager
    ChunkSize, 87
    Chunks, 86
    ChunksAcross, 86
    ChunksAwayToLoad, 86
    ChunksDown, 86
    CurrentChunk, 87
    FreezeLoading, 87
    Init, 86
CityController, 88
CityInstance
    Game, 231
ClearAttributesAndActions
    ItemStackDetailPanelBehavior, 363
ClearOptions
    WorldSelectionGUIDirector, 666
ClearSubActionPanel
    ItemStackDetailPanelBehavior, 363
Climb
    LandMovement, 387
    Movement, 417
    RaftMovement, 520
    WaterMovement, 634
climbHeight
    Movement, 418
ClimbingRaycastMask
    PlayerController, 462
Clock
    AnglePerSecond, 92
    CurrentGameTimeInHours, 92
    CurrentTime, 92
    FormattedTime, 92
    GetTwelveHourPercentage, 91
    HalfHour, 93
    HalfHourDelegateUpdate, 91
    HalfHourUpdate, 94
    Hour, 93
    HourDelegateUpdate, 91
    HourUpdate, 94
    IsDay, 93
    SecondDelegateUpdate, 92
    SecondUpdate, 94
    Tick, 93
    TwelveHours, 94
    TwentyFourHours, 94
ClockInstance
    Game, 231
categoryNames, 117
categoryTypes, 117
Filename, 117
SetUpCategoryInformation, 116
uriPrefix, 117
CreateBuilding
  ProceduralBuildingCreator, 497
CreateGameLoaderTask
  GameLoader, 237
CreateGenericInteractableItem
  WorldItemFactory, 660
CreateInteractableItem
  NoteFactory, 428
CreatePickUpInteractableItem
  WorldItemFactory, 661
CreateRandomPickupInteractableItem
  WorldItemFactory, 661
CreateSplash
  FXSplashManager, 227, 228
CreateSubAction
  ItemStackDetailPanelBehavior, 363
CreateTestBuilding
  ProceduralBuildingCreator, 497
CreatureCount
  CreatureManager, 120
CreatureKilled
  CreatureSpawner, 123
CreatureManager, 118
CreatureCount, 120
CreatureManager, 118
  IdealCreatureCount, 120
  PutCreatureInPool, 119
  Spawn, 119
  UpdateCreatureCount, 120
  UpdateCreatureInfo, 120
creatureManager
  CreatureSpawner, 124
CreatureSpawner, 121
  Count, 124
  CreatureKilled, 123
  creatureManager, 124
  findSpawnLocation, 123
  Init, 123
  maxCreatureCount, 124
  meanCreatureCount, 124
  minCreatureCount, 124
CreatureTracker, 125
  Index, 126
  IsDead, 126
CureAll
  MedicineCategory, 412
CureFoodPoisoning
  MedicineCategory, 412
CurePneumonia
  MedicineCategory, 412
Current
  City::BuildingEnumerator, 63
  CurrentChunk
INDEX

DistrictConfiguration, 143
DistrictMaxWindowSpacings
  DistrictConfiguration, 143
DistrictMinWindowSpacings
  DistrictConfiguration, 143
DistrictName
  ItemDistrictModel, 326
DistrictPopup, 153
  OnGUI, 154
DistrictPopupAttribute, 155
DistrictPosters
  DistrictConfiguration, 143
districtProceduralMaterials
  DistrictConfiguration, 141
districtProceduralWindowMaterials
  DistrictConfiguration, 141
DistrictSizeXBases
  DistrictConfiguration::ProceduralBuildingBases
    SizeXDefinition, 494
DistrictSizeXRoofs
  DistrictConfiguration::ProceduralBuildingBases
    SizeXDefinition, 495
DistrictWindowMaterials
  DistrictConfiguration, 144
DistrictWindowWashers
  DistrictConfiguration, 144
DistrictWindows
  DistrictConfiguration, 144
Districts
  City, 77
    WaterPointGenerator, 642
DiurnalTemperatureVariance, 156
  TemperatureVariance, 156
DoorExtents
  DistrictItemConfiguration, 149
DoorTemplates
  DistrictItemConfiguration, 149
Doors
  DistrictConfiguration, 142
doors
    DummyGenerator, 158
Dry
  PlantCategory, 446
DummyGenerator, 157
  buildings, 158
    doors, 158
    shelters, 158
DummyNoteGenerator, 159
Durability
  SolidCategory, 585
Eat
  FleshCategory, 216
    PlantCategory, 447
EdgeVerticies
  District, 138
Edible
  ItemTypes, 376
Elasticity
  SolidCategory, 586
ElementsPerRow
  GridLayoutManager, 254
EnableWeather
  WeatherSystem, 647
EnabledColor
  RecipeButtonGUIBehavior, 540
EndCraftingAttempt
  RecipePageBehavior, 545
EndCreditsBehavior, 160
EndGameFade
  FaderManager, 178
EndPoint
  BezierLine, 49
EndingController, 161
  BeaconTurnedOn, 162
    disableThese, 162
    EndingTrigger, 162
    hideThese, 163
EndingTrigger, 163
    EndingController, 162
    EndingTriggered
      CameraController, 70
Equal
  BooleanOperator, 55
Equip
  Barometer, 28
    EquipableCategory, 167
    FishingRod, 207
    Idol, 266
    IdolCategory, 270
    LightItem, 398
    Thermometer, 606
    Tool, 609
  equipActionName
    EquipableCategory, 168
Equipable
  ItemTypes, 376
  EquipableCategory, 165
    Equip, 167
      equipActionName, 168
    Equiped, 169
      equipedAttributeName, 168
      GetDuplicate, 167
      ReadyCategory, 167
      UnEquip, 168
      unequipActionName, 168
  Equiped
    EquipableCategory, 169
    equipedAttributeName
      EquipableCategory, 168
EquipedImage
  PlayerInventory, 471
EquippedItemDropdown
  contentPanel, 172
  DisplayOptions, 171
    equippedItemHolder, 172
    GetSelected, 171

Generated by Doxygen
HideOptions, 171  
SetEquipped, 171  
ToggleDisplay, 172  
Unequipped, 172

equippedItemHolder  
EquippedItemDropdown, 172  
EquippedTool  
PlayerTools, 477

EventManager, 173  
Game, 232  
PlayerBoardRaftDelegate, 174  
PlayerBoardRaftSubscription, 176  
RadioMusicDelegate, 174  
RadioMusicSubscription, 176  
RadioMusicTurnedOff, 174  
RadioMusicTurnedOn, 174  
RaftBoarded, 174  
StormStart, 175  
StormStartedDelegate, 175  
StormStartedSubscription, 176  
StormStop, 175  
StormStoppedDelegate, 175  
StormStoppedSubscription, 176  
WeatherUpdated, 175  
WeatherUpdatedDelegate, 175  
WeatherUpdatedSubscription, 176

EventPath  
MysteryAnnouncement, 420

FXSplash, 224  
setAnimationSpeed, 225  
SplashAnimator, 226  
FXSplashManager, 226  
CreateSplash, 227, 228  
FillPool, 229  
SplashPoolSize, 229

FabricThickness  
ClothCategory, 98

FaderManager, 177  
EndGameFade, 178  
GoToMain, 179  
ShowMainMenuButton, 179

FileManager, 179  
BuildPath, 180  
GetDocument, 180  
SaveAllDocuments, 181  
SaveFile, 181

Filename  
CraftingSystemSerializer, 117

FillPool  
FXSplashManager, 229

FinalizeAction  
ChooseItemAmountPanelBehavior, 74

findSpawnLocation  
CreatureSpawner, 123  
WhaleSpawner, 653

finishDuplication  
ItemCategory, 318

FinishSelection  
WorldSelectionGUIDirector, 667

FireBase  
FireInteractable, 190  
FireBaseCategory, 182  
AddFuel, 184  
BurnRateMultiplier, 185  
CalculateRemainingFuel, 184  
FuelRemaining, 186  
GetDuplicate, 185  
ReadyCategory, 185  
SetDown, 185  
StartingFuel, 186

FireInteractable, 187  
BurnItem, 189  
FireBase, 190  
IgniteFire, 189  
OnTriggerEnter, 189  
SetUp, 189  
UpdateFire, 190  
UseFirePit, 190

FireItemNameChangedEvent  
Baseltem, 35  
FireItemSpriteChangedEvent  
Baseltem, 35

First  
WeightedPair, 651

FishAgent, 191  
agentLayer, 195  
Alignment, 193  
Cohesion, 193  
Combine, 194  
config, 195  
InFieldOfView, 194  
predatorLayer, 196  
Separation, 194  
setLayers, 195  
Velocity, 196  
Wander, 195

FishAgentConfig, 197  
AlignmentRadius, 199  
AlignmentWeight, 199  
AvoidRadius, 199  
AvoidWeight, 199  
CohesionRadius, 200  
CohesionWeight, 200  
Jitter, 200  
MaxAcceleration, 200  
MaxFieldOfViewAngle, 200  
MaxRadius, 200  
MaxVelocity, 201  
MaxWeight, 201

RandomizeSelf, 199  
SeparationRadius, 201  
SeparationWeight, 201  
WanderDistanceRadius, 201  
WanderRadius, 201  
WanderWeight, 202  
waterLevelOffset, 202
FishSpawner, 212
Init, 213
FishingLure, 202
Cast, 203
IsReeling, 204
Position, 204
Reel, 204
ReelingSpeed, 204
FishingRod, 205
CanCast, 208
Equip, 207
SetUpTool, 207
Unequip, 207
Use, 207
WasCast, 208
FishingRodCategory, 209
GetDuplicate, 211
ReadyCategory, 211
FlavorText
BaseItem, 41
flavorText
DeathFlavorText, 130
FleshCategory, 214
Cook, 216
Eat, 216
GetDuplicate, 216
HealthEffect, 217
HungerGain, 217
ReadyCategory, 217
Flexibility
SolidCategory, 586
FloatBehavior, 218
SetFloatHeight, 219
FloatableObject
ItemTypes, 377
FloodWater, 219
WaterLevelHeight, 220
FogStartThreshold
RainController, 526
FoodPoisoningChance
Player, 452
FormattedTime
Clock, 92
Forward
ControlScheme, 111
ForwardSecondary
ControlScheme, 112
FreezeLoading
CityChunkManager, 87
Fuel
ItemTypes, 377
FuelCategory, 221
BurnTime, 223
GetDuplicate, 223
ReadyCategory, 223
FuelRemaining
FireBaseCategory, 186
Game, 229
CityBounds, 231
CityInstance, 231
ClockInstance, 231
DeathManagerInstance, 232
DebugMode, 232
DebugModeDelegate, 231
DebugModeSubscription, 236
EventManager, 232
GameSettingsInstance, 232
GameViewInstance, 233
Instance, 233
IsRadioActive, 233
ItemFactoryInstance, 233
ItemPoolInstance, 233
Loader, 234
PauseInstance, 234
Player, 234
PlayerInstance, 234
RadioInstance, 234
Reset, 231
Scheme, 235
WaterLevelHeight, 235
WeatherInstance, 235
WorldItemFactoryInstance, 235
GameLoader, 236
CreateGameLoaderTask, 237
CurrentTask, 237
GameLoader, 237
PercentageComplete, 238
GameLoaderTask, 238
GameLoaderTask, 239
Name, 239
PercentageComplete, 239
GameSettings, 240
GameSettings, 241
ProceduralCityGenerationSeed, 241
Scheme, 241
SoundOn, 241
VolumeValue, 242
GameSettingsInstance
Game, 232
GameViewBehavior, 242
OnCraftingClick, 243
OnInventoryClick, 244
OnInventoryOpen, 244
OnPauseClick, 244
OnRadioClick, 244
OnResumeClick, 244
GameViewInstance
Game, 233
Generate
BlockGenerator, 54
BuildingGenerator, 85
DistrictGenerator, 147
GenerateInWater
WaterItemGeneration, 631
generateInitialPoints
WaterPointGenerator, 641
generateRandomPointAround
  SamplingPointGenerator, 564
GenerationUtility, 245
  AddNonDuplicateVertex, 246
  DistributeSeedPoints, 246
  GetMidpoint, 247
  GetMostCommonVertex, 247
  IsClockwise, 247
  IsPointInPolygon, 248
  SortVertices, 248
  ToAlignedVector2, 249
  ToAlignedVector3, 249
Generator
  ProceduralBuilding, 486
GetAllInteractableItemsByDistrict
  WorldItemFactory, 661
GetAllItemsWithTag
  Inventory, 289
GetAngle
  VectorUtility, 617
GetAttribute
  ItemCategory, 319
GetBaseItem
  ItemFactory, 329, 330
GetClimbHeight
  LandMovement, 387
  Movement, 417
  RaftMovement, 520
  WaterMovement, 634
GetClosestPressureSystem
  PressureSystems, 480
GetDocument
  FileManager, 180
GetDuplicate
  Attribute, 25
  ClothCategory, 97
  ContainerCategory, 108
  EquipableCategory, 167
  FirebaseCategory, 185
  FishingRodCategory, 211
  FleshCategory, 216
  FuelCategory, 223
  ItemCategory, 319
  LightCategory, 394
  MedicineCategory, 413
  PlantCategory, 447
  RaftCategory, 514
  ShelterCategory, 576
  SolidCategory, 584
  WarmthIdolCategory, 624
GetEdges
  Voroni, 621
GetEnumerator
  City, 77
GetInventory
  Inventory, 289
GetInventoryBaseItem
  Inventory, 289
GetInventoryContents
  InventoryYamlParser, 305
GetItemAttribute
  BaseItem, 35
GetItemAttributes
  BaseItem, 36
  ItemStackUI, 369
GetItemCategories
  BaseItem, 36
GetItemCategoryByClass
  BaseItem, 36
GetItemCategoryByParentClass
  BaseItem, 37
GetItemExtents
  ItemGenerator, 336
GetItemName
  ItemStackUI, 370
GetItemNamesByTag
  ItemFactory, 330
GetItemPool
  ItemPoolManager, 347
GetItemToModify
  BaseItem, 37
GetItemsByType
  Inventory, 290
GetMaxAmount
  ItemStackUI, 370
GetMidpoint
  GenerationUtility, 247
GetMostCommonVertex
  GenerationUtility, 247
GetMysteryAnnouncement
  Radio, 508
GetNextOpenSlot
  Inventory, 290
GetNumberOfActionsCompleted
  BaseItem, 37
GetOnlineDriveDocument
  GoogleDrive, 251
GetPerpindicularSlope
  VectorUtility, 618
GetPointsInWater
  WaterPointGenerator, 641
GetPossibleActions
  BaseItem, 38
  CollectableItem, 101
  ItemCategory, 319
  ItemStackUI, 370
GetRandom
  LoadingScreenFlavorTextManager, 405
GetRandomEntryByType
  DistrictConfiguration::ProceduralBuildingBases::<SizeXDefinition, 494
GetRandomItemIndex
  WorldItemFactory, 662
GetRarity
  ItemRarity, 350
  getRarityInformation
ItemGenerator, 336
GetRaycastHeight
LandMovement, 387
Movement, 417
RaftMovement, 521
WaterMovement, 635
GetRoofMeshes
ProceduralBuildingInstance, 500
GetRoofs
ProceduralBuildingInstance, 500
GetSelected
EquippedItemDropdown, 171
GetSlope
VectorUtility, 618
GetSprite
SpriteManager, 589
GetStack
ItemStackUI, 370
GetStackUI
InventoryUI, 299
GetStacks
Inventory, 290
GetToolByBaseItem
PlayerTools, 476
GetTwelveHourPercentage
Clock, 91
GetTypeLengthByType
DistrictConfiguration::ProceduralBuildingBases::SizeXDefinition, 494
GetValidPoints
RooftopPointGenerator, 561
GetWeatherAnnouncement
Radio, 508
GetWeightedRandomBaseItem
ItemFactory, 331
GetWeightedRandomItemIndex
DistrictItemRarityConfiguration, 152
ItemFactory, 331
GetterURL
GoogleDrive, 251
GoToMain
FaderManager, 179
GoogleDrive, 250
CertificateValidationCallback, 250
ConnectedToInternet, 251
GetOnlineDriveDocument, 251
GetterURL, 251
GoogleURL, 252
MethodType, 252
GoogleURL
GoogleDrive, 252
Greater
BooleanOperator, 56
GreaterOrEqual
BooleanOperator, 56
grid
ItemPoolManager, 349
SamplingPointGenerator, 565
GridLayoutManager, 252
CheckGridSize, 253
ElementsPerRow, 254
MaxRows, 254
SetGridSize, 253
UseMaxRows, 254
GridPoint
ItemPlacementSamplePoint, 339
HUDBehavior, 259
UpdateHealthSlider, 260
UpdateHungerSlider, 260
UpdateWarmthSlider, 261
HalfHour
Clock, 93
HalfHourDelegateUpdate
Clock, 91
HalfHourUpdate
Clock, 94
HandleAddSelectEvent
SelectedIngredientButton, 568
HandleItemAmountTextChangeEvent
ItemStackUI, 371
HandleItemIconChangeEvent
ItemStackUI, 371
HandleItemNameTextChangeEvent
ItemStackUI, 371
HasEquipped
PlayerTools, 477
HasOverlappingNeighbors
SamplingPointGenerator, 564
HasRoof
ProceduralBuildingAttachment, 489
ProceduralBuildingBase, 492
HasSelectionInterface, 254
ItemSelectedCallbackAction, 255
Heal
MedicineCategory, 413
Health
Player, 452
HealthEffect
FleshCategory, 217
HealthGain
MedicineCategory, 413
HealthRate
PlayerStatManager, 474
HealthRateManager, 256
HealthRateManager, 256
HealthRateManager, 256
HealthRateManager, 256
TakeFallDamage, 258
UseDefaultHealthRate, 258
UseHealthEnergy, 258
UseHungerZeroHealthReductionRate, 259
UseWarmthZeroHealthReductionRate, 259
HealthStatus
Player, 452
HeightType
ProceduralBuildingBase, 492
ProceduralBuildingCreator.cs, 671
HideHover
ItemAttributeUI, 314

HideOptions
  EquippedItemDropdown, 171

HidePanel
  RecipePageBehavior, 545

hideThese
  EndingController, 163

highMusic
  Radio, 509

highMystery
  Radio, 509

highWeather
  Radio, 509

Highlight
  RecipeButtonGUIBehavior, 539

HighlightColor
  RecipeButtonGUIBehavior, 540

Hook
  ItemTypes, 377

Hour
  Clock, 93

HourDelegateUpdate
  Clock, 91

HourUpdate
  Clock, 94

HoverPanel
  ItemStackUI, 373

Hunger
  Player, 453

HungerGain
  FleshCategory, 217

HungerRate
  PlayerStatManager, 474

HungerRateManager, 261
  HungerRateManager, 262
  UseDefaultHungerReductionRate, 263
  UseFoodEnergy, 263

Id
  ItemStack, 360

IdealCreatureCount
  CreatureManager, 120

Idle
  LandMovement, 388
  Movement, 417
  RaftMovement, 521
  WaterMovement, 635

Idol, 264
  Equip, 266
  SetUpTool, 266
  Unequip, 266
  Use, 266

IdolCategory, 268
  ApplyBenefit, 270
  Equip, 270
  ReadyCategory, 270
  RemoveBenefit, 270

IgniteFire
  FireInteractable, 189

Igniter
  ItemTypes, 377

Impermiability
  ClothCategory, 99

InFieldOfView
  FishAgent, 194

InUse
  Tool, 610

Index
  CreatureTracker, 126

Ingredient, 271
  Amount, 272
  AssociatedStackId, 272
  Ingredient, 272
  IngredientName, 272
  UseType, 273

IngredientButtonBehavior, 273
  DeselectIngredient, 274
  SelectIngredient, 275
  SetUpIngredient, 275
  UpdateIngredientEvent, 275
  UpdateIngredientSelection, 275

IngredientName
  Ingredient, 272

Init
  CityChunkManager, 86
  CreatureSpawner, 123
  FishSpawner, 213

InitializeBaseItem
  BaseItem, 38

InsertRecipeButton
  RecipeBookBehavior, 535

Instance
  Building, 60
  Game, 233
  NoteUIController, 431

Interactable
  PlayerController, 463

interactable
  NoteData, 425

InteractableObject, 276
  PerformAction, 277
  SetAction, 277
  SetUp, 278
  setupComplete, 278
  Show, 278
  Text, 278

InteractableRadioModel, 279
  ActivateRadio, 281
  DeactivateRadio, 281
  PushPowerButton, 281
  PushVolumeDownButton, 281
  PushVolumeUpButton, 281
  SetKnobRotation, 281
  SetSlider, 282
  SetUpRadioCanvas, 282

InteractableSetUp, 283
  ItemName, 284
Inventory, 284
  AddItem, 287
  CheckRecipePossible, 288
  CheckRequirementMet, 288
  CombineStacks, 288
  contents, 292
  ControlScheme, 112
  GetAllItemsWithTag, 289
  GetInventory, 289
  GetInventoryBaseItem, 289
  GetItemsByType, 289
  GetNextOpenSlot, 289
  GetStacks, 290
  Inventory, 286, 287
  InventorySize, 292
  LoadInventory, 291
  Player, 453
  RemoveStack, 291
  UpdateTypeAmount, 291
  UseItem, 292
InventoryInteractable, 293
  AttachedInventory, 295
  SetUp, 294
InventoryItemYAMLModel, 295
  Item, 296
  ItemAmount, 296
  ItemCategories, 296
  StackId, 297
InventoryName
  InventoryYAMLModel, 301
InventorySize
  Inventory, 292
  InventoryYAMLModel, 302
InventorySprite
  BaseItem, 41
  ItemStackUI, 374
  NoteData, 426
InventoryUI, 297
  DisplayInventory, 299
  GetStackUI, 299
  ItemSpriteManager, 300
  ItemsToDiscard, 300
  LoadNewInventory, 300
  RefreshInventoryPanel, 300
  TargetInventory, 300
InventoryYAMLModel, 301
  InventoryName, 301
  InventorySize, 302
  Items, 302
InventoryYAMLParser, 303
  AddNewInventory, 305
  GetInventoryContents, 305
  InventoryYAMLParser, 305
  LoadInventories, 306
  SaveInventoriesToFile, 306
  SaveInventory, 306
IsActive
  RaftMovement, 523
IsByFire
  PlayerController, 463
IsClockwise
  GenerationUtility, 247
IsDay
  Clock, 93
IsDead
  CreatureTracker, 126
  isFullfilled
    Requirement, 553
IsGrounded
  PlayerController, 463
IsHighPressure
  PressureSystem, 478
IsInShelter
  PlayerController, 464
IsInWater
  PlayerController, 464
IsInWorld
  Player, 453
IsLoaded
  Building, 60
  CityChunk, 83
IsOnLand
  PlayerController, 464
IsOnRaft
  PlayerController, 464
IsPaused
  PauseSystem, 440
IsPointInPolygon
  GenerationUtility, 248
IsRadioActive
  Game, 233
IsReading
  PlayerController, 464
IsReeling
  FishingLure, 204
IsWaterInView
  PlayerController, 465
Item
  InventoryItemYAMLModel, 296
  ItemStack, 360
  PickUpItem, 443
ItemAction, 306
  ActionComplete, 308
  ActionName, 308
  AssignedAction, 308
  Conditions, 308
  ItemAction, 307
  SubActions, 309
  TypeUsed, 309
ItemActionButtonUI, 310
  PerformAction, 311
  SetAction, 311
  ShowSubActions, 312
  StartAction, 312
  SubActions, 312
ItemAddedDelegate
Legendary, 350
Rare, 350
Uncommon, 351
ItemRemovedDelegate
PlayerInventory, 470
ItemRemovedSubscription
PlayerInventory, 472
ItemSelectedCallbackAction
HasSelectionInterface, 255
ItemSerializer, 351
DeserializeDistrictItemData, 353
DeserializeItemInformation, 353
ItemSerializer, 352
ItemSpawner, 354
MaxSpawnNumber, 356
MinSpawnNumber, 356
PosterPositions, 357
PosterRotationModMax, 357
SetUpSpawner, 356
SpawnAfterActions, 356
SpawnRadius, 357
SpawnWithoutInteraction, 357
ItemSpriteManager
InventoryUI, 300
ItemStack, 358
Amount, 359
Id, 360
Item, 360
ItemStack, 359
UpdateStackAmount, 360
UpdateStackAmountEvent, 359
ItemStackDetailPanelBehavior, 361
Attributes, 365
ClearAttributesAndActions, 363
ClearSubActionPanel, 363
ClosePanel, 363
CreateSubAction, 363
DisplayAttributesAndAction, 364
PossibleActions, 365
PossibleSubActions, 366
RefreshItemActions, 364
RefreshItemPanel, 364
SetItemData, 364
SetSelectedItem, 365
UpdateSelectedAmount, 365
ItemStackUI, 367
Action, 373
CheckForModification, 369
GetItemAttributes, 369
GetItemName, 370
GetMaxAmount, 370
GetPossibleActions, 370
GetStack, 370
HandleItemAmountTextChangedEvent, 371
HandleItemIconChangeEvent, 371
HandleItemNameTextChangedEvent, 371
HoverPanel, 373
InventorySprite, 374
ItemAmount, 374
itemName, 374
PreserveOriginal, 372
RefreshInventoryItem, 372
SetHoverPanelActive, 372
SetUpInventoryItem, 372
Unsubscribe, 373
UpdateTargetStack, 373
ItemTemplates
DistrictItemConfiguration, 150
ItemType
Requirement, 554
ItemTypes, 374
AdvancedTool, 375
BaseSolid, 375
Blade, 376
Cloth, 376
Container, 376
Edible, 376
Equipable, 376
FloatableObject, 377
Fuel, 377
Hook, 377
Igniter, 377
Medicinal, 377
Rod, 378
Rope, 378
Sharp, 378
Types, 378
ItemUnequippedDelegate
PlayerInventory, 470
ItemUnequippedSubscription
PlayerInventory, 472
ItemYAMLMap, 379
BaseItem, 379
ItemCategories, 379
itemName, 380
Items
InventoryYAMLModel, 302
ItemPoolInfo, 343
ItemsToDiscard
InventoryUI, 300
JellyFishSpawner, 381
Jitter
FishAgentConfig, 200
Jump
ControlScheme, 112
LandMovement, 388
Movement, 417
RatMovement, 521
WaterMovement, 635
JumpForce
LandMovement, 388
JumpLand
LandMovement, 389
KeyCodeConfigMenuBehavior, 383
OnCloseClick, 384
MaxAttempts
  WaterPointGenerator, 643
maxCreatureCount
  CreatureSpawner, 124
MaxFieldOfViewAngle
  FishAgentConfig, 200
MaxFloors
  DistrictConfiguration, 142
MaxFuel
  LightCategory, 395
MaxHealth
  Player, 453
MaxHunger
  Player, 453
MaxLowerDistance
  WindowWasher, 658
MaxNumberOfItems
  WaterPointGenerator, 643
MaxRadius
  FishAgentConfig, 200
MaxRows
  GridLayoutManager, 254
MaxSpawnNumber
  ItemSpawner, 356
MaxStat
  StatRate, 592
MaxVelocity
  FishAgentConfig, 201
MaxWarmth
  Player, 454
MaxWeight
  FishAgentConfig, 201
meanCreatureCount
  CreatureSpawner, 124
Medicinal
  ItemTypes, 377
MedicineCategory, 410
  CureAll, 412
  CureFoodPoisoning, 412
  CurePneumonia, 412
  GetDuplicate, 413
  Heal, 413
  HealthGain, 413
  ReadyCategory, 413
  Sickness, 414
MenuPause
  PauseSystem, 439
  PlayerController, 461
MenuPauseDelegateUpdate
  PauseSystem, 439
MenuPauseUpdate
  PauseSystem, 440
MethodType
  GoogleDrive, 252
MinAttachmentChance
  DistrictConfiguration, 145
minCreatureCount
  CreatureSpawner, 124
MinDistance
  ItemPlacementSamplePoint, 339
MinFloors
  DistrictConfiguration, 142
MinSpawnNumber
  ItemSpawner, 356
ModifyingActionNames
  BaseItem, 42
Move
  LandMovement, 389
  Movement, 417
  RaftMovement, 521
  WaterMovement, 636
MoveNext
  City::BuildingEnumerator, 63
Movement, 415
  AccumulatedFallDamage, 418
  Climb, 417
  climbHeight, 418
  CurrentFallDammage, 419
  GetClimbHeight, 417
  GetRaycastHeight, 417
  Idle, 417
  Jump, 417
  Move, 417
  OnStateEnter, 418
  OnStateExit, 418
  RigidBody, 418
  Speed, 419
Music_Clicked
  TestGUI, 602
MusicDefaultPath
  Radio, 510
MysteryAnnouncementPath
  Category, 420
  EventPath, 420
  Speed, 420
  Text, 421
  Voice, 421
MysteryDefaultPath
  Radio, 510
Name
  Attribute, 25
  District, 139
  DistrictConfiguration, 145
  GameLoaderTask, 239
New<T1, T2>
  Tuple, 613
newClip
  TestGUI, 603
NewPointsPerSamplingPoint
  WaterPointGenerator, 643
NextStep
  RecipePageBehavior, 545
NormalizedOctantWindDirection2d
  WeatherSystem, 648
NotEqual
  BooleanOperator, 57
Note, DisplayText, Read, SetUpNote, NoteData, interactable, InventorySprite, Text, Title, Type, WorldModel, NoteFactory, CreateInteractableItem, LoadTemplates, NoteText, NoteUIController, DisableUI, NoteUIPanel, SetText, NoteUIPanel, NoteUIController, NoteYAMLParser, LoadNotes, NullParent, NumberOfFloors, ProceduralBuilding, NumberOfInitialPoints, WaterPointGenerator, On LightItem, OnBackToMainClick, OnBeginCraftingClick, OnChannelClick, OnCloseClick, OnConfigKeysClick, OnCraftingClick, OnExitClick, OnGUI, BaselinePopup, OnHelpClick, OnInventoryClick, OnInventoryOpen, OnKeyButtonClick, OnPauseClick, OnPlayer, ClothCategory, onPlayerAttributeName, ClothCategory, OnRadioClick, OnResumeClick, OnSeedValueChanged, OnSettingsClick, OnStartClick, OnStateEnter, OnStateExit, OnTriggerEnter, OnTriggerExit, OnTriggerExit, OpenItemAmountPanel, OpenItemDetailPanel, OptionSelectedCallbackAction, OverworldItemOptionSelection, AddPossibleAction, OptionSelectedCallbackAction, OverworldItemOptionSelection, Reset, SelectedItem, ShowPossibleActions, ShowPossibleItems, TargetContainerPanel, TargetContentPanel, Parent, Building, TransformParent, PartNumber, buildingIndex, Pause, ControlScheme, PauseSystem, PlayerController.
PauseDelegateUpdate
PauseSystem, 439
PauseInstance
Game, 234
PauseSystem, 438
IsPaused, 440
MenuPause, 439
MenuPauseDelegateUpdate, 439
MenuPauseUpdate, 440
Pause, 439
PauseDelegateUpdate, 439
PauseSystem, 439
PauseUpdate, 440
Resume, 440
ResumeDelegateUpdate, 440
ResumeUpdate, 441
PauseUpdate
PauseSystem, 440
PerSeconds
StatRate, 592
PercentageComplete
GameLoader, 238
GameLoaderTask, 239
PerformAction
InteractableObject, 277
ItemActionButtonUI, 311
PickUpItem, 441
Amount, 443
Item, 443
SetUp, 443
PlacementCenter
WindowWasher, 658
PlacementFrequency
DistrictConfiguration::BuildingTemplatePlacement, 68
PlantCategory, 444
Bitter, 447
Cook, 446
Dry, 446
Eat, 447
GetDuplicate, 447
PneumoniaEffect, 448
ReadyCategory, 447
Salty, 448
Sour, 448
Spicy, 448
StomachEffect, 449
Sweet, 449
Toughness, 449
WaterContent, 449
Player, 450
Controller, 452
FoodPoisoningChance, 452
Game, 234
Health, 452
HealthStatus, 452
Hunger, 453
Inventory, 453
IsInWorld, 453
MaxHealth, 453
MaxHunger, 453
MaxWarmth, 454
Player, 451
ResetStatus, 452
Toolbox, 454
Warmth, 454
WorldPosition, 454
WorldTransform, 454
Player.cs
PlayerHealthStatus, 696
PlayerAnimationEvents, 455
CallJumpForce, 456
CallJumpLand, 456
PlayerAnimator
PlayerController, 465
RaftMovement, 523
PlayerBoardRaftDelegate
EventManager, 174
PlayerBoardRaftSubscription
EventManager, 176
PlayerController, 457
BoardRaft, 460
CheckClosestInteractable, 460
ClimbingRaycastMask, 462
ClosestItem, 460
DirFromAngle, 460
DisembarkRaft, 461
Interactable, 463
IsByFire, 463
IsGrounded, 463
IsInShelter, 464
IsInWater, 464
IsOnLand, 464
IsOnRaft, 464
IsReading, 464
IsWaterInView, 465
MenuPause, 461
Pause, 461
PlayerAnimator, 465
PlayerIKSetUp, 462
PlayerStatManager, 463
Resume, 461
SetIsOnLand, 462
StartWalkingSound, 462
StopWalkingSound, 462
ViewAngle, 463
ViewRadius, 463
PlayerDatabase
AnnouncementFactory, 22
PlayerFocus, 465
PlayerHealthStatus
Player.cs, 696
PlayerIKSetUp
PlayerController, 462
PlayerInstance
Game, 234
AddItem, 470
EquippedItem, 471
ItemAddedDelegate, 470
ItemAddedSubscription, 471
ItemEquippedDelegate, 470
ItemEquippedSubscription, 471
ItemRemovedDelegate, 470
ItemRemovedSubscription, 472
ItemUnequippedDelegate, 472
ItemUnequippedSubscription, 472
PlayerInventory, 469
RemoveStack, 471
PlayerStandHeight
PlayerStatManager, 472
AddItem, 470
EquippedItem, 471
ItemAddedDelegate, 470
ItemAddedSubscription, 471
ItemEquippedDelegate, 470
ItemEquippedSubscription, 471
ItemRemovedDelegate, 470
ItemRemovedSubscription, 472
ItemUnequippedDelegate, 472
ItemUnequippedSubscription, 472
PlayerInventory, 469
RemoveStack, 471
PlayerStatManager, 472
ApplyCorrectHealthReductionRate, 473
HealthRate, 474
HungerRate, 474
PlayerController, 463
PlayerStatManager, 473
StartStatUpdates, 474
StopStatUpdates, 474
StopStats, 475
WarmthRate, 475
PlayerTools, 475
EquippedTool, 477
GetToolByBaseItem, 476
HasEquipped, 477
PlayerTools, 476
PneumoniaEffect
PlantCategory, 448
PointToGrid
ItemPoolManager, 347
RooftopPointGenerator, 561
WaterPointGenerator, 642
PointType
ItemPlacementSamplePoint, 338
poolManager
ItemGenerator, 337
PopulateRoof
RooftopGeneration, 558
Position
Building, 61
FishingLure, 204
PressureSystem, 478
PossibleActions
ItemStackDetailPanelBehavior, 365
PossibleSubActions
ItemStackDetailPanelBehavior, 366
PosterPositions
ItemSpawner, 357
PosterRotationModMax
ItemSpawner, 357
Power
Radio, 508
predatorLayer
FishAgent, 196
Prediction
Regression, 551
PreserveOriginal
ItemStackUI, 372
Pressure
PressureSystem, 478
PressureConstants
PressureSystems.cs, 704
PressureSystem, 477
IsHighPressure, 478
Position, 478
Pressure, 478
PressureSystemVisualization, 481
PressureSystems, 479
CityBounds, 481
GetClosestPressureSystem, 480
LocalPressureSystems, 481
LowPressureSystems, 480
PressureSystems, 479
UpdatePressureSystem, 480
PressureSystems.cs
PressureConstants, 704
PrintWeather
ConsoleCommandRouter, 104
ProceduralBuilding, 482
BuildingScale, 486
Configuration, 486
Generator, 486
Load, 485
NumberOfFloors, 486
ProceduralBuilding, 485
Seed, 487
ProceduralBuildingAttachment, 487
AddWindows, 489
AttachmentPoint, 489
HasRoof, 489
RooftopLocation, 489
WindowPoints, 489
ProceduralBuildingBase, 490
AddWindows, 492
AttachmentPoints, 492
HasRoof, 492
HeightType, 492
RooftopLocation, 492
StackableObject, 492
WindowPoints, 493
ProceduralBuildingCreator, 495
CreateBuilding, 497
CreateTestBuilding, 497
TestAttachmentPercentage, 497
TestDistrict, 497
TestSize, 497
TestStoriesTall, 498
ProceduralBuildingCreator.cs
BaseSize, 670
BuildingParts, 671
HeightType, 671
ProceduralBuildingInstance, 498
ItemPoolManager, 348
RemoveItemInfo
ItemPoolInfo, 342
RemoveSelection
SelectedIngredientButton, 568
RemoveStack
Inventory, 291
PlayerInventory, 471
RendererComponent
BezierLine, 49
Requirement, 552
AmountRequired, 554
isFilled, 553
ItemType, 554
Requirement, 553
SubmitItem, 553
RequirementInstance
RecipeRequirementsUI, 548
Reset
City::BuildingEnumerator, 63
Game, 231
OverworldItemOptionSelection, 436
ResetCraftingPanel
RecipePageBehavior, 545
ResetPanel
RecipeBookBehavior, 536
ResetStatus
Player, 452
ResourceRequirements
Recipe, 532
Restart
RestartBehavior, 555
RestartBehavior, 554
Restart, 555
Resume
PauseSystem, 440
PlayerController, 461
ResumeDelegateUpdate
PauseSystem, 440
ResumeUpdate
PauseSystem, 441
Right
ControlScheme, 113
RightSecondary
ControlScheme, 113
RigidBody
Movement, 418
Rod
ItemTypes, 378
RoofLocation
ProceduralBuildingAttachment, 489
ProceduralBuildingBase, 492
RoofTopGeneration, 556
AddDoorsToDistrict, 558
AddSheltersToDistrict, 558
PopulateRoof, 558
RoofTopPointGenerator, 559
GetValidPoints, 561
PointToGrid, 561
Rope
ItemTypes, 378
RotateLeft
CameraController, 70
RotateRight
CameraController, 70
Salty
PlantCategory, 448
SamplingPointGenerator, 562
cellSize, 565
defaultMinDistanceAway, 565
generateRandomPointAround, 564
grid, 565
HasOverlappingNeighbors, 564
verifyRaycastHit, 564
waterTag, 565
SaveAllDocuments
FileManager, 181
SaveFile
FileManager, 181
SavelInventoriesToFile
InventoryYamlParser, 306
SavelInventory
InventoryYamlParser, 306
Scheme
Game, 235
GameSettings, 241
Second
WeightedPair, 651
SecondDelegateUpdate
Clock, 92
SecondUpdate
Clock, 94
Seed
ProceduralBuilding, 487
SeedPoint
District, 139
SelectIngredient
IngredientButtonBehavior, 275
SelectedBackColor
RecipeButtonGUIBehavior, 540
SelectedColor
RecipeButtonGUIBehavior, 541
SelectedIngredientButton, 566
Amount, 569
AssociatedStackId, 569
HandleAddSelectEvent, 568
ItemName, 569
RemoveSelection, 568
SetUpSelection, 568
Unsubscribe, 568
SelectedItem
ChooseItemAmountPanelBehavior, 75
OverworldItemOptionSelection, 437
Separation
FishAgent, 194
SeparationRadius
FishAgentConfig, 201
SeparationWeight
FishAgentConfig, 201
SetAction
  InteractableObject, 277
  ItemActionButtonUI, 311
  WorldSelectionButtonBehavior, 664
SetActionComplete
  ItemCategory, 320
setAnimationSpeed
FXSplash, 225
SetAttributeName
  ItemAttributeUI, 314
SetAttributeValue
  ItemAttributeUI, 314
SetBaseItem
  ItemCategory, 320
SetChannel
  Radio, 508
SetCityInformation
  WaterItemGeneration, 631
SetCreatureGUI, 570
SetDown
  ContainerCategory, 108
  FireBaseCategory, 185
  RaftCategory, 514
  ShelterCategory, 576
SetEnvironmentGUI, 571
SetEquipped
  EquippedItemDropdown, 171
SetFloatHeight
  FloatBehavior, 219
setGraffitiText
  TutorialGraffiti, 615
SetGridSize
  GridLayoutManager, 253
SetHealth
  ConsoleCommandRouter, 104
SetHoverPanelActive
  ItemStackUI, 372
SetIsOnLand
  PlayerController, 462
SetItemData
  ItemStackDetailPanelBehavior, 364
SetKey
  KeyCodeConfigMenuBehavior, 384
SetKnobRotation
  InteractableRadioModel, 282
setLayers
  FishAgent, 195
SetMaxSpeed
  RaftMovement, 522
SetNewModel
  BaselineItem, 39
SetSeed
  ItemGenerator, 336
SetSelectedIndex
  ItemStackDetailPanelBehavior, 365
SetSlider
  InteractableRadioModel, 282
SetText
  NoteUIController, 430
SetTimeScale
  ConsoleCommandRouter, 105
SetUnitsInShelter
  WarmthRateManager, 628
SetUp
  FireInteractable, 189
  InteractableObject, 278
  InventoryInteractable, 294
  PickUpItem, 443
  RaftInteractable, 517
  ShelterInteractable, 579
SetUpBaseItem
  BaselineItem, 39
SetUpButton
  RecipeButtonGUIBehavior, 539
SetUpCategoryInformation
  CraftingSystemSerializer, 116
SetUpIngredient
  IngredientButtonBehavior, 275
SetUpInventoryItem
  ItemStackUI, 372
SetUpItemPoolManager
  ItemPoolManager, 348
SetUpNote
  Note, 424
SetUpRadioCanvas
  InteractableRadioModel, 282
SetUpRecipePage
  RecipePageBehavior, 545
SetUpRequirement
  RecipeRequirementsUI, 548
SetUpSelection
  SelectedIngredientButton, 568
SetUpSpawner
  ItemSpawner, 356
SetUpTool
  Barometer, 28
  FishingRod, 207
  Idol, 266
  LightItem, 398
  Thermometer, 606
  Tool, 609
SetUpVoseAlias
  DistrictItemRarityConfiguration, 152
SettingsMenuBehavior, 572
changeVolume, 573
OnConfigKeysClick, 573
OnSeedValueChanged, 573
turnSoundOnOff, 573
setupComplete
  InteractableObject, 278
Sharp
  ItemTypes, 378
Sharpen
<table>
<thead>
<tr>
<th>StatAffectingItems</th>
<th>CraftingStat, 114</th>
</tr>
</thead>
<tbody>
<tr>
<td>StatName</td>
<td>CraftingStat, 115</td>
</tr>
<tr>
<td>StatRate</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ApplyRateToStat, 591</td>
</tr>
<tr>
<td></td>
<td>ChangeRateValues, 592</td>
</tr>
<tr>
<td></td>
<td>CurrentStat, 592</td>
</tr>
<tr>
<td></td>
<td>MaxStat, 592</td>
</tr>
<tr>
<td></td>
<td>PerSeconds, 592</td>
</tr>
<tr>
<td></td>
<td>StatRate, 591</td>
</tr>
<tr>
<td></td>
<td>Units, 593</td>
</tr>
<tr>
<td>StaticDefaultPath</td>
<td>Radio, 510</td>
</tr>
<tr>
<td>StatsToCheck</td>
<td>Recipe, 532</td>
</tr>
<tr>
<td>Stickiness</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SolidCategory, 586</td>
</tr>
<tr>
<td>StomachEffect</td>
<td>PlantCategory, 449</td>
</tr>
<tr>
<td>StopStatUpdates</td>
<td>PlayerStatManager, 474</td>
</tr>
<tr>
<td>StopStats</td>
<td>PlayerStatManager, 475</td>
</tr>
<tr>
<td>StopWalkingSound</td>
<td>PlayerController, 462</td>
</tr>
<tr>
<td>StormStart</td>
<td>EventManager, 175</td>
</tr>
<tr>
<td>StormStartedDelegate</td>
<td>EventManager, 175</td>
</tr>
<tr>
<td>StormStartedSubscription</td>
<td>EventManager, 176</td>
</tr>
<tr>
<td>StormStop</td>
<td>EventManager, 175</td>
</tr>
<tr>
<td>StormStoppedDelegate</td>
<td>EventManager, 175</td>
</tr>
<tr>
<td>StormStoppedSubscription</td>
<td>EventManager, 176</td>
</tr>
<tr>
<td>SubActions</td>
<td>ItemAction, 309</td>
</tr>
<tr>
<td></td>
<td>ItemActionButtonUI, 312</td>
</tr>
<tr>
<td>SubmitItem</td>
<td>Requirement, 553</td>
</tr>
<tr>
<td>Sweet</td>
<td>PlantCategory, 449</td>
</tr>
<tr>
<td>SwimmingHeight</td>
<td>WaterMovement, 637</td>
</tr>
<tr>
<td>TakeFallDamage</td>
<td>HealthRateManager, 258</td>
</tr>
<tr>
<td>TakeOff</td>
<td>ClothCategory, 98</td>
</tr>
<tr>
<td>TallestBuilding</td>
<td>City, 77</td>
</tr>
<tr>
<td>TargetContainerPanel</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OverworldItemOptionSelection, 437</td>
</tr>
<tr>
<td>TargetContentPanel</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OverworldItemOptionSelection, 437</td>
</tr>
<tr>
<td>TargetInventory</td>
<td></td>
</tr>
<tr>
<td></td>
<td>InventoryUI, 300</td>
</tr>
<tr>
<td>TeleportPlayer</td>
<td></td>
</tr>
<tr>
<td>TeleportToA</td>
<td>TeleportToB, 596</td>
</tr>
<tr>
<td></td>
<td>WindowWasher, 657</td>
</tr>
<tr>
<td>TeleportPlayer::TeleportLocation</td>
<td></td>
</tr>
<tr>
<td>TeleportPlayer:::TeleportLocation</td>
<td>CanTeleportTo, 594</td>
</tr>
<tr>
<td></td>
<td>Location, 594</td>
</tr>
<tr>
<td>TeleportToEnding</td>
<td>ConsoleCommandRouter, 105</td>
</tr>
<tr>
<td>TeleportToA</td>
<td>TeleportPlayer, 596</td>
</tr>
<tr>
<td>TeleportToB</td>
<td>TeleportPlayer, 596</td>
</tr>
<tr>
<td>TemperatureVariance</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DiurnalTemperatureVariance, 156</td>
</tr>
<tr>
<td>Template</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TemplateBuilding, 599</td>
</tr>
<tr>
<td>TemplateBuilding</td>
<td>Load, 599</td>
</tr>
<tr>
<td></td>
<td>Template, 599</td>
</tr>
<tr>
<td></td>
<td>TemplateBuilding, 598</td>
</tr>
<tr>
<td>Test</td>
<td>ConsoleCommandRouter, 105</td>
</tr>
<tr>
<td>TestAttachementPercentage</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ProceduralBuildingCreator, 497</td>
</tr>
<tr>
<td>TestDistrict</td>
<td>ProceduralBuildingCreator, 497</td>
</tr>
<tr>
<td>TestGUI</td>
<td>600</td>
</tr>
<tr>
<td></td>
<td>audios, 602</td>
</tr>
<tr>
<td></td>
<td>buttons, 603</td>
</tr>
<tr>
<td></td>
<td>Music_Clicked, 602</td>
</tr>
<tr>
<td></td>
<td>newClip, 603</td>
</tr>
<tr>
<td></td>
<td>radio, 603</td>
</tr>
<tr>
<td></td>
<td>ShowRadioButtons, 602</td>
</tr>
<tr>
<td></td>
<td>soundA, 603</td>
</tr>
<tr>
<td></td>
<td>soundB, 603</td>
</tr>
<tr>
<td></td>
<td>Weather_Clicked, 602</td>
</tr>
<tr>
<td>TestSize</td>
<td>ProceduralBuildingCreator, 497</td>
</tr>
<tr>
<td>TestStoriesTall</td>
<td>ProceduralBuildingCreator, 498</td>
</tr>
<tr>
<td>Text</td>
<td>InteractableObject, 278</td>
</tr>
<tr>
<td></td>
<td>LoadingScreenFlavorText, 403</td>
</tr>
<tr>
<td></td>
<td>MysteryAnnouncement, 421</td>
</tr>
<tr>
<td></td>
<td>NoteData, 426</td>
</tr>
<tr>
<td>Thermometer</td>
<td>604</td>
</tr>
<tr>
<td></td>
<td>CheckThermometer, 607</td>
</tr>
<tr>
<td></td>
<td>Equip, 606</td>
</tr>
<tr>
<td></td>
<td>SetUpTool, 606</td>
</tr>
<tr>
<td></td>
<td>Unequip, 606</td>
</tr>
<tr>
<td></td>
<td>Use, 606</td>
</tr>
<tr>
<td>Thickness</td>
<td>SolidCategory, 587</td>
</tr>
<tr>
<td>ThreadDensity</td>
<td>ClothCategory, 99</td>
</tr>
</tbody>
</table>
Threshold
  WeightedPair, 651
ThresholdValue
  ItemCondition, 324
Tick
  Clock, 93
Tiered
  Recipe, 533
TimeDatabase
  AnnouncementFactory, 22
Title
  NoteData, 426
ToAlignedVector2
  GenerationUtility, 249
ToAlignedVector3
  GenerationUtility, 249
ToString
  WeatherSystem, 647
ToggleDisplay
  EquippedItemDropdown, 172
ToggleIngredientSelectionPanel
  RecipePageBehavior, 546
 ToggleOn
  LightItem, 399
Tool, 607
  attachJoint, 609
  Equip, 609
  InUse, 610
  SetUpTool, 609
  ToolName, 610
  toolName, 610
  Unequip, 609
  unequipActName, 610
  Use, 609
ToolName
  Tool, 610
toolName
  Tool, 610
ToolRequirements
  Recipe, 533
Toolbox
  Player, 454
Toughness
  PlantCategory, 449
TransformParent
  Parent, 612
  UsePositionX, 612
  UsePositionY, 612
  UsePositionZ, 612
Tuple
  New< T1, T2 >, 613
  X, 614
  Y, 614
Tuple< T1, T2 >, 613
TurnLightOn
  BeaconInteractable, 47
turnSoundOnOff
  SettingsMenuBehavior, 573
TutorialGraffiti, 614
  setGraffitiText, 615
TwelveHours
  Clock, 94
TwentyFourHours
  Clock, 94
twoDimensional3d
  VectorUtility, 619
Type
  ItemPlacementSamplePoint, 340
  NoteData, 426
TypeUsed
  ItemAction, 309
Types
  BaseItem, 42
  ItemTypes, 378
uGUITools, 616
UnEquip
  EquipableCategory, 168
UnHighlight
  RecipeButtonGUIBehavior, 539
Uncommon
  ItemRarity, 351
Unequip
  Barometer, 28
  FishingRod, 207
  Idol, 266
  LightItem, 399
  Thermometer, 606
  Tool, 609
unequipActName
  Tool, 610
unequipActionName
  EquipableCategory, 168
Unequipped
  EquippedItemDropdown, 172
Units
  StatRate, 593
Unload
  Building, 59
  CityChunk, 83
UnlockRecipe
  RecipeBookBehavior, 536
Unselect
  RecipeButtonGUIBehavior, 540
Unsubscribe
  ItemStackUI, 373
  SelectedIngredientButton, 568
Update
  WindowWasher, 658
UpdateCreatureCount
  CreatureManager, 120
UpdateCreatureInfo
  CreatureManager, 120
UpdateExistingFlag
  BaseItem, 40
UpdateFire
  FireInteractable, 190

Generated by Doxygen
UpdateHealthSlider
HUDBehavior, 260
UpdateHungerSlider
HUDBehavior, 260
UpdateIngredientEvent
IngredientButtonBehavior, 275
UpdateIngredientSelection
IngredientButtonBehavior, 275
UpdateItemEvent
BaseItem, 40
UpdateItemName
BaseItem, 43
UpdateItemSprite
BaseItem, 43
UpdatePressureSystem
PressureSystems, 480
UpdateRopePosition
WindowWasher, 658
UpdateSelectedAmount
ItemStackDetailPanelBehavior, 365
UpdateSelection
RecipePageBehavior, 546
UpdateStackSize
ItemStack, 360
UpdateStackSizeAmountEvent
ItemStack, 359
UpdateSystem
WeatherSystem, 648
UpdateTargetStack
ItemStackUI, 373
UpdateTypeAmount
Inventory, 291
UpdateWarmthSlider
HUDBehavior, 261
UpdateWeather
WeatherSystem, 648
uriPrefix
CraftingSystemSerializer, 117
Use
Barometer, 28
FishingRod, 207
Idol, 266
LightItem, 399
Thermometer, 606
Tool, 609
UseClothRate
WarmthRateManager, 628
UseCustomValues
DebrisController, 133
RainController, 526
UseDefaultHealthRate
HealthRateManager, 258
UseDefaultHungerReductionRate
HungerRateManager, 263
UseDefaultWarmthReductionRate
WarmthRateManager, 628
UseFirePit
FireInteractable, 190
UseFoodEnergy
HungerRateManager, 263
UseHealthEnergy
HealthRateManager, 258
UseHeatSourceWarmthIncreaseRate
WarmthRateManager, 628
UseHungerZeroHealthReductionRate
HealthRateManager, 259
UseItem
Inventory, 292
UseMaxRows
GridLayoutManager, 254
UsePositionX
TransformParent, 612
UsePositionY
TransformParent, 612
UsePositionZ
TransformParent, 612
UseTool
ControlScheme, 113
UseType
Ingredient, 273
UseWarmthZeroHealthReductionRate
HealthRateManager, 259
UseWaterWarmthReductionRate
WarmthRateManager, 629
Value
Attribute, 25
VectorUtility, 617
GetAngle, 617
GetPerpendicularSlope, 618
GetSlope, 618
twoDimensional3d, 619
XZ, 619
Velocity
FishAgent, 196
verifyRaycastHit
SamplingPointGenerator, 564
Vertices
Block, 52
ViewAngle
PlayerController, 463
ViewRadius
PlayerController, 463
Voice
MysteryAnnouncement, 421
VolumeValue
GameSettings, 242
Voroni
Center, 621
GetEdges, 621
Voroni, 620
Wander
FishAgent, 195
WanderDistanceRadius
FishAgentConfig, 201
WanderRadius
FishAgentConfig, 201
WanderWeight
FishAgentConfig, 202
Warmth
Player, 454
WarmthBenefit
WarmthIdolCategory, 625
WarmthIdolCategory, 622
ApplyBenefit, 624
GetDuplicate, 624
ReadyCategory, 624
RemoveBenefit, 625
WarmthBenefit, 625
WarmthRate
PlayerStatManager, 475
Category, 577
WarmthRateManager, 626
SetUnitsInShelter, 628
UseClothRate, 628
UseDefaultWarmthReductionRate, 628
UseHeatSourceWarmthIncreaseRate, 628
UseWaterWarmthReductionRate, 629
WarmthRateManager, 628
WasCast
FishingRod, 208
WasherBaseLength
WindowWasher, 659
WaterContent
PlantCategory, 449
WaterDistrictItemRarityInfo
ItemFactory, 332
WaterItemGeneration, 629
AddBuildingToWaterGenerationMap, 631
GenerateInWater, 631
SetCityInformation, 631
WaterItems
ItemDistrictModel, 327
WaterItemsByDistrict
ItemFactory, 332
WaterLevelHeight
FloodWater, 220
Game, 235
waterLevelOffset
FishAgentConfig, 202
WaterMovement, 632
Climb, 634
GetClimbHeight, 634
GetRaycastHeight, 635
Idle, 635
Jump, 635
Move, 636
OnStateEnter, 636
OnStateExit, 636
SplashManager, 637
SwimmingHeight, 637
WaterPointGenerator, 638
AddBuildingBounds, 641
DistrictCheck, 642
Districts, 642
generateInitialPoints, 641
GetPointsInWater, 641
MaxAttempts, 643
MaxNumberOfItems, 643
NewPointsPerSamplingPoint, 643
NumberOfInitialPoints, 643
PointToGrid, 642
WaterPointGenerator, 640
waterTag
SamplingPointGenerator, 565
Weather
WeatherSystem.cs, 705
Weather_Clicked
TestGUI, 602
WeatherDatabase
AnnouncementFactory, 22
WeatherInformation
WeatherSystem, 648
WeatherInstance
Game, 235
WeatherPressureSystems
WeatherSystem, 649
WeatherSoundSystem, 644
DefaultWeatherEventPath, 645
WeatherSystem, 646
DisableWeather, 647
EnableWeather, 647
NormalizedOctantWindDirection2d, 648
ToString, 647
UpdateSystem, 648
UpdateWeather, 648
WeatherInformation, 648
WeatherPressureSystems, 649
WeatherSystem, 647
WindDirection2d, 649
WindDirection3d, 649
WeatherSystem.cs
Weather, 705
WeatherUpdated
EventManager, 175
WeatherUpdatedDelegate
EventManager, 175
WeatherUpdatedSubscription
EventManager, 176
WeaveBasket
SolidCategory, 585
WeaveRope
SolidCategory, 585
WeenieBuildingTemplate
DistrictConfiguration, 142
WeightedPair, 650
First, 651
Second, 651
Threshold, 651
WeightedPair, 650
WhaleSpawner, 652
findSpawnLocation, 653
Width
   ProceduralBuildingWindow, 504
WindDirection2d
   WeatherSystem, 649
WindDirection3d
   WeatherSystem, 649
WindMovement, 654
WindVectorXZ
   DebrisController, 133
   RainController, 527
WindowPoints
   ProceduralBuildingAttachment, 489
   ProceduralBuildingBase, 493
WindowSpacing
   ProceduralBuildingInstance, 501
WindowWasher, 655
   MaxLowerDistance, 658
   PlacementCenter, 658
   Start, 657
   StartUp, 658
   TeleportPlayer, 657
   Update, 658
   UpdateRopePosition, 658
   WasherBaseLength, 659
WindowWasherChance
   DistrictConfiguration, 145
WindowWasherMaxLengthCurve
   DistrictConfiguration, 145
WindowWasherStartUpChance
   DistrictConfiguration, 146
Windows
   ProceduralBuildingInstance, 501
WorldItemFactory, 659
   CreateGenericInteractableItem, 660
   CreatePickUpInteractableItem, 661
   CreateRandomPickupInteractableItem, 661
   GetAllInteractableItemsByDistrict, 661
   GetRandomItemIndex, 662
   LoadTemplates, 662
   WorldItemFactoryInstance
WorldModel
   BaseItem, 43
   NoteData, 427
WorldPosition
   Player, 454
WorldSelectionButtonBehavior, 663
   ButtonName, 664
   SetAction, 664
WorldSelectionGUIDirector, 665
   ClearOptions, 666
   CloseSelection, 666
   DisplayActions, 667
   DisplayItemOptions, 667
   FinishSelection, 667
WorldSpaceLocation
   ItemPlacementSamplePoint, 340
WorldTransform
   Player, 454
X
   Tuple, 614
XZ
   VectorUtility, 619
Y
   Tuple, 614
Zoom
   CameraController, 70
ZoomLevel
   CameraController, 71