VoxelGenerator

Generated by Doxygen 1.10.0

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

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

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

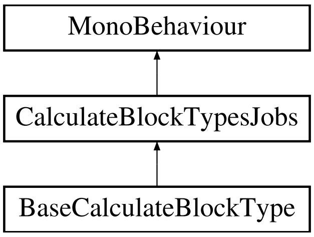
Chapter 4

Class Documentation

4.1 BaseCalculateBlockType Class Reference

Calculates block types in a chunk.

Inheritance diagram for BaseCalculateBlockType:



Public Member Functions

- override void **AssignValues** (NativeArray< **MeshUtils.BlockType** > chunkData, int width, int height, Vector3 location, NativeArray< **Random** > randoms)

  Assigns values for block type calculation.

Additional Inherited Members

Public Attributes inherited from CalculateBlockTypesJobs

- **CalculateBlockTypes generationJob**

4.1.1 Detailed Description

Calculates block types in a chunk.

4.1.2 Member Function Documentation

4.1.2.1 AssignValues()

```
override void BaseCalculateBlockType.AssignValues (
            NativeArray< MeshUtils::BlockType > chunkData,
            int width,
            int height,
            Vector3 location,
            NativeArray< Random > randoms ) [virtual]
```

Assigns values for block type calculation.

Parameters

chunkData	The native array containing block types for the chunk.
width	The width of the chunk.
height	The height of the chunk.
location	The location of the chunk.
randoms	The native array of random numbers.

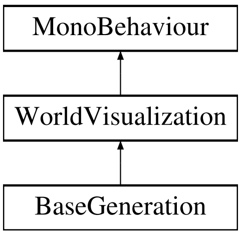
Reimplemented from **CalculateBlockTypesJobs** (p. 14).

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

- Assets/scripts/Experiment2/Scripts/ **BaseCalculateBlockType.cs**

4.2 BaseGeneration Class Reference

Inheritance diagram for BaseGeneration:



Additional Inherited Members

Public Member Functions inherited from WorldVisualization

- void **CreateSettings** ()

  Creates Perlin noise settings based on PerlinGraphers and ***Perlin3DGrapher*** *(p. 34).*

Public Attributes inherited from WorldVisualization

- **CalculateBlockTypesJobs calculate**
- List< **PerlinGrapher** > **perlinGraphers** = new List< **PerlinGrapher**>()
- **Perlin3DGrapher perlinGrapher3D**
- List< **PerlinSettings** > **perlinSettings** = new List< **PerlinSettings**>()

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

- Assets/scripts/Experiment2/Scripts/ **BaseGeneration.cs**

4.3 Block Class Reference

Represents a block in a chunk.

Public Member Functions

- **Block** (Vector3 offset, **MeshUtils.BlockType** type, **ChunkBlock** chunk)
  Constructor for creating a block.
- bool **HasSolidNeighbour** (int x, int y, int z)
  Checks if there is a solid neighbor at the specified position.

Public Attributes

- Mesh **mesh**
  The mesh associated with the block.

Private Attributes

- **ChunkBlock parentChunk**
  Reference to the parent chunk.

4.3.1 Detailed Description

Represents a block in a chunk.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 Block()

```
Block.Block (
             Vector3 offset,
              MeshUtils::BlockType type,
              ChunkBlock chunk )
```

Constructor for creating a block.

Parameters

offset	The offset of the block.
type	The type of the block.
chunk	The parent chunk of the block.

4.3.3 Member Function Documentation

4.3.3.1 HasSolidNeighbour()

```
bool Block.HasSolidNeighbour (
             int x,
             int y,
             int z )
```

Checks if there is a solid neighbor at the specified position.

Parameters

x	The x-coordinate.
y	The y-coordinate.
z	The z-coordinate.

Returns

True if there is a solid neighbor; false otherwise.

4.3.4 Member Data Documentation

4.3.4.1 mesh

```
Mesh Block.mesh
```

The mesh associated with the block.

4.3.4.2 parentChunk

```
ChunkBlock Block.parentChunk  [private]
```

Reference to the parent chunk.

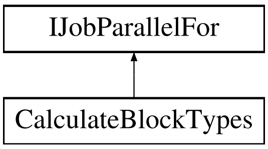
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **Block.cs**

4.4 CalculateBlockTypes Struct Reference

Job structure for parallel block generation.

Inheritance diagram for CalculateBlockTypes:



Public Member Functions

- void **Execute** (int i)

  Executes the block generation job in parallel for each block.
- void **BaseGenerator** (int i)

  Generates blocks based on base terrain settings.
- void **IslandGenerator** (int i)

  Generates blocks for an island terrain.
- void **FlatGenerator** (int i)

  Generates blocks for a flat terrain.

Public Attributes

- NativeArray< **MeshUtils.BlockType** > **chunkData**
- int **width**
- int **height**
- Vector3 **location**
- NativeArray< Unity.Mathematics.Random > **randoms**
- int **function**

4.4.1 Detailed Description

Job structure for parallel block generation.

4.4.2 Member Function Documentation

4.4.2.1 BaseGenerator()

```
void CalculateBlockTypes.BaseGenerator (
            int i )
```

Generates blocks based on base terrain settings.

Parameters

i	Index of the block.

4.4.2.2 Execute()

```
void CalculateBlockTypes.Execute (
             int i )
```

Executes the block generation job in parallel for each block.

Parameters

i	Index of the block.

4.4.2.3 FlatGenerator()

```
void CalculateBlockTypes.FlatGenerator (
             int i )
```

Generates blocks for a flat terrain.

Parameters

i	Index of the block.

4.4.2.4 IslandGenerator()

```
void CalculateBlockTypes.IslandGenerator (
             int i )
```

Generates blocks for an island terrain.

Parameters

i	Index of the block.

4.4.3 Member Data Documentation

4.4.3.1 chunkData

```
NativeArray< MeshUtils.BlockType> CalculateBlockTypes.chunkData
```

Native array to store block types.

4.4.3.2 function

```
int CalculateBlockTypes.function
```

Function identifier for block generation.

4.4.3.3 height

```
int CalculateBlockTypes.height
```

Height of the chunk.

4.4.3.4 location

```
Vector3 CalculateBlockTypes.location
```

Location of the chunk in the world.

4.4.3.5 randoms

```
NativeArray<Unity.Mathematics.Random> CalculateBlockTypes.randoms
```

Array of random numbers for block generation.

4.4.3.6 width

```
int CalculateBlockTypes.width
```

Width of the chunk.

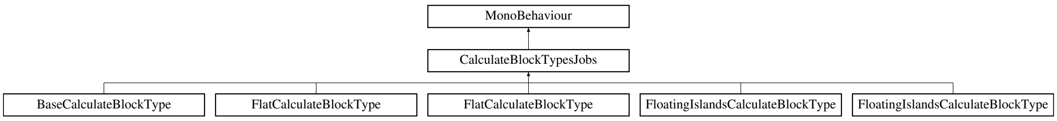
The documentation for this struct was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **CalculateBlockTypesJobs.cs**

4.5 CalculateBlockTypesJobs Class Reference

Manages the block generation job and assigns values to the generation job instance.

Inheritance diagram for CalculateBlockTypesJobs:



Public Member Functions

- virtual void **AssignValues** (NativeArray< **MeshUtils.BlockType** > chunkData, int width, int height, Vector3 location, NativeArray< **Random** > randoms)

  Assigns values to the generation job instance.

Public Attributes

- **CalculateBlockTypes generationJob**

4.5.1 Detailed Description

Manages the block generation job and assigns values to the generation job instance.

4.5.2 Member Function Documentation

4.5.2.1 AssignValues()

```
virtual void CalculateBlockTypesJobs.AssignValues (
             NativeArray< MeshUtils::BlockType > chunkData,
             int width,
             int height,
             Vector3 location,
             NativeArray< Random > randoms )  [virtual]
```

Assigns values to the generation job instance.

Parameters

chunkData	Native array to store block types.
width	Width of the chunk.
height	Height of the chunk.
location	Location of the chunk in the world.
randoms	Array of random numbers for block generation.

Reimplemented in **BaseCalculateBlockType** (p. 8), **FlatCalculateBlockType** (p. 26), and **FloatingIslands↩ CalculateBlockType** (p. 27).

4.5.3 Member Data Documentation

4.5.3.1 generationJob

```
CalculateBlockTypes CalculateBlockTypesJobs.generationJob
```

Instance of the **CalculateBlockTypes** (p. 11) class for block generation.

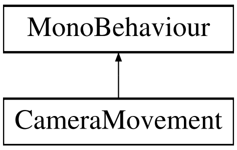
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **CalculateBlockTypesJobs.cs**

4.6 CameraMovement Class Reference

Manages the movement and rotation of the camera.

Inheritance diagram for CameraMovement:



Public Attributes

- float **movementSpeed** = 10f
  The movement speed of the camera.

Private Member Functions

- void **Update** ()
  Updates the camera's position based on user input.

4.6.1 Detailed Description

Manages the movement and rotation of the camera.

4.6.2 Member Function Documentation

4.6.2.1 Update()

```
void CameraMovement.Update ( )  [private]
```

Updates the camera's position based on user input.

4.6.3 Member Data Documentation

4.6.3.1 movementSpeed

```
float CameraMovement.movementSpeed = 10f
```

The movement speed of the camera.

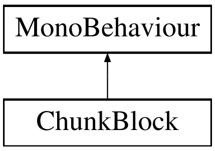
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **CameraMovement.cs**

4.7 ChunkBlock Class Reference

Manages the generation and rendering of chunks in the world.

Inheritance diagram for ChunkBlock:



Classes

- struct **ProcessMeshDataJob**

  Job for processing mesh data in parallel.

Public Member Functions

- void **CreateChunk** (Vector3 dimension, Vector3 position, bool rebuildBlocks=true)

  Creates a chunk with the specified dimensions and position.

Public Attributes

- Material **atlas**
- int **width** = 2
- int **height** = 2
- int **depth** = 2
- **Block**[„] **blocks**
- **MeshUtils.BlockType**[ ] **cData**
- MeshRenderer **meshRenderer**
- Vector3 **location**

Properties

- NativeArray< Unity.Mathematics.Random > **RandomArray** `[get, private set]`

Private Member Functions

- void **BuildChunk** ()

  Builds the chunk by generating block types and initializing block instances.

Private Attributes

- **CalculateBlockTypes** **calculateBlockTypes**
- **CalculateBlockTypesJobs** **calculateBlockTypesJobs**
- JobHandle **handle**

4.7.1 Detailed Description

Manages the generation and rendering of chunks in the world.

4.7.2 Member Function Documentation

4.7.2.1 BuildChunk()

```
void ChunkBlock.BuildChunk ( )  [private]
```

Builds the chunk by generating block types and initializing block instances.

4.7.2.2 CreateChunk()

```
void ChunkBlock.CreateChunk (
             Vector3 dimension,
             Vector3 position,
             bool rebuildBlocks = true )
```

Creates a chunk with the specified dimensions and position.

Parameters

dimension	The dimensions of the chunk.
position	The position of the chunk in the world.
rebuildBlocks	Whether to rebuild blocks for the chunk.

4.7.3 Member Data Documentation

4.7.3.1 atlas

```
Material ChunkBlock.atlas
```

Material for the chunk's blocks.

4.7.3.2 blocks

```
Block [„] ChunkBlock.blocks
```

3D array to store individual blocks in the chunk.

4.7.3.3 calculateBlockTypes

```
CalculateBlockTypes ChunkBlock.calculateBlockTypes  [private]
```

Instance of the **CalculateBlockTypes** (p. 11) class.

4.7.3.4 calculateBlockTypesJobs

```
CalculateBlockTypesJobs ChunkBlock.calculateBlockTypesJobs  [private]
```

Instance of the **CalculateBlockTypesJobs** (p. 13) class.

4.7.3.5 cData

```
MeshUtils.BlockType [] ChunkBlock.cData
```

Array to store block types for each position in the chunk.

4.7.3.6 depth

```
int ChunkBlock.depth = 2
```

Depth of the chunk.

4.7.3.7 handle

```
JobHandle ChunkBlock.handle  [private]
```

Job handle for parallel jobs.

4.7.3.8 height

```
int ChunkBlock.height = 2
```

Height of the chunk.

4.7.3.9 location

```
Vector3 ChunkBlock.location
```

Location of the chunk in the world.

4.7.3.10 meshRenderer

```
MeshRenderer ChunkBlock.meshRenderer
```

Mesh renderer for the chunk.

4.7.3.11 width

```
int ChunkBlock.width = 2
```

Width of the chunk.

4.7.4 Property Documentation

4.7.4.1 RandomArray

```
NativeArray<Unity.Mathematics.Random> ChunkBlock.RandomArray  [get], [private set]
```

Array of random numbers for block generation.

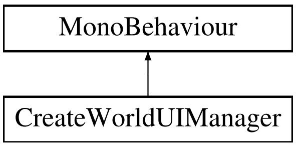
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **ChunkBlock.cs**

4.8 CreateWorldUIManager Class Reference

Manages UI elements related to world creation.

Inheritance diagram for CreateWorldUIManager:



Public Member Functions

- void **ChooseMenu** ()

    Switches between different UI menus.
- void **ChangeHideTerrain** ()

    Toggles the visibility of terrain in the world.
- void **CheckGraphs** ()

    Toggles visibility of terrain graphs.
- void **ChangeColor** ()

    Changes the color of the cave toggle button.
- void **GenerateWorld** ()

    Generates a new world based on the selected parameters.

Public Attributes

- List< GameObject > **avaibleMenus**
- TMP_Dropdown **menuDropdown**
- Slider **worldDimensionSlider**
- Slider **chunkDimensionSlider**
- Slider **renderDistanceSlider**
- Button **createWorld**
- Button **showGraphs**
- Toggle **caveToggle**
- Toggle **hideWorld**
- List< GameObject > **toHide**
- int **currentIndexOfDropdown** = 0
- bool **useCave** = false
- bool **hideTerrain** = false
- **WorldVisualization chosenWorldVisualization**

Private Member Functions

- void **Start** ()
  Initializes the UI manager.

4.8.1 Detailed Description

Manages UI elements related to world creation.

4.8.2 Member Function Documentation

4.8.2.1 ChangeColor()

```
void CreateWorldUIManager.ChangeColor ( )
```

Changes the color of the cave toggle button.

4.8.2.2 ChangeHideTerrain()

```
void CreateWorldUIManager.ChangeHideTerrain ( )
```

Toggles the visibility of terrain in the world.

4.8.2.3 CheckGraphs()

```
void CreateWorldUIManager.CheckGraphs ( )
```

Toggles visibility of terrain graphs.

4.8.2.4 ChooseMenu()

```
void CreateWorldUIManager.ChooseMenu ( )
```

Switches between different UI menus.

4.8.2.5 GenerateWorld()

```
void CreateWorldUIManager.GenerateWorld ( )
```

Generates a new world based on the selected parameters.

4.8.2.6 Start()

```
void CreateWorldUIManager.Start ( ) [private]
```

Initializes the UI manager.

4.8.3 Member Data Documentation

4.8.3.1 avaibleMenus

```
List<GameObject> CreateWorldUIManager.avaibleMenus
```

4.8.3.2 caveToggle

```
Toggle CreateWorldUIManager.caveToggle
```

4.8.3.3 chosenWorldVisualization

```
WorldVisualization CreateWorldUIManager.chosenWorldVisualization
```

4.8.3.4 chunkDimensionSlider

```
Slider CreateWorldUIManager.chunkDimensionSlider
```

4.8.3.5 createWorld

```
Button CreateWorldUIManager.createWorld
```

4.8.3.6 currentIndexOfDropdown

```
int CreateWorldUIManager.currentIndexOfDropdown = 0
```

4.8.3.7 hideTerrain

```
bool CreateWorldUIManager.hideTerrain = false
```

4.8.3.8 hideWorld

```
Toggle CreateWorldUIManager.hideWorld
```

4.8.3.9 menuDropdown

```
TMP_Dropdown CreateWorldUIManager.menuDropdown
```

4.8.3.10 renderDistanceSlider

```
Slider CreateWorldUIManager.renderDistanceSlider
```

4.8.3.11 showGraphs

```
Button CreateWorldUIManager.showGraphs
```

4.8.3.12 toHide

```
List<GameObject> CreateWorldUIManager.toHide
```

4.8.3.13 useCave

```
bool CreateWorldUIManager.useCave = false
```

4.8.3.14 worldDimensionSlider

```
Slider CreateWorldUIManager.worldDimensionSlider
```

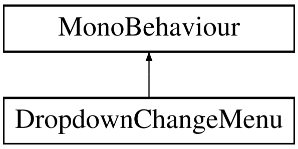
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/UI/ **CreateWorldUIManager.cs**

4.9 DropdownChangeMenu Class Reference

Manages the activation and deactivation of menus based on the selection in a dropdown.

Inheritance diagram for DropdownChangeMenu:



Public Attributes

- List< GameObject > **listToChange**

  The list of menus to change based on dropdown selection.
- TMP_Dropdown **dropdown**

  The TMP_Dropdown component responsible for menu selection.

Private Member Functions

- void **Start** ()

  Initializes the event listener for dropdown value changes.
- void **ChangeMenu** ()

  Changes the active menu based on the selection in the dropdown.

Private Attributes

- int **activeMenu** = 0
  The index of the currently active menu.

4.9.1 Detailed Description

Manages the activation and deactivation of menus based on the selection in a dropdown.

4.9.2 Member Function Documentation

4.9.2.1 ChangeMenu()

```
void DropdownChangeMenu.ChangeMenu ( )  [private]
```

Changes the active menu based on the selection in the dropdown.

4.9.2.2 Start()

```
void DropdownChangeMenu.Start ( )  [private]
```

Initializes the event listener for dropdown value changes.

4.9.3 Member Data Documentation

4.9.3.1 activeMenu

```
int DropdownChangeMenu.activeMenu = 0  [private]
```

The index of the currently active menu.

4.9.3.2 dropdown

```
TMP_Dropdown DropdownChangeMenu.dropdown
```

The TMP_Dropdown component responsible for menu selection.

4.9.3.3 listToChange

```
List<GameObject> DropdownChangeMenu.listToChange
```

The list of menus to change based on dropdown selection.

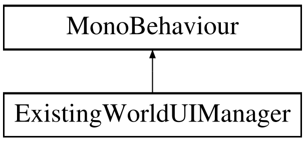
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/UI/ **DropdownChangeMenu.cs**

4.10 ExistingWorldUIManager Class Reference

Manages the user interface for selecting and loading existing worlds in Unity.

Inheritance diagram for ExistingWorldUIManager:



Private Member Functions

- void **Start** ()

  Initializes the UI elements and event listeners.
- void **StartLoading** ()

  Initiates the process of loading the selected existing world.

Static Private Member Functions

- static string **GetFolder** ()

  Gets the folder path for saving world data.

Private Attributes

- TMP_Dropdown **existingWorlds**
- Slider **renderSlider**
- Button **loadWorld**
- List< string > **allWorlds**

4.10.1 Detailed Description

Manages the user interface for selecting and loading existing worlds in Unity.

4.10.2 Member Function Documentation

4.10.2.1 GetFolder()

```
static string ExistingWorldUIManager.GetFolder ( )  [static], [private]
```

Gets the folder path for saving world data.

Returns

The folder path for saving world data.

4.10.2.2 Start()

```
void ExistingWorldUIManager.Start ( )  [private]
```

Initializes the UI elements and event listeners.

4.10.2.3 StartLoading()

```
void ExistingWorldUIManager.StartLoading ( )  [private]
```

Initiates the process of loading the selected existing world.

4.10.3 Member Data Documentation

4.10.3.1 allWorlds

```
List<string> ExistingWorldUIManager.allWorlds  [private]
```

4.10.3.2 existingWorlds

```
TMP_Dropdown ExistingWorldUIManager.existingWorlds  [private]
```

4.10.3.3 loadWorld

```
Button ExistingWorldUIManager.loadWorld  [private]
```

4.10.3.4 renderSlider

```
Slider ExistingWorldUIManager.renderSlider  [private]
```

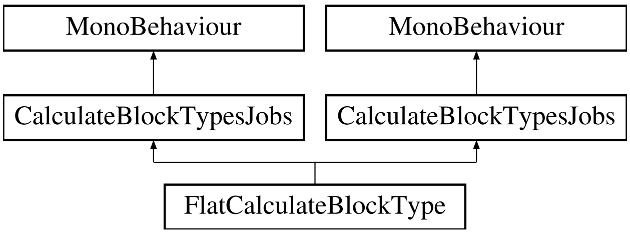
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/UI/ **ExistingWorldUIManager.cs**

4.11 FlatCalculateBlockType Class Reference

Manages the calculation of block types for flat terrain in a Unity environment.

Inheritance diagram for FlatCalculateBlockType:



Public Member Functions

- override void **AssignValues** (NativeArray< **MeshUtils.BlockType** > chunkData, int width, int height, Vector3 location, NativeArray< **Random** > randoms)

  Assigns values to the generation job for calculating block types.

Additional Inherited Members

Public Attributes inherited from CalculateBlockTypesJobs

- **CalculateBlockTypes generationJob**

4.11.1 Detailed Description

Manages the calculation of block types for flat terrain in a Unity environment.

4.11.2 Member Function Documentation

4.11.2.1 AssignValues()

```
override void FlatCalculateBlockType.AssignValues (
            NativeArray< MeshUtils::BlockType > chunkData,
            int width,
            int height,
            Vector3 location,
            NativeArray< Random > randoms ) [virtual]
```

Assigns values to the generation job for calculating block types.

Parameters

chunkData	NativeArray of **MeshUtils.BlockType** (p. 32) representing the chunk's block data.
width	Width of the chunk.
height	Height of the chunk.
location	World position of the chunk.
randoms	NativeArray of Unity.Mathematics.Random for generating random values.

Reimplemented from **CalculateBlockTypesJobs** (p. 14).

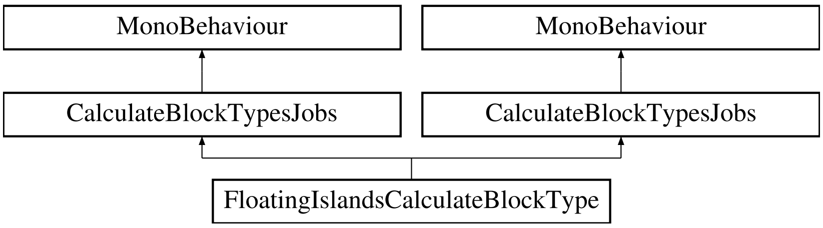
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **FlatCalculateBlockType.cs**

4.12 FloatingIslandsCalculateBlockType Class Reference

Manages the calculation of block types for floating islands in a Unity environment.

Inheritance diagram for FloatingIslandsCalculateBlockType:



Public Member Functions

- override void **AssignValues** (NativeArray< **MeshUtils.BlockType** > chunkData, int width, int height, Vector3 location, NativeArray< **Random** > randoms)

  Assigns values to the generation job for calculating block types.

Additional Inherited Members

Public Attributes inherited from CalculateBlockTypesJobs

- **CalculateBlockTypes generationJob**

4.12.1 Detailed Description

Manages the calculation of block types for floating islands in a Unity environment.

4.12.2 Member Function Documentation

4.12.2.1 AssignValues()

```
override void FloatingIslandsCalculateBlockType.AssignValues (
            NativeArray< MeshUtils::BlockType > chunkData,
            int width,
            int height,
            Vector3 location,
            NativeArray< Random > randoms ) [virtual]
```

Assigns values to the generation job for calculating block types.

Parameters

chunkData	NativeArray of **MeshUtils.BlockType** (p. 32) representing the chunk's block data.
width	Width of the chunk.
height	Height of the chunk.
location	World position of the chunk.
randoms	NativeArray of Unity.Mathematics.Random for generating random values.

Reimplemented from **CalculateBlockTypesJobs** (p. 14).

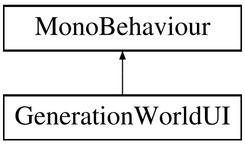
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **FloatingIslandsCalculateBlockType.cs**

4.13 GenerationWorldUI Class Reference

Manages the UI elements related to world generation.

Inheritance diagram for GenerationWorldUI:



Public Attributes

- **WorldVisualization worldVisualization**
  *Reference to the **WorldVisualization** (p. 64) component used for world generation.*

4.13.1 Detailed Description

Manages the UI elements related to world generation.

4.13.2 Member Data Documentation

4.13.2.1 worldVisualization

```
GenerationWorldUI.worldVisualization
```

Reference to the **WorldVisualization** (p. 64) component used for world generation.

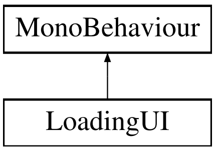
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/UI/ **GenerationWorldUI.cs**

4.14 LoadingUI Class Reference

Manages loading UI elements, such as a loading bar.

Inheritance diagram for LoadingUI:



Public Member Functions

- void **SetMaxValue** (int size)

  Sets the maximum value of the loading bar.
- void **UpdateValue** ()

  Updates the value of the loading bar.
- void **CloseLoading** ()

  Closes the loading UI and switches to the hub UI.

Public Attributes

- Slider **loadingBar**

  Reference to the loading bar UI element.

Properties

- static **LoadingUI** **instance** `[get]`

  Gets the singleton instance of the ***LoadingUI*** *(p. 28) class.*

Private Member Functions

- void **Awake** ()

  Called when the script instance is being loaded.

Static Private Attributes

- static **LoadingUI** **_instance**

4.14.1 Detailed Description

Manages loading UI elements, such as a loading bar.

4.14.2 Member Function Documentation

4.14.2.1 Awake()

```
void LoadingUI.Awake ( )  [private]
```

Called when the script instance is being loaded.

4.14.2.2 CloseLoading()

```
void LoadingUI.CloseLoading ( )
```

Closes the loading UI and switches to the hub UI.

4.14.2.3 SetMaxValue()

```
void LoadingUI.SetMaxValue (
             int size )
```

Sets the maximum value of the loading bar.

Parameters

size	The maximum value for the loading bar.

4.14.2.4 UpdateValue()

```
void LoadingUI.UpdateValue ( )
```

Updates the value of the loading bar.

4.14.3 Member Data Documentation

4.14.3.1 _instance

```
LoadingUI LoadingUI._instance  [static], [private]
```

4.14.3.2 loadingBar

```
LoadingUI.loadingBar
```

Reference to the loading bar UI element.

4.14.4 Property Documentation

4.14.4.1 instance

```
LoadingUI.instance  [static], [get]
```

Gets the singleton instance of the **LoadingUI** (p. 28) class.

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

- Assets/scripts/Experiment2/Scripts/UI/ **LoadingUI.cs**

4.15 MeshUtils Class Reference

Provides utility functions for working with meshes in Unity.

Public Types

- enum **BlockType** {
  GRASSTOP , **GRASSSIDE** , **DIRT** , **WATER** ,
  STONE , **SAND** , **GOLD** , **BEDROCK** ,
  REDSTONE , **DIAMOND** , **NOCRACK** , **CRACK1** ,
  CRACK2 , **CRACK3** , **CRACK4** , **AIR** }

  Represents different types of blocks.
- enum **BlockSide** {
  BOTTOM , **TOP** , **LEFT** , **RIGHT** ,
  FRONT , **BACK** }

  Represents different sides of a block.

Static Public Member Functions

- static float **fBM** (float x, float z, int octaves, float scale, float heightScale, float heightOffset)

  Calculates 1D Perlin noise at a given position.
- static float **fBM3D** (float x, float y, float z, int octaves, float scale, float heightScale, float heightOffset)

  Calculates 3D Perlin noise at a given position.
- static Mesh **MergeMeshes** (Mesh[ ] meshes)

  Merges multiple meshes into a single mesh.
- static void **ExtractArrays** (Dictionary< **VertexData**, int > list, Mesh mesh)

  Extracts vertex arrays from a dictionary and assigns them to a mesh.

Static Public Attributes

- static Vector2[,] **blockUVs**

  Defines UV coordinates for different block types and sides.

4.15.1 Detailed Description

Provides utility functions for working with meshes in Unity.

4.15.2 Member Enumeration Documentation

4.15.2.1 BlockSide

`enum MeshUtils.BlockSide`

Represents different sides of a block.

Enumerator

BOTTOM	
TOP	
LEFT	
RIGHT	
FRONT	
BACK	

4.15.2.2 BlockType

```
enum MeshUtils.BlockType
```

Represents different types of blocks.

Enumerator

GRASSTOP	
GRASSSIDE	
DIRT	
WATER	
STONE	
SAND	
GOLD	
BEDROCK	
REDSTONE	
DIAMOND	
NOCRACK	
CRACK1	
CRACK2	
CRACK3	
CRACK4	
AIR	

4.15.3 Member Function Documentation

4.15.3.1 ExtractArrays()

```
static void MeshUtils.ExtractArrays (
             Dictionary< VertexData, int > list,
             Mesh mesh ) [static]
```

Extracts vertex arrays from a dictionary and assigns them to a mesh.

Parameters

list	A dictionary containing vertex data and their corresponding indices.
mesh	The mesh to assign the vertex arrays.

4.15.3.2 fBM()

```
static float MeshUtils.fBM (
             float x,
             float z,
             int octaves,
             float scale,
             float heightScale,
             float heightOffset ) [static]
```

Calculates 1D Perlin noise at a given position.

Parameters

x	The x-coordinate.
z	The z-coordinate.
octaves	The number of octaves in the Perlin noise.
scale	The scale factor.
heightScale	The height scale factor.
heightOffset	The height offset.

Returns

The calculated Perlin noise value.

4.15.3.3 fBM3D()

```
static float MeshUtils.fBM3D (
             float x,
             float y,
             float z,
             int octaves,
             float scale,
             float heightScale,
             float heightOffset )  [static]
```

Calculates 3D Perlin noise at a given position.

Parameters

x	The x-coordinate.
y	The y-coordinate.
z	The z-coordinate.
octaves	The number of octaves in the Perlin noise.
scale	The scale factor.
heightScale	The height scale factor.
heightOffset	The height offset.

Returns

The calculated 3D Perlin noise value.

4.15.3.4 MergeMeshes()

```
static Mesh MeshUtils.MergeMeshes (
             Mesh[ ] meshes )  [static]
```

Merges multiple meshes into a single mesh.

Parameters

meshes	An array of meshes to merge.

Returns

The merged mesh.

4.15.4 Member Data Documentation

4.15.4.1 blockUVs

```
MeshUtils.blockUVs  [static]
```

Defines UV coordinates for different block types and sides.

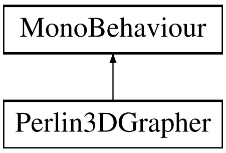
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **MeshUtils.cs**

4.16 Perlin3DGrapher Class Reference

Graphs 3D Perlin noise in a 3D grid of cubes in Unity.

Inheritance diagram for Perlin3DGrapher:



Public Member Functions

- void **Graph** ()
  Graphs the 3D Perlin noise in the grid of cubes.

Public Attributes

- float **heightScale** = 2
- float **scale** = 0.5f
- int **octaves** = 1
- float **heightOffset** = 1
- float **drawCutOff** = 1

Private Member Functions

- void **CreateCubes** ()

  Creates a grid of cubes in the scene.
- void **OnValidate** ()

  Called when the script is loaded or a value is changed in the Inspector.

Private Attributes

- Vector3 **dimensions** = new Vector3(10, 10, 10)

4.16.1 Detailed Description

Graphs 3D Perlin noise in a 3D grid of cubes in Unity.

This class is marked with [ExecuteInEditMode] to allow for graphing in the Unity Editor.

4.16.2 Member Function Documentation

4.16.2.1 CreateCubes()

```
void Perlin3DGrapher.CreateCubes ( )  [private]
```

Creates a grid of cubes in the scene.

4.16.2.2 Graph()

```
void Perlin3DGrapher.Graph ( )
```

Graphs the 3D Perlin noise in the grid of cubes.

4.16.2.3 OnValidate()

```
void Perlin3DGrapher.OnValidate ( )  [private]
```

Called when the script is loaded or a value is changed in the Inspector.

4.16.3 Member Data Documentation

4.16.3.1 dimensions

```
Vector3 Perlin3DGrapher.dimensions = new Vector3(10, 10, 10)  [private]
```

Dimensions of the 3D grid.

4.16.3.2 drawCutOff

```
float Perlin3DGrapher.drawCutOff = 1
```

Cutoff value for drawing cubes based on the Perlin noise.

4.16.3.3 heightOffset

```
float Perlin3DGrapher.heightOffset = 1
```

Offset applied to the height values in the Perlin noise.

4.16.3.4 heightScale

```
float Perlin3DGrapher.heightScale = 2
```

Height scale factor applied to the Perlin noise.

4.16.3.5 octaves

```
int Perlin3DGrapher.octaves = 1
```

Number of octaves used in the Perlin noise calculation.

4.16.3.6 scale

```
float Perlin3DGrapher.scale = 0.5f
```

Scale factor influencing the scale of the Perlin noise.

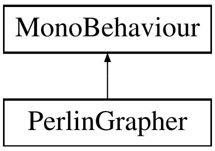
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **Perlin3DGrapher.cs**

4.17 PerlinGrapher Class Reference

Graphs 2D Perlin noise using a LineRenderer in Unity.

Inheritance diagram for PerlinGrapher:



Public Member Functions

- void **StartSetup** ()

  Initializes the LineRenderer and sets up initial parameters.
- void **Graph** ()

  Graphs the 2D Perlin noise using the LineRenderer.

Public Attributes

- LineRenderer **lr**
- float **heightScale** = 2
- float **scale** = 0.5f
- int **octaves** = 1
- float **heightOffset** = 1
- float **probability** = 1
- Material **lineMaterial**

Private Member Functions

- void **Start** ()

  Called when the script is loaded or a value is changed in the Inspector.
- void **OnValidate** ()

  Called when the script is loaded or a value is changed in the Inspector.

Private Attributes

- bool **isInitComplete** = false

4.17.1 Detailed Description

Graphs 2D Perlin noise using a LineRenderer in Unity.

This class is marked with [ExecuteInEditMode] to allow for graphing in the Unity Editor.

4.17.2 Member Function Documentation

4.17.2.1 Graph()

```
void PerlinGrapher.Graph ( )
```

Graphs the 2D Perlin noise using the LineRenderer.

4.17.2.2 OnValidate()

```
void PerlinGrapher.OnValidate ( )  [private]
```

Called when the script is loaded or a value is changed in the Inspector.

4.17.2.3 Start()

```
void PerlinGrapher.Start ( )  [private]
```

Called when the script is loaded or a value is changed in the Inspector.

4.17.2.4 StartSetup()

```
void PerlinGrapher.StartSetup ( )
```

Initializes the LineRenderer and sets up initial parameters.

4.17.3 Member Data Documentation

4.17.3.1 heightOffset

```
float PerlinGrapher.heightOffset = 1
```

Offset applied to the height values in the Perlin noise.

4.17.3.2 heightScale

```
float PerlinGrapher.heightScale = 2
```

Height scale factor applied to the Perlin noise.

4.17.3.3 isInitComplete

```
bool PerlinGrapher.isInitComplete = false  [private]
```

Flag indicating whether the initialization is complete.

4.17.3.4 lineMaterial

```
Material PerlinGrapher.lineMaterial
```

Material used for the LineRenderer.

4.17.3.5 lr

```
LineRenderer PerlinGrapher.lr
```

LineRenderer component for drawing the graph.

4.17.3.6 octaves

```
int PerlinGrapher.octaves = 1
```

Number of octaves used in the Perlin noise calculation.

4.17.3.7 probability

```
float PerlinGrapher.probability = 1
```

Probability factor influencing the graph appearance.

4.17.3.8 scale

```
float PerlinGrapher.scale = 0.5f
```

Scale factor influencing the scale of the Perlin noise.

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

- Assets/scripts/Experiment2/Scripts/ **PerlinGrapher.cs**

4.18 PerlinSettings Struct Reference

Represents settings for Perlin noise generation.

Public Member Functions

- **PerlinSettings** (float hs, float s, int o, float ho, float p)
  Initializes Perlin noise settings.

Public Attributes

- float **heightScale**
- float **scale**
- int **octaves**
- float **heightOffset**
- float **probability**

4.18.1 Detailed Description

Represents settings for Perlin noise generation.

4.18.2 Constructor & Destructor Documentation

4.18.2.1 PerlinSettings()

```
PerlinSettings.PerlinSettings (
             float hs,
             float s,
             int o,
             float ho,
             float p )
```

Initializes Perlin noise settings.

Parameters

hs	Height scale.
s	Scale.
o	Octaves.
ho	Height offset.
p	Probability.

4.18.3 Member Data Documentation

4.18.3.1 heightOffset

```
float PerlinSettings.heightOffset
```

4.18.3.2 heightScale

```
float PerlinSettings.heightScale
```

4.18.3.3 octaves

```
int PerlinSettings.octaves
```

4.18.3.4 probability

```
float PerlinSettings.probability
```

4.18.3.5 scale

```
float PerlinSettings.scale
```

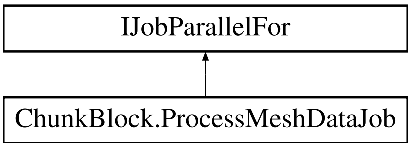
The documentation for this struct was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **WorldVisualization.cs**

4.19 ChunkBlock.ProcessMeshDataJob Struct Reference

Job for processing mesh data in parallel.

Inheritance diagram for ChunkBlock.ProcessMeshDataJob:



Public Member Functions

- void **Execute** (int index)

    Executes the job in parallel for each mesh.

Public Attributes

- Mesh.MeshDataArray **meshData**
- Mesh.MeshData **outputMesh**
- NativeArray< int > **vertexStart**
- NativeArray< int > **triStart**

4.19.1 Detailed Description

Job for processing mesh data in parallel.

4.19.2 Member Function Documentation

4.19.2.1 Execute()

```
void ChunkBlock.ProcessMeshDataJob.Execute (
             int index )
```

Executes the job in parallel for each mesh.

Parameters

index	The index of the mesh data to process.

4.19.3 Member Data Documentation

4.19.3.1 meshData

```
Mesh.MeshDataArray ChunkBlock.ProcessMeshDataJob.meshData
```

Read-only array of mesh data.

4.19.3.2 outputMesh

```
Mesh.MeshData ChunkBlock.ProcessMeshDataJob.outputMesh
```

Output mesh data.

4.19.3.3 triStart

```
NativeArray<int> ChunkBlock.ProcessMeshDataJob.triStart
```

Array to store the start index of each triangle in the output mesh.

4.19.3.4 vertexStart

```
NativeArray<int> ChunkBlock.ProcessMeshDataJob.vertexStart
```

Array to store the start index of each vertex in the output mesh.

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

- Assets/scripts/Experiment2/Scripts/ **ChunkBlock.cs**

4.20 Quad Class Reference

Represents a quad mesh in Unity for a specific block side and type.

Public Member Functions

- **Quad** (**MeshUtils.BlockSide** side, Vector3 offset, **MeshUtils.BlockType** bType)
  Constructor for creating a quad mesh.

Public Attributes

- Mesh **mesh**

4.20.1 Detailed Description

Represents a quad mesh in Unity for a specific block side and type.

4.20.2 Constructor & Destructor Documentation

4.20.2.1 Quad()

```
Quad.Quad (
              MeshUtils::BlockSide side,
             Vector3 offset,
              MeshUtils::BlockType bType )
```

Constructor for creating a quad mesh.

Parameters

side	The block side for which the quad is created.
offset	The offset position of the quad.
bType	The block type determining the UV coordinates.

4.20.3 Member Data Documentation

4.20.3.1 mesh

```
Mesh Quad.mesh
```

The mesh representing the quad.

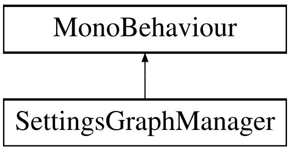
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **Quad.cs**

4.21 SettingsGraphManager Class Reference

Manages the UI and interactions for adjusting Perlin noise graph settings in Unity.

Inheritance diagram for SettingsGraphManager:



Public Member Functions

- void **HeightScaleValueChangeCheck** ()

  Handles the change in the height scale input field value.
- void **ScaleValueChangeCheck** ()

  Handles the change in the graph scale slider value.
- void **OctavesValueChangeCheck** ()

  Handles the change in the octaves input field value.
- void **ProbabilityValueChangeCheck** ()

  Handles the change in the probability slider value.
- void **HeightOffsetValueChangeCheck** ()

  Handles the change in the height offset input field value.

Public Attributes

- **PerlinGrapher perlinGrapherSettings**
- **Perlin3DGrapher perlin3DGrapherSettings**

Private Member Functions

- void **Start** ()

    Initializes the UI elements with default or saved settings when the script starts.

Private Attributes

- TMP_InputField **heightScaleInputField**
- Slider **scaleSlider**
- TMP_InputField **octavesInputField**
- Slider **probabilitySlider**
- TMP_InputField **heightOffsetInputField**
- Image **background**

4.21.1 Detailed Description

Manages the UI and interactions for adjusting Perlin noise graph settings in Unity.

4.21.2 Member Function Documentation

4.21.2.1 HeightOffsetValueChangeCheck()

```
void SettingsGraphManager.HeightOffsetValueChangeCheck ( )
```

Handles the change in the height offset input field value.

4.21.2.2 HeightScaleValueChangeCheck()

```
void SettingsGraphManager.HeightScaleValueChangeCheck ( )
```

Handles the change in the height scale input field value.

4.21.2.3 OctavesValueChangeCheck()

```
void SettingsGraphManager.OctavesValueChangeCheck ( )
```

Handles the change in the octaves input field value.

4.21.2.4 ProbabilityValueChangeCheck()

```
void SettingsGraphManager.ProbabilityValueChangeCheck ( )
```

Handles the change in the probability slider value.

4.21.2.5 ScaleValueChangeCheck()

```
void SettingsGraphManager.ScaleValueChangeCheck ( )
```

Handles the change in the graph scale slider value.

4.21.2.6 Start()

```
void SettingsGraphManager.Start ( )  [private]
```

Initializes the UI elements with default or saved settings when the script starts.

4.21.3 Member Data Documentation

4.21.3.1 background

```
Image SettingsGraphManager.background  [private]
```

Background image of the UI.

4.21.3.2 heightOffsetInputField

```
TMP_InputField SettingsGraphManager.heightOffsetInputField  [private]
```

Input field for adjusting the height offset.

4.21.3.3 heightScaleInputField

```
TMP_InputField SettingsGraphManager.heightScaleInputField  [private]
```

Input field for adjusting the height scale.

4.21.3.4 octavesInputField

```
TMP_InputField SettingsGraphManager.octavesInputField  [private]
```

Input field for adjusting the number of octaves.

4.21.3.5 perlin3DGrapherSettings

```
Perlin3DGrapher SettingsGraphManager.perlin3DGrapherSettings
```

Reference to the **Perlin3DGrapher** (p. 34) for 3D graph settings.

4.21.3.6 perlinGrapherSettings

```
PerlinGrapher SettingsGraphManager.perlinGrapherSettings
```

Reference to the **PerlinGrapher** (p. 36) for 2D graph settings.

4.21.3.7 probabilitySlider

```
Slider SettingsGraphManager.probabilitySlider  [private]
```

Slider for adjusting the probability value.

4.21.3.8 scaleSlider

```
Slider SettingsGraphManager.scaleSlider  [private]
```

Slider for adjusting the graph scale.

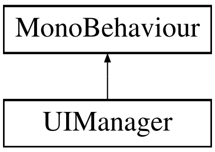
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/UI/ **SettingsGraphManager.cs**

4.22 UIManager Class Reference

Manages UI elements in the game.

Inheritance diagram for UIManager:



Public Member Functions

- void **ChangeToLoading** ()

  Changes UI elements to display loading state.
- void **ChangeToHub** ()

  Changes UI elements to display the game hub state.
- void **CloseApp** ()

  Closes the application.

Public Attributes

- GameObject **settings**
- GameObject **visualization**
- GameObject **loading**
- GameObject **gameHub**

Properties

- static **UIManager** **instance** `[get]`

Private Member Functions

- void **Awake** ()

  Method called when the script instance is being loaded. Ensures only one instance of ***UIManager*** *(p. 46) exists.*

Static Private Attributes

- static **UIManager** **_instance**

4.22.1 Detailed Description

Manages UI elements in the game.

4.22.2 Member Function Documentation

4.22.2.1 Awake()

```
void UIManager.Awake ( )  [private]
```

Method called when the script instance is being loaded. Ensures only one instance of **UIManager** (p. 46) exists.

4.22.2.2 ChangeToHub()

```
void UIManager.ChangeToHub ( )
```

Changes UI elements to display the game hub state.

4.22.2.3 ChangeToLoading()

```
void UIManager.ChangeToLoading ( )
```

Changes UI elements to display loading state.

4.22.2.4 CloseApp()

```
void UIManager.CloseApp ( )
```

Closes the application.

4.22.3 Member Data Documentation

4.22.3.1 _instance

```
UIManager UIManager._instance [static], [private]
```

Instance of the **UIManager** (p. 46) to allow for easy access from other scripts.

4.22.3.2 gameHub

```
GameObject UIManager.gameHub
```

Reference to the game hub UI GameObject.

4.22.3.3 loading

```
GameObject UIManager.loading
```

Reference to the loading UI GameObject.

4.22.3.4 settings

```
GameObject UIManager.settings
```

Reference to the settings UI GameObject.

4.22.3.5 visualization

```
GameObject UIManager.visualization
```

Reference to the visualization UI GameObject.

4.22.4 Property Documentation

4.22.4.1 instance

```
UIManager UIManager.instance [static], [get]
```

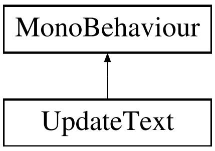
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/UI/ **UIManager.cs**

4.23 UpdateText Class Reference

Updates the text based on the Slider value.

Inheritance diagram for UpdateText:



Public Member Functions

- void **UpdateTextFrom** ()

    Updates the text based on the Slider value with rounding to 3 decimal places.
- void **UpdateTextFromInt** ()

    Updates the text based on the integer value of the Slider.

Public Attributes

- TMP_Text **text**
- Slider **slider**

Private Member Functions

- void **Start** ()

    Method called during the object's start. Updates the text based on the Slider value.

4.23.1 Detailed Description

Updates the text based on the Slider value.

4.23.2 Member Function Documentation

4.23.2.1 Start()

```
void UpdateText.Start ( )  [private]
```

Method called during the object's start. Updates the text based on the Slider value.

4.23.2.2 UpdateTextFrom()

```
void UpdateText.UpdateTextFrom ( )
```

Updates the text based on the Slider value with rounding to 3 decimal places.

4.23.2.3 UpdateTextFromInt()

```
void UpdateText.UpdateTextFromInt ( )
```

Updates the text based on the integer value of the Slider.

4.23.3 Member Data Documentation

4.23.3.1 slider

```
Slider UpdateText.slider
```

Reference to the Slider object whose value affects the text.

4.23.3.2 text

```
TMP_Text UpdateText.text
```

Reference to the TMP_Text object to be updated.

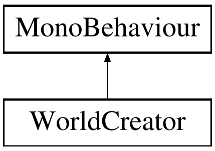
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/UI/ **UpdateText.cs**

4.24 WorldCreator Class Reference

Generates and manages the game world in Unity.

Inheritance diagram for WorldCreator:



Public Member Functions

- void **SaveWorld** ()

  Saves the current state of the world.
- void **StartWorld** (**WorldVisualization** chosenWorldVisualization, Vector3Int dataVector, bool useCaves↩Choose, bool isHideTerrain)

  Initializes the world based on the provided parameters.
- void **StartBuilding** (bool fromFile, string fileName="")

  Initiates the building process of the world.
- void **HideChunkColumn** (int x, int z)

  Hides a chunk column at the specified coordinates.

Public Attributes

- GameObject **chunkPrefab**
- GameObject **fpc**
- GameObject **mCamera**
- int **drawRadius** = 5
- HashSet< Vector3Int > **chunkChecker** = new HashSet<Vector3Int>()
- HashSet< Vector2Int > **chunkColumns** = new HashSet<Vector2Int>()
- Dictionary< Vector3Int, **ChunkBlock** > **chunks** = new Dictionary<Vector3Int, **ChunkBlock**>()
- bool **load** = true

Static Public Attributes

- static Vector3Int **worldDimensions** = new Vector3Int(3, 4, 3)
- static Vector3Int **extraWorldDimensions** = new Vector3Int(1, 4, 1)
- static Vector3Int **chunkDimensions** = new Vector3Int(10, 10, 10)
- static **WorldVisualization** **worldVisualization**
- static bool **useCaves**
- static bool **hideTerrain**

Properties

- static **WorldCreator** **instance** `[get]`

Private Member Functions

- void **Awake** ()

  *Method called when the script instance is being loaded. Ensures only one instance of **WorldCreator** (p. 50) exists.*
- IEnumerator **BuildCoordinator** ()

  Coroutine to manage the build tasks.
- IEnumerator **BuildChunkColumn** (int x, int z, bool meshEnable=true)

  Coroutine to build a column of chunks at the specified coordinates.
- IEnumerator **HideCollums** (int x, int z)

  Coroutine to hide columns of chunks based on the player's position.
- IEnumerator **BuildExtraWorld** ()

  Coroutine to build the extra world for optimization.
- IEnumerator **BuildWorld** ()

  Coroutine to build the main world.
- IEnumerator **LoadWorldFromFile** (string fileName)

  Coroutine to load the world from a file.
- void **RedrawChunk** (**ChunkBlock** c)

  Redraws the mesh for a given chunk.
- IEnumerator **BuildRecursiveWorld** (int x, int z, int rad)

  Coroutine to build chunks recursively based on the player's position.
- IEnumerator **UpdateWorld** ()

  Coroutine to continuously update the world based on the player's position.

Private Attributes

- Vector3Int **lastBuildPosition**
- Queue< IEnumerator > **buildQueue** = new Queue<IEnumerator>()
- WaitForSeconds **waitForSeconds** = new WaitForSeconds(0.1f)

Static Private Attributes

- static **WorldCreator** **_instance**

4.24.1 Detailed Description

Generates and manages the game world in Unity.

4.24.2 Member Function Documentation

4.24.2.1 Awake()

```
void WorldCreator.Awake ( )  [private]
```

Method called when the script instance is being loaded. Ensures only one instance of **WorldCreator** (p. 50) exists.

4.24.2.2 BuildChunkColumn()

```
IEnumerator WorldCreator.BuildChunkColumn (
             int x,
             int z,
             bool meshEnable = true )  [private]
```

Coroutine to build a column of chunks at the specified coordinates.

Parameters

x	X-coordinate of the chunk column.
z	Z-coordinate of the chunk column.
meshEnable	Flag indicating whether the mesh should be enabled for the chunks.

4.24.2.3 BuildCoordinator()

```
IEnumerator WorldCreator.BuildCoordinator ( )  [private]
```

Coroutine to manage the build tasks.

4.24.2.4 BuildExtraWorld()

```
IEnumerator WorldCreator.BuildExtraWorld ( )  [private]
```

Coroutine to build the extra world for optimization.

4.24.2.5 BuildRecursiveWorld()

```
IEnumerator WorldCreator.BuildRecursiveWorld (
             int x,
             int z,
             int rad )  [private]
```

Coroutine to build chunks recursively based on the player's position.

Parameters

x	X-coordinate of the player's position.
z	Z-coordinate of the player's position.
rad	Current radius for chunk generation.

4.24.2.6 BuildWorld()

```
IEnumerator WorldCreator.BuildWorld ( )  [private]
```

Coroutine to build the main world.

4.24.2.7 HideChunkColumn()

```
void WorldCreator.HideChunkColumn (
             int x,
             int z )
```

Hides a chunk column at the specified coordinates.

Parameters

x	X-coordinate of the chunk column.
z	Z-coordinate of the chunk column.

4.24.2.8 HideCollums()

```
IEnumerator WorldCreator.HideCollums (
             int x,
             int z )  [private]
```

Coroutine to hide columns of chunks based on the player's position.

Parameters

x	X-coordinate of the player's position.
z	Z-coordinate of the player's position.

4.24.2.9 LoadWorldFromFile()

```
IEnumerator WorldCreator.LoadWorldFromFile (
             string fileName )  [private]
```

Coroutine to load the world from a file.

Parameters

fileName	Name of the file to load.

4.24.2.10 RedrawChunk()

```
void WorldCreator.RedrawChunk (
              ChunkBlock c )  [private]
```

Redraws the mesh for a given chunk.

Parameters

c	The **ChunkBlock** (p. 16) to redraw.

4.24.2.11 SaveWorld()

```
void WorldCreator.SaveWorld ( )
```

Saves the current state of the world.

4.24.2.12 StartBuilding()

```
void WorldCreator.StartBuilding (
             bool fromFile,
             string fileName = "" )
```

Initiates the building process of the world.

Parameters

fromFile	Flag indicating whether to load the world from a file.
fileName	Name of the file to load if loading from a file.

4.24.2.13 StartWorld()

```
void WorldCreator.StartWorld (
              WorldVisualization chosenWorldVisualization,
             Vector3Int dataVector,
             bool useCavesChoose,
             bool isHideTerrain )
```

Initializes the world based on the provided parameters.

Parameters

chosenWorldVisualization	The selected model for terrain generation.
dataVector	The vector containing world dimensions, chunk dimensions, and draw radius.
useCavesChoose	Flag indicating whether caves should be used.
isHideTerrain	Flag indicating whether terrain should be hidden.

4.24.2.14 UpdateWorld()

```
IEnumerator WorldCreator.UpdateWorld ( )  [private]
```

Coroutine to continuously update the world based on the player's position.

4.24.3 Member Data Documentation

4.24.3.1 _instance

```
 WorldCreator WorldCreator._instance  [static], [private]
```

Instance of the **WorldCreator** (p. 50) to allow for easy access from other scripts.

4.24.3.2 buildQueue

```
Queue<IEnumerator> WorldCreator.buildQueue = new Queue<IEnumerator>()  [private]
```

Queue for managing coroutine-based build tasks.

4.24.3.3 chunkChecker

```
HashSet<Vector3Int> WorldCreator.chunkChecker = new HashSet<Vector3Int>()
```

HashSet to store positions of generated chunks.

4.24.3.4 chunkColumns

```
HashSet<Vector2Int> WorldCreator.chunkColumns = new HashSet<Vector2Int>()
```

HashSet to store column positions of generated chunks.

4.24.3.5 chunkDimensions

```
Vector3Int WorldCreator.chunkDimensions = new Vector3Int(10, 10, 10) [static]
```

Dimensions of each chunk in the world.

4.24.3.6 chunkPrefab

```
GameObject WorldCreator.chunkPrefab
```

Prefab for the chunk GameObject.

4.24.3.7 chunks

```
Dictionary<Vector3Int, ChunkBlock> WorldCreator.chunks = new Dictionary<Vector3Int, Chunk←
Block>()
```

Dictionary to store **ChunkBlock** (p. 16) instances based on their positions.

4.24.3.8 drawRadius

```
int WorldCreator.drawRadius = 5
```

Radius for drawing chunks around the player.

4.24.3.9 extraWorldDimensions

```
Vector3Int WorldCreator.extraWorldDimensions = new Vector3Int(1, 4, 1) [static]
```

Dimensions of the extra world for optimization.

4.24.3.10 fpc

```
GameObject WorldCreator.fpc
```

First-person character GameObject.

4.24.3.11 hideTerrain

```
bool WorldCreator.hideTerrain [static]
```

Flag to determine if terrain should be hidden.

4.24.3.12 lastBuildPosition

```
Vector3Int WorldCreator.lastBuildPosition  [private]
```

Last built position for optimizing chunk generation.

4.24.3.13 load

```
bool WorldCreator.load = true
```

Flag to determine if the world should be loaded from a file.

4.24.3.14 mCamera

```
GameObject WorldCreator.mCamera
```

Main camera GameObject.

4.24.3.15 useCaves

```
bool WorldCreator.useCaves  [static]
```

Flag to determine if caves should be used in the world.

4.24.3.16 waitForSeconds

```
WaitForSeconds WorldCreator.waitForSeconds = new WaitForSeconds(0.1f)  [private]
```

WaitForSeconds instance for controlling update intervals.

4.24.3.17 worldDimensions

```
Vector3Int WorldCreator.worldDimensions = new Vector3Int(3, 4, 3)  [static]
```

Dimensions of the main world.

4.24.3.18 worldVisualization

```
WorldVisualization WorldCreator.worldVisualization  [static]
```

Instance of the **WorldVisualization** (p. 64) used for creating the world.

4.24.4 Property Documentation

4.24.4.1 instance

```
WorldCreator WorldCreator.instance  [static], [get]
```

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

- Assets/scripts/Experiment2/Scripts/ **WorldCreator.cs**

4.25 WorldData Class Reference

Represents the data structure for saving and loading the game world.

Public Member Functions

- **WorldData** ()
- **WorldData** (HashSet< Vector3Int > cc, HashSet< Vector2Int > cCols, Dictionary< Vector3Int, **ChunkBlock** > chunks, Vector3 fpc, Vector3Int worldDimension, Vector3Int chunkDimension, **WorldVisualization** worldVisualization)

  *Constructor to initialize **WorldData** (p. 58) with relevant information from the game world.*

Public Attributes

- int[ ] **chunkCheckerValues**
- int[ ] **chunkColumnsValues**
- int[ ] **chunkData**
- bool[ ] **chunkVisibility**
- int **fpcX**
- int **fpcY**
- int **fpcZ**
- int[ ] **worldDimensions**
- int[ ] **chunkDimensions**
- **PerlinSettings**[ ] **perlinSettings**
- **CalculateBlockTypes** **calculateBlockTypes**
- bool **hideTerrain**
- bool **useCave**

4.25.1 Detailed Description

Represents the data structure for saving and loading the game world.

4.25.2 Constructor & Destructor Documentation

4.25.2.1 WorldData() [1/2]

```
WorldData.WorldData ( )
```

Default constructor.

4.25.2.2 WorldData() [2/2]

```
WorldData.WorldData (
            HashSet< Vector3Int > cc,
            HashSet< Vector2Int > cCols,
            Dictionary< Vector3Int, ChunkBlock > chunks,
            Vector3 fpc,
            Vector3Int worldDimension,
            Vector3Int chunkDimension,
             WorldVisualization worldVisualization )
```

Constructor to initialize **WorldData** (p. 58) with relevant information from the game world.

Parameters

cc	Set of chunk checker positions.
cCols	Set of chunk column positions.
chunks	Dictionary containing **ChunkBlock** (p. 16) instances.
fpc	Position of the first-person character.
worldDimension	Dimensions of the game world.
chunkDimension	Dimensions of each chunk.
worldVisualization	**WorldVisualization** (p. 64) instance for Perlin noise settings.

4.25.3 Member Data Documentation

4.25.3.1 calculateBlockTypes

```
CalculateBlockTypes WorldData.calculateBlockTypes
```

CalculateBlockTypes (p. 11) instance for generating block types in the world.

4.25.3.2 chunkCheckerValues

```
int [ ] WorldData.chunkCheckerValues
```

Array to store values of chunk checker positions.

4.25.3.3 chunkColumnsValues

```
int [ ] WorldData.chunkColumnsValues
```

Array to store values of chunk column positions.

4.25.3.4 chunkData

```
int [ ] WorldData.chunkData
```

Array to store block data for each chunk.

4.25.3.5 chunkDimensions

```
int [ ] WorldData.chunkDimensions
```

Array to store the dimensions of each chunk.

4.25.3.6 chunkVisibility

```
bool [ ] WorldData.chunkVisibility
```

Array to store visibility status for each chunk.

4.25.3.7 fpcX

```
int WorldData.fpcX
```

X-coordinate of the first-person character.

4.25.3.8 fpcY

```
int WorldData.fpcY
```

Y-coordinate of the first-person character.

4.25.3.9 fpcZ

```
int WorldData.fpcZ
```

Z-coordinate of the first-person character.

4.25.3.10 hideTerrain

```
bool WorldData.hideTerrain
```

Flag indicating whether terrain should be hidden.

4.25.3.11 perlinSettings

```
PerlinSettings [ ] WorldData.perlinSettings
```

Array to store Perlin noise settings for world generation.

4.25.3.12 useCave

```
bool WorldData.useCave
```

Flag indicating whether caves should be used in world generation.

4.25.3.13 worldDimensions

```
int [] WorldData.worldDimensions
```

Array to store the dimensions of the game world.

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

- Assets/scripts/Experiment2/Scripts/ **WorldSaver.cs**

4.26 WorldSaver Class Reference

Provides methods for saving and loading the game world in Unity.

Static Public Member Functions

- static void **Save** (**WorldCreator** worldCreator)
  Saves the current state of the game world.
- static **WorldData** **Load** (string fileLocation)
  Loads the game world from a specified file.

Static Public Attributes

- static List< string > **allFiles**

Static Private Member Functions

- static string **CreateBuildFileName** ()
  Creates a unique file name for saving the world.
- static string **LoadBuildFileName** (int index)
  Loads the file name at the specified index.

Static Private Attributes

- static **WorldData** **worldData**

4.26.1 Detailed Description

Provides methods for saving and loading the game world in Unity.

4.26.2 Member Function Documentation

4.26.2.1 CreateBuildFileName()

```
static string WorldSaver.CreateBuildFileName ( )  [static], [private]
```

Creates a unique file name for saving the world.

Returns

The generated file name.

4.26.2.2 Load()

```
static  WorldData WorldSaver.Load (
             string fileLocation )  [static]
```

Loads the game world from a specified file.

Parameters

fileLocation	The location of the file to load.

Returns

The loaded **WorldData** (p. 58) instance.

4.26.2.3 LoadBuildFileName()

```
static string WorldSaver.LoadBuildFileName (
             int index )  [static], [private]
```

Loads the file name at the specified index.

Parameters

index	Index of the file name in the list of available files.

Returns

The file name at the specified index.

4.26.2.4 Save()

```
static void WorldSaver.Save (
              WorldCreator worldCreator )  [static]
```

Saves the current state of the game world.

Parameters

worldCreator	Instance of **WorldCreator** (p. 50) representing the game world.

4.26.3 Member Data Documentation

4.26.3.1 allFiles

```
List<string> WorldSaver.allFiles  [static]
```

List of all available saved files.

4.26.3.2 worldData

```
WorldData WorldSaver.worldData  [static], [private]
```

Instance of **WorldData** (p. 58) for storing data during saving and loading.

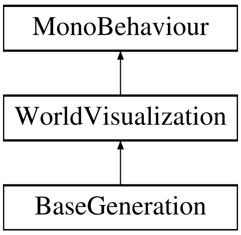
The documentation for this class was generated from the following file:

- Assets/scripts/Experiment2/Scripts/ **WorldSaver.cs**

4.27 WorldVisualization Class Reference

Visualizes the world using Perlin noise.

Inheritance diagram for WorldVisualization:



Public Member Functions

- void **CreateSettings** ()

  Creates Perlin noise settings based on PerlinGraphers and ***Perlin3DGrapher*** *(p. 34).*

Public Attributes

- **CalculateBlockTypesJobs calculate**
- List< **PerlinGrapher** > **perlinGraphers** = new List< **PerlinGrapher**>()
- **Perlin3DGrapher perlinGrapher3D**
- List< **PerlinSettings** > **perlinSettings** = new List< **PerlinSettings**>()

Private Member Functions

- void **Start** ()

  Initializes the world visualization.

4.27.1 Detailed Description

Visualizes the world using Perlin noise.

4.27.2 Member Function Documentation

4.27.2.1 CreateSettings()

```
void WorldVisualization.CreateSettings ( )
```

Creates Perlin noise settings based on PerlinGraphers and **Perlin3DGrapher** (p. 34).

4.27.2.2 Start()

```
void WorldVisualization.Start ( )  [private]
```

Initializes the world visualization.

4.27.3 Member Data Documentation

4.27.3.1 calculate

```
CalculateBlockTypesJobs WorldVisualization.calculate
```

4.27.3.2 perlinGrapher3D

```
Perlin3DGrapher WorldVisualization.perlinGrapher3D
```

4.27.3.3 perlinGraphers

```
List< PerlinGrapher> WorldVisualization.perlinGraphers = new List< PerlinGrapher>()
```

4.27.3.4 perlinSettings

```
List< PerlinSettings> WorldVisualization.perlinSettings = new List< PerlinSettings>()
```

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

- Assets/scripts/Experiment2/Scripts/ **WorldVisualization.cs**

Chapter 5

File Documentation

5.1 Assets/scripts/Experiment2/Scripts/BaseCalculateBlockType.cs File Reference

Defines a class for calculating block types in a chunk.

Classes

- class **BaseCalculateBlockType**
  Calculates block types in a chunk.

Typedefs

- using **Random** = Unity.Mathematics.Random

5.1.1 Detailed Description

Defines a class for calculating block types in a chunk.

5.1.2 Typedef Documentation

5.1.2.1 Random

```
using  Random = Unity.Mathematics.Random
```

5.2 Assets/scripts/Experiment2/Scripts/BaseGeneration.cs File Reference

Classes

- class **BaseGeneration**

5.3 Assets/scripts/Experiment2/Scripts/Block.cs File Reference

Defines a class for representing a block in a chunk.

Classes

- class **Block**

  Represents a block in a chunk.

5.3.1 Detailed Description

Defines a class for representing a block in a chunk.

5.4 Assets/scripts/Experiment2/Scripts/CalculateBlockTypesJobs.cs File Reference

Defines the **CalculateBlockTypesJobs** (p. 13) and **CalculateBlockTypes** (p. 11) classes for block generation.

Classes

- class **CalculateBlockTypesJobs**

  Manages the block generation job and assigns values to the generation job instance.
- struct **CalculateBlockTypes**

  Job structure for parallel block generation.

Typedefs

- using **Random** = Unity.Mathematics.Random

5.4.1 Detailed Description

Defines the **CalculateBlockTypesJobs** (p. 13) and **CalculateBlockTypes** (p. 11) classes for block generation.

5.4.2 Typedef Documentation

5.4.2.1 Random

```
using  Random = Unity.Mathematics.Random
```

5.5 Assets/scripts/Experiment2/Scripts/CameraMovement.cs File Reference

Defines a class for handling camera movement in Unity.

Classes

- class **CameraMovement**

  Manages the movement and rotation of the camera.

5.5.1 Detailed Description

Defines a class for handling camera movement in Unity.

5.6 Assets/scripts/Experiment2/Scripts/ChunkBlock.cs File Reference

Defines the **ChunkBlock** (p. 16) class responsible for generating and managing chunks of blocks in the world.

Classes

- class **ChunkBlock**

  Manages the generation and rendering of chunks in the world.
- struct **ChunkBlock.ProcessMeshDataJob**

  Job for processing mesh data in parallel.

5.6.1 Detailed Description

Defines the **ChunkBlock** (p. 16) class responsible for generating and managing chunks of blocks in the world.

5.7 Assets/scripts/Experiment2/Scripts/FlatCalculateBlockType.cs File Reference

Defines a class for calculating block types in a flat terrain scenario in Unity.

Classes

- class **FlatCalculateBlockType**

  Manages the calculation of block types for flat terrain in a Unity environment.

Typedefs

- using **Random** = Unity.Mathematics.Random

5.7.1 Detailed Description

Defines a class for calculating block types in a flat terrain scenario in Unity.

5.7.2 Typedef Documentation

5.7.2.1 Random

```
using Random = Unity.Mathematics.Random
```

5.8 Assets/scripts/Experiment2/Scripts/FloatingIslandsCalculateBlock↩ Type.cs File Reference

Defines a class for calculating block types in a floating islands scenario in Unity.

Classes

- class **FloatingIslandsCalculateBlockType**

  Manages the calculation of block types for floating islands in a Unity environment.

Typedefs

- using **Random** = Unity.Mathematics.Random

5.8.1 Detailed Description

Defines a class for calculating block types in a floating islands scenario in Unity.

5.8.2 Typedef Documentation

5.8.2.1 Random

```
using Random = Unity.Mathematics.Random
```

5.9 Assets/scripts/Experiment2/Scripts/MeshUtils.cs File Reference

Defines utility functions for working with meshes in Unity.

Classes

- class **MeshUtils**

  Provides utility functions for working with meshes in Unity.

Typedefs

- using **VertexData** = System.Tuple<UnityEngine.Vector3, UnityEngine.Vector3, UnityEngine.Vector2>

5.9.1 Detailed Description

Defines utility functions for working with meshes in Unity.

5.9.2 Typedef Documentation

5.9.2.1 VertexData

```
using VertexData = System.Tuple<UnityEngine.Vector3, UnityEngine.Vector3, UnityEngine.←
Vector2>
```

5.10 Assets/scripts/Experiment2/Scripts/Perlin3DGrapher.cs File Reference

Defines the **Perlin3DGrapher** (p. 34) class for graphing 3D Perlin noise in Unity.

Classes

- class **Perlin3DGrapher**

  Graphs 3D Perlin noise in a 3D grid of cubes in Unity.

5.10.1 Detailed Description

Defines the **Perlin3DGrapher** (p. 34) class for graphing 3D Perlin noise in Unity.

5.11 Assets/scripts/Experiment2/Scripts/PerlinGrapher.cs File Reference

Defines the **PerlinGrapher** (p. 36) class for graphing 2D Perlin noise in Unity using a LineRenderer.

Classes

- class **PerlinGrapher**

  Graphs 2D Perlin noise using a LineRenderer in Unity.

5.11.1 Detailed Description

Defines the **PerlinGrapher** (p. 36) class for graphing 2D Perlin noise in Unity using a LineRenderer.

5.12 Assets/scripts/Experiment2/Scripts/Quad.cs File Reference

Defines the **Quad** (p. 42) class responsible for creating a quad mesh in Unity.

Classes

- class **Quad**

  Represents a quad mesh in Unity for a specific block side and type.

5.12.1 Detailed Description

Defines the **Quad** (p. 42) class responsible for creating a quad mesh in Unity.

5.13 Assets/scripts/Experiment2/Scripts/UI/CreateWorldUIManager.cs File Reference

Defines a class for managing UI elements related to world creation.

Classes

- class **CreateWorldUIManager**

  Manages UI elements related to world creation.

5.13.1 Detailed Description

Defines a class for managing UI elements related to world creation.

5.14 Assets/scripts/Experiment2/Scripts/UI/DropdownChangeMenu.cs File Reference

Defines a class for changing active menus based on the selection in a dropdown in Unity.

Classes

- class **DropdownChangeMenu**

  Manages the activation and deactivation of menus based on the selection in a dropdown.

5.14.1 Detailed Description

Defines a class for changing active menus based on the selection in a dropdown in Unity.

5.15 Assets/scripts/Experiment2/Scripts/UI/ExistingWorldUIManager.cs File Reference

Defines a class for managing the user interface of existing worlds in Unity.

Classes

- class **ExistingWorldUIManager**

  Manages the user interface for selecting and loading existing worlds in Unity.

5.15.1 Detailed Description

Defines a class for managing the user interface of existing worlds in Unity.

5.16 Assets/scripts/Experiment2/Scripts/UI/GenerationWorldUI.cs File Reference

Defines a class for managing the UI related to world generation in Unity.

Classes

- class **GenerationWorldUI**

  Manages the UI elements related to world generation.

5.16.1 Detailed Description

Defines a class for managing the UI related to world generation in Unity.

5.17 Assets/scripts/Experiment2/Scripts/UI/LoadingUI.cs File Reference

Defines a class for managing loading UI in Unity.

Classes

- class **LoadingUI**

  Manages loading UI elements, such as a loading bar.

5.17.1 Detailed Description

Defines a class for managing loading UI in Unity.

5.18 Assets/scripts/Experiment2/Scripts/UI/SettingsGraphManager.cs File Reference

Defines the **SettingsGraphManager** (p. 43) class responsible for managing Perlin noise graph settings in Unity.

Classes

- class **SettingsGraphManager**

  Manages the UI and interactions for adjusting Perlin noise graph settings in Unity.

5.18.1 Detailed Description

Defines the **SettingsGraphManager** (p. 43) class responsible for managing Perlin noise graph settings in Unity.

5.19 Assets/scripts/Experiment2/Scripts/UI/UIManager.cs File Reference

Defines the **UIManager** (p. 46) class responsible for managing UI elements in the game.

Classes

- class **UIManager**

  Manages UI elements in the game.

5.19.1 Detailed Description

Defines the **UIManager** (p. 46) class responsible for managing UI elements in the game.

5.20 Assets/scripts/Experiment2/Scripts/UI/UpdateText.cs File Reference

Defines the **UpdateText** (p. 49) class responsible for updating text based on the Slider value.

Classes

- class **UpdateText**

  Updates the text based on the Slider value.

5.20.1 Detailed Description

Defines the **UpdateText** (p. 49) class responsible for updating text based on the Slider value.

5.21 Assets/scripts/Experiment2/Scripts/WorldCreator.cs File Reference

Defines the **WorldCreator** (p. 50) class responsible for generating and managing the game world in Unity.

Classes

- class **WorldCreator**

  Generates and manages the game world in Unity.

5.21.1 Detailed Description

Defines the **WorldCreator** (p. 50) class responsible for generating and managing the game world in Unity.

5.22 Assets/scripts/Experiment2/Scripts/WorldSaver.cs File Reference

Defines the **WorldSaver** (p. 61) class for saving and loading the game world in Unity.

Classes

- class **WorldData**

  Represents the data structure for saving and loading the game world.
- class **WorldSaver**

  Provides methods for saving and loading the game world in Unity.

5.22.1 Detailed Description

Defines the **WorldSaver** (p. 61) class for saving and loading the game world in Unity.

5.23 Assets/scripts/Experiment2/Scripts/WorldVisualization.cs File Reference

Defines a class for visualizing the world using Perlin noise.

Classes

- class **WorldVisualization**

  Visualizes the world using Perlin noise.
- struct **PerlinSettings**

  Represents settings for Perlin noise generation.

5.23.1 Detailed Description

Defines a class for visualizing the world using Perlin noise.

Index