HexPlanet Pro

HEX PLANET USER GUIDE

Thank you for purchasing HexPlanet Pro! This user guide will walk you through the basic operation of the asset, as well as providing some tips for creating awesome planets with ease.

Inspector Guides

Hexsphere.cs

The main component of HexPlanet is the Hexsphere.cs script. It contains all of the logic for the generation of the planet geometry, as well as some functions for defining the generated tile attributes. The custom inspector is outlined here.

- bool GenerateOnPlay: If true, the planet will generate meshes in the Start function, meaning the geometry will appear as soon as you hit play in the unity editor.
- bool GenerateAsSingleMesh: If true, the planet will not generate individual Tile gameobjects and will instead create a single closed mesh. This will remove many of the functionality associated with individual tiles, such as pathfinding, extrusion and selection. If false, the planet will be generated as separate tile gameobjects which will be children of the planet.
- bool Invert: If true, the generated meshes of the planet will have their normals pointing inward towards the center of the planet. If false, the normals will point outward like a standard sphere.
- NavigationManager NavManager: A reference to the planet's navigation logic component. In the demo prefab, it is attached as a child of the planet but you can give it a reference to any NavigationManager component as long as multiple planets do not share the same one.
- Int DetailLevel: The number of subdivisions to apply to the planet. A level of 0 will create a dodecahedron with 12 pentagonal faces. Level 1 results in 42 faces. Level 2 gives 162 faces. Level 3 gives 642 faces. Level 4 gives 2562 faces. Level 5 gives 10242 faces. Keep in mind that the higher this value, the longer the generation algorithm takes and values above 5 tend to produce too many vertices for most computers to handle. Negative values are not acceptable. If youd like to increase the cap and try to generate a planet with more than 5 subdivisions, you can edit the conditional statement in the beginning of the BuildPlanet function in Hexsphere.cs.

- Material[] GroupMaterials: This array defines the materials that will be assigned to the tiles according to their group ID. Each tile has a group ID which a useful integer for grouping your tiles into similar categories, such as water, land, mountains etc. You can assign any tile's group ID through the Tile inspector to automatically set its material to the corresponding slot set in the Hexsphere inspector. The max group ID value is determined by the size of this material array.
- bool GenerateTileColliders: If true, then all Tiles will be given a collider when the planet is first generated and a dropdown menu will appear giving you the option to choose the collider type. If false, no colliders will be added to the Tile objects.
- Int PlanetID: A handy ID integer assigned to planets when there are multiple in a scene. Useful for differentiating and tracking the individual planets.
- GeneratePlanet Button: When clicked, generates the planet with all of the current settings. Will be disabled once the tiles have been generated.
- GenerateRandomRegions Button: Assigns a random group ID to each tile thus setting their material according to the GroupMaterials array.
- DeleteTiles Button: If the planet has been generated, clicking this button will destroy all of the tiles.
- PlanetScale Slider: A helpful slider to set the planet's scale in all three spatial axes at once. It also sets the planetScale variable in the planet which the tiles use to calculate scaled extrusion heights, so use this slider to scale the planet rather than directly editing the planet's transform scale.
- Save to Prefab input: This input box and button allow you to safely save your planet as a prefab. Enter the file path to where youd like to save your planet prefab, such as "Assets/HexPlanet/PlanetPrefabs/". Make sure to include the trailing slash. The prefab will be given the name of the planet gameobject, so if the planet's name is "ExamplePlanet" and your path is the example above, the overall asset path will be "Assets/HexPlanet/PlanetPrefabs/ExamplePlanet.prefab". To load a planet prefab, simply drag it into the scene like any other prefab.

Tile.cs

The Tile class represents an individual tile object and contains all of the logic for extrusion, tile selection and more.

- Hexsphere ParentPlanet: A reference to the planet that generated this tile. It is automatically set when generated.
- List<Tile> NeighborTiles: A list of the tiles that directly surround this tile. If the tile is a pentagon, there will be 5 neighbors and if a hexagon, 6 neighbors.

- bool Navigable: If true, then any pathfinding logic will allow navigation across this tile. If false, navigation will avoid these tiles.
- Int PathCost: The navigation pathfinding weight assigned to this tile. The higher the value, the more the navigation logic will attempt to avoid this tile. Tiles with lower path costs will be prioritized when finding paths. The value has an upper bound of 100 but you can change that value in the Tile class as it was only limited for the convenience of a slider and does not need to be limited.
- List<Gameobject> PlacedObjects: A list of all objects that have been placed on this tile via the PlaceObject function. Useful if you choose not to parent any of the placed objects but still need to keep track of them.
- Int GroupID: An integer representing which group this tile is a member of. Tile groups are useful for organizing tiles with common attributes or appearances and determines which material the tile will be assigned via the GroupMaterials array in the parent planet. The tiles material need not match the one associated to its group ID but setting the group ID via the inspector will automatically set the tiles material to the corresponding material in the planets array.
- Place Object interface: Drag any gameobject into the object field and then click the Place Object button to position the selected object onto the tile and align that object's local up vector with the tile's local up vector. If the object exists in the scene then it will be moved and rotated. If the object is a prefab it will be instantiated before placement. All placed objects will become children of the selected tile(s) and then added to that tile's PlacedObjects list.
- Extrusion Tools:
    - Set Absolute Extrusion: Enter any value (positive or negative) and then click the Extrude button to set that tiles absolute extrusion height relative to its starting height when generated (a value of 0).
    - Add/Subtract Height: Enter any value (positive or negative) and click the Extrude button to change that tile's extrusion height by the entered value. For example, if the current extrusion height is 0.75 and a value of -0.25 is entered, clicking Extrude would result in the tile being extruded inwards by 0.25 for an absolute extrusion height of 0.5.
- Delete All Placed Objects Button: This button will appear on any tile that has had an object placed on it. Clicking this button will destroy all of the placed objects on that tile and clear the PlacedObjects list.
- Selection Options: Tools to mass select tiles based on input parameters
    - Same Path Cost: Chooses all tiles that have the same navigation PathCost value as this tile.
    - Same Navigability: Chooses all tiles that have the same value for Navigable as this tile.

    - Same Group ID: Chooses all tiles that have the same GroupID as this tile.
    - Same Extrusion Height: Chooses all tiles that have the same absolute extrusion height as this tile.
    - Clicking the Select Tiles Button will select all of the tiles on the planet with the given options AND'ed together, meaning all tiles that satisfy all of the checked options listed above.
    - Clicking the SelectConnectedGroup button will select all of the tiles that have the same group ID as this tile which are reachable, i.e. all tiles sharing the group ID enclosed by tiles with different group IDs

PlanetTools.cs

The PlanetTools script and inspector add a set of features to the scene view similar to the Unity Terrain Editor. Attaching the PlanetTools component to a gameobject with the Hexsphere component gives access to these features. When the component is expanded and the Shift key is held down, a circle will appear under the mouse cursor and moving this cursor over a planet with its tiles generated will highlight any tiles within the circle. Holding down the Control key (while shift is held) and scrolling the mouse wheel, the size of the selection circle can be changed to select more or less tiles. Clicking while the Shift key is held will perform the selected action on any highlighted tiles.

The available tools are outlined below.

- Paint Group ID:
    - Allows you to set the group ID for all tiles under the cursor when clicked.
    - The planet's GroupMaterials array determines how many GroupID buttons appear in the inspector.
- Paint Height:
    - Allows you to set the extrusion height of all tiles under the cursor when clicked with the following options:
    - Set Height: Uses the SetAbsoluteHeight function on the tiles to set absolute height for any selected tiles to the input amount.
    - Add Height: Uses the Add/SubtractHeight function on tiles to change the current extrusion height by the input amount.
- Paint Object:
    - Allows you to mass place prefabs onto the tiles under the cursor when clicked. Supports two modes.

    - Add Object Mode: Instantiates and places the selected prefab onto the tiles.
    - Remove Object Mode: Destroys the last instantiated object on the selected tiles.
    - Prefab List: Click the (+) button to add a prefab slot. Select a prefab for that slot by either dragging it into the object field or selecting it from Assets. Click the (x) button to delete the selected prefab slot. Click the Select button to choose that prefab as the one to place onto tiles when painting
- Paint Nav Weight:
    - Allows you to set the navigation PathCost for tiles when clicked. Supports three modes.
    - Add: Adds the selected nav weight to the selected tiles.
    - Subtract: Subtracts the selected nav weight from the selected tiles
    - Set: Sets the absolute nav weight for the selected tiles.

Heres an outline of the various advanced features and functions and how they work:

Navigation Manager Class

public bool findPath(Tile start, Tile end, out Stack<Tile> pathStack)

Finds the shortest path between the tiles start and end and returns true if this path exists.

Feed it a Stack<Tile> when calling the function and if a path was found, the input stack will now be filled with all of the tiles that make up the path with the top of the stack the starting tile, and the bottom of the stack being the destination tile.

public List<Tile> DFS(Tile start)

Performs a Depth First Search on the input tile "start" and returns a list of all tiles that are connected to start by navigable paths.

Any tiles with navigable marked true can be isolated into a closed regions of tiles if non navigable tiles surround that region.

public void drawPath(Stack<Tile> pathStack)

Uses the LineRenderer attached to the default navigationManager object in the prefabs to draw the input path represented by the stack of tiles.

This is useful for previewing unit movements or testing your maps navigation properties.

NOTE that the input stack will be consumed when this function terminates.

MobileUnit Class

public void moveOnPath(Stack<Tile> path)

Simply calls the coroutine which performs the actual movement

public IEnumerator move(Stack<Tile> path)

Moves the unit along the input path at its defined moveSpeed by popping tiles from the input path stack and spherically interpolating between its current tile position and the next tiles position.

Updates its currentTile variable every time it reaches the end of the Slerp. This is useful for triggering events when units move over certain tiles.

NOTES:

-The supplied unit template prefab has a specific transform hierarchy where the visible character mesh/model is placed on a transform as a child of the root transform holding the unit script itself.

-This is necessary as the root transform should be flush with the faces of the tiles, meaning that the units visible model should be positioned higher in the local Y axis so its feet or base are touching the root transforms center.