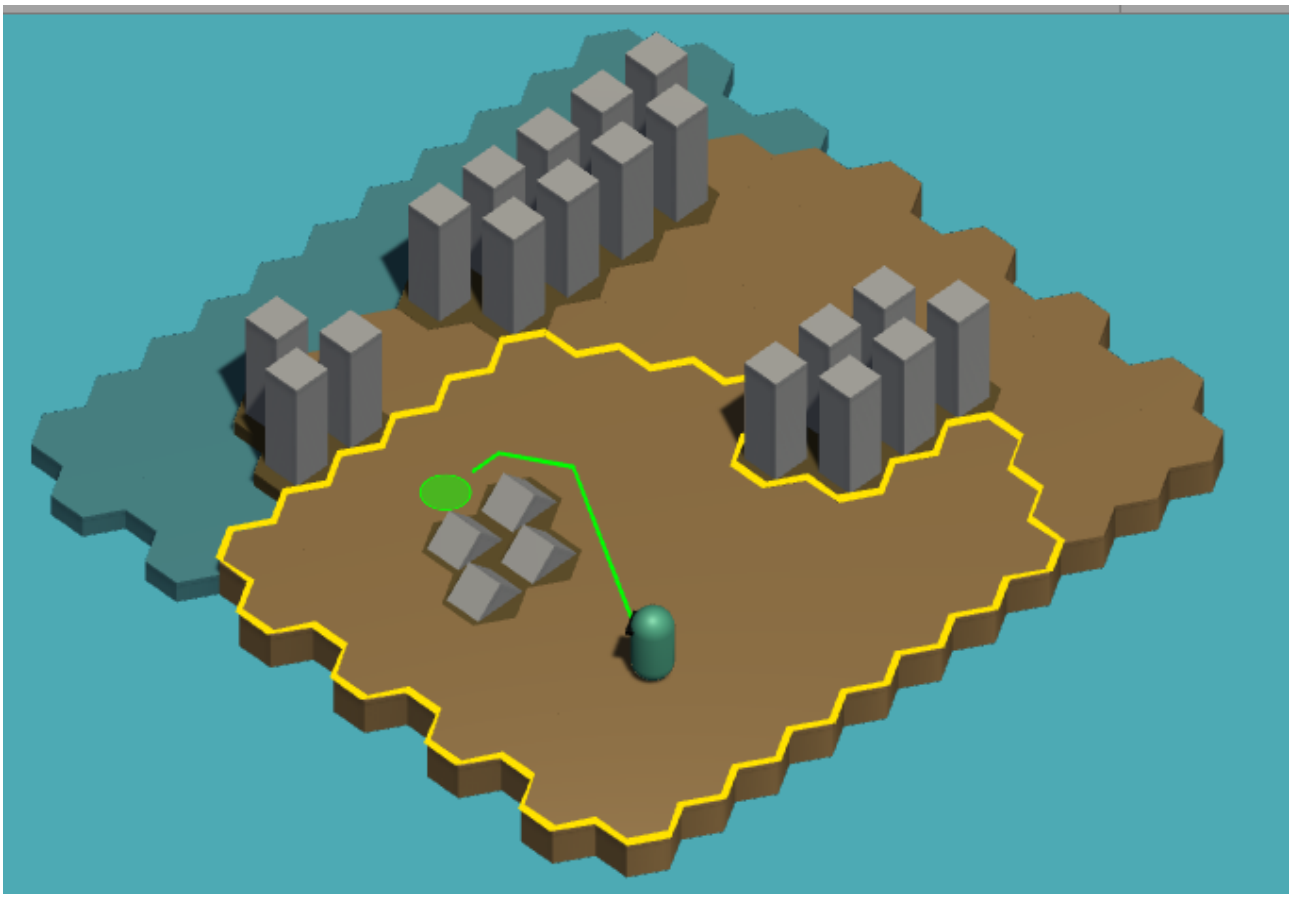
ProtoTiles

Documentation



Contents

Join our Discord

Overview

This asset is developed to speed up prototyping process of game developing.

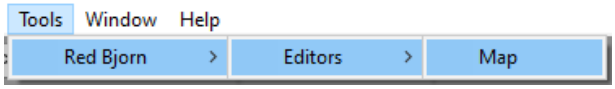
How? Just by creating helpful data with an easy tool

By the meaning of helpful data, we understand an asset (Map Settings) which contains information about bunch of tiles which represent a field where any kind of gameplay can be created. It is supposed that turn-based games use such data more often than other genres.

Easy tool means than we have clear way of creating data and it is called Map Editor. Simple steps will lead you to your first map

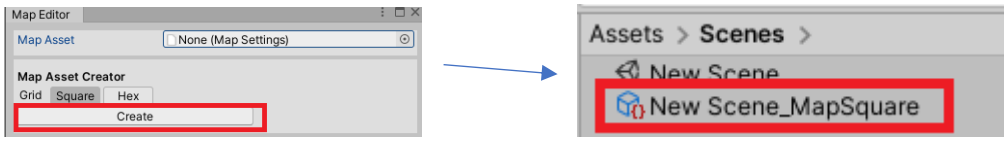
Quick Start

- Open Map Editor from Tools menu

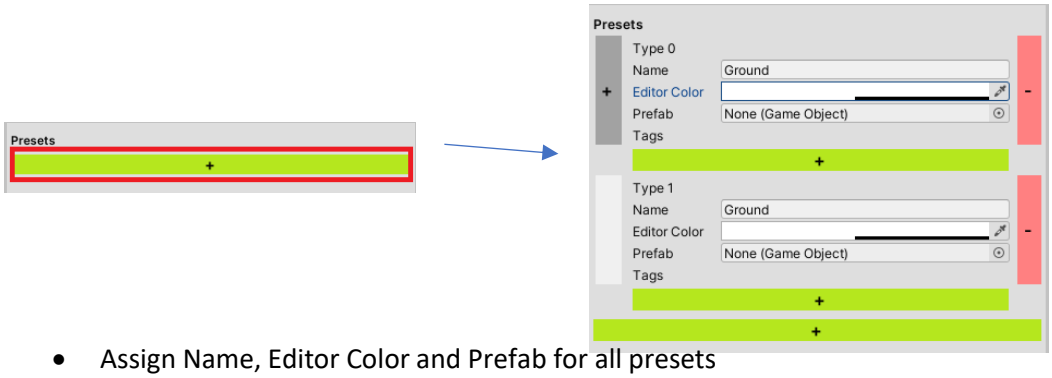


- Define which grid type should be used Square or Hex. And press Create

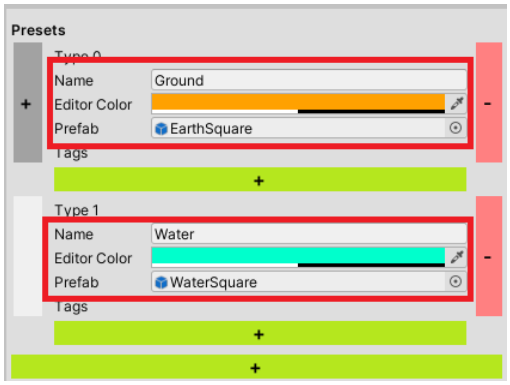
MapSettings asset will be created at the folder where active scene asset is located



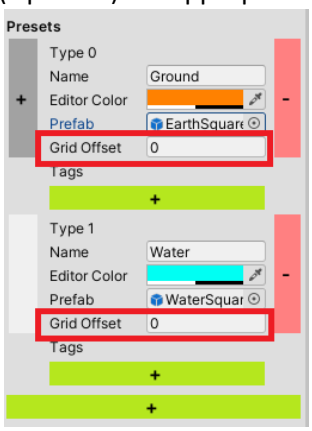
- Add two (or at least one) tile presets by pressing green button with plus sign.



- Assign Name, Editor Color and Prefab for all presets



- (Optional) Set approporiate Grid Offset.



For example, our water tile prefab's (WaterSquare) top surface is lower than zero on the Y-axis. So, setting offset to -0.1 will make it looks better. Compare Fig.1 and Fig.2

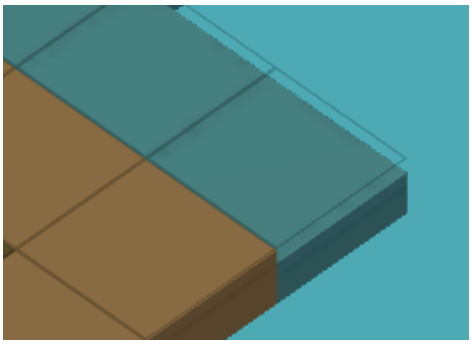


Figure 1 WaterSquare Grid Offset = 0

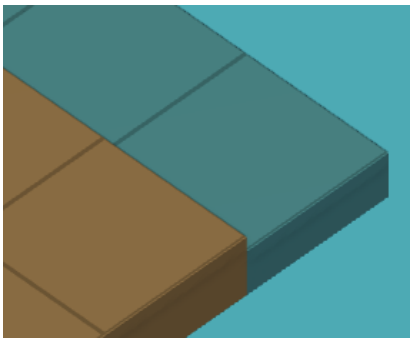
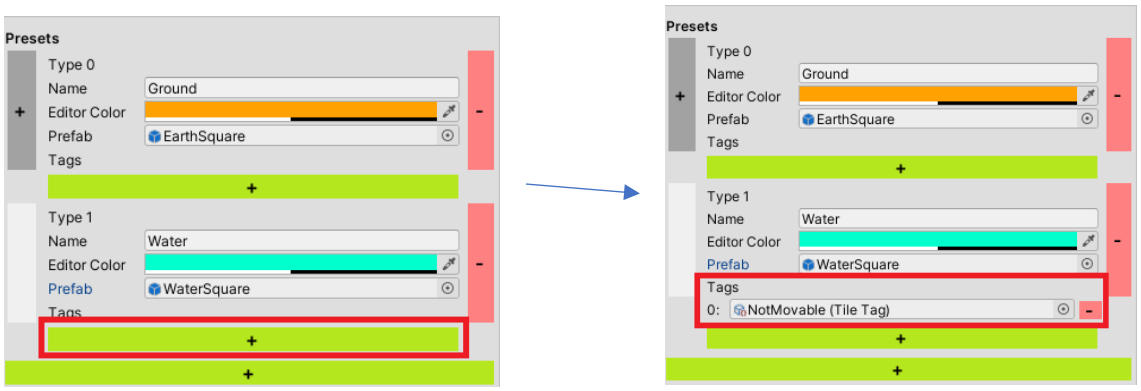


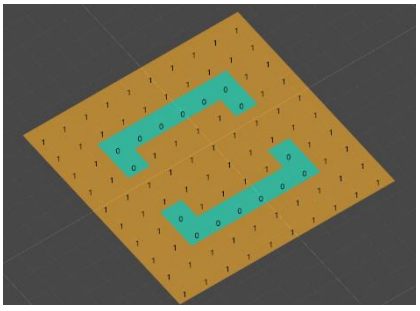
Figure 2 WaterSquare Grid Offset = -0.1

- Choose which preset will be marked as NotMovable by adding tag with same name

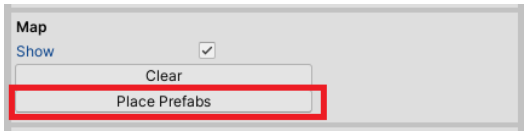


- Select brush tool

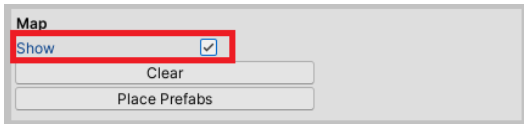
- Paint your map at Scene view

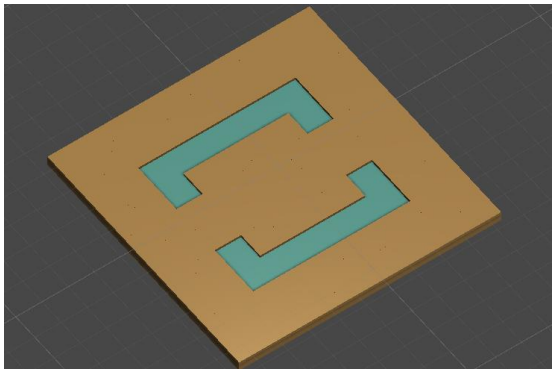


- Place Prefabs (from tile presets to scene view)



- Disable map

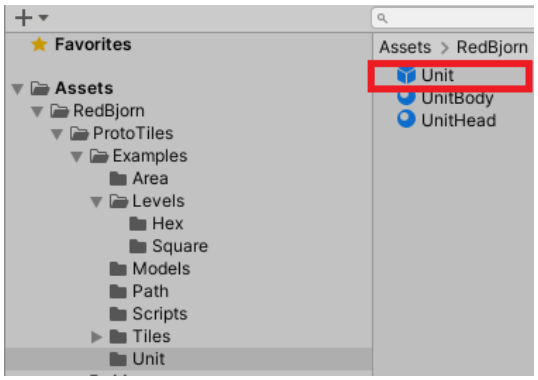




Join our Discord

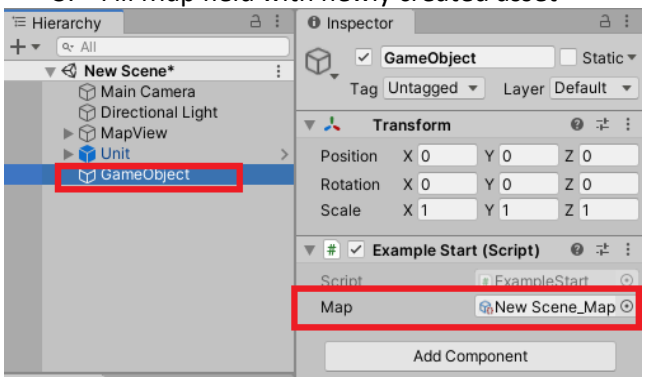
Additionally (For quick test)

Add Unit (Unit prefab from Examples)



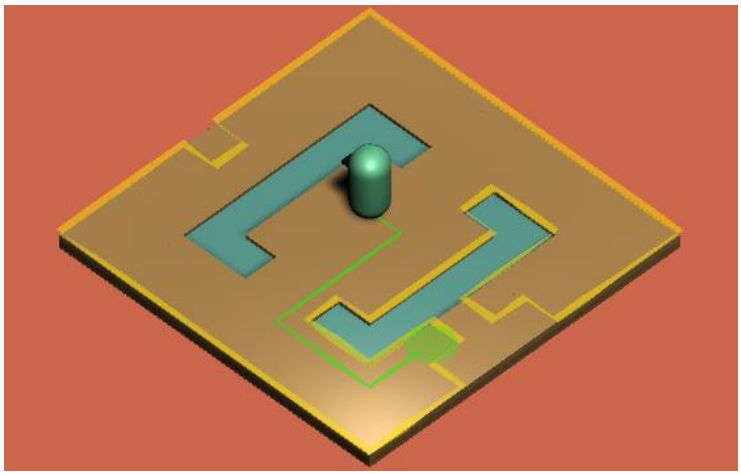
Add MapSettings link

1. Create new GameObject.
2. Add Example Start component.
3. Fill Map field with newly created asset



- Click playmode.

Ta-dam! Congratulations, you have created your first map with walkable information. Move Unit by clicking left mouse button to test how does it work



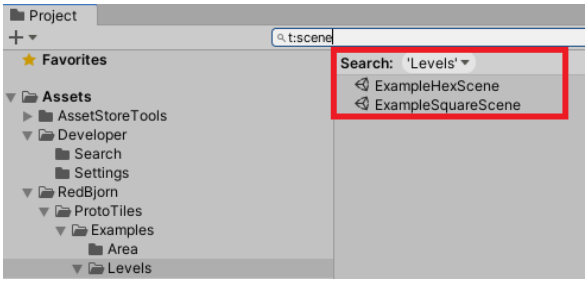
Essentials

Lets have a look at folder structure in extraordinary way: from bottom to top

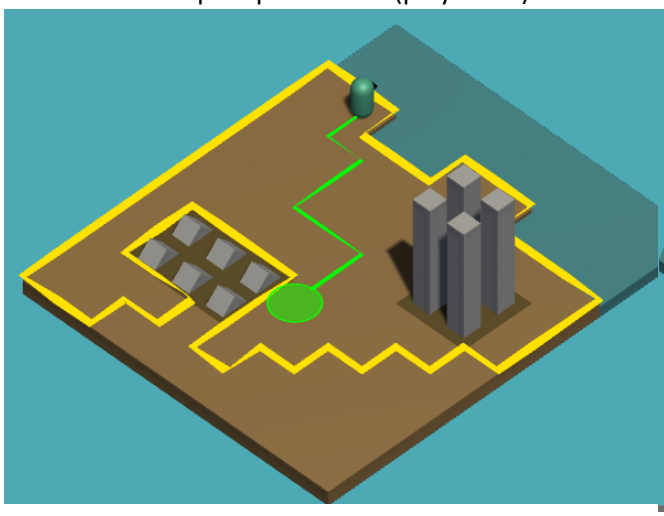
Utils	Folder includes plenty of useful extensions and methods for speeding up the developing process
ProtoTiles -> Map Editor	Folder contains scripts for drawing Map Editor window and logic of Brush tools at the scene View. Also, it includes several resources for UI of Map window
ProtoTiles -> Map -> Rules	Rules implemented as scriptable objects which specify logic for identifying walkable tiles
ProtoTiles -> Map -> Scripts	Includes wrappers for initials settings which usually contain Entity word. Also, the main script MapSettings with its dependencies is located here.
ProtoTiles -> Map -> Tags	Only one tag named NotMovable. It should be added to tile presets which will stop considering tile as vacant for pathfinding algorithm
ProtoTiles -> Examples	Contains 2 example maps with square and hexagonal tiles. Details at the Examples section

Examples

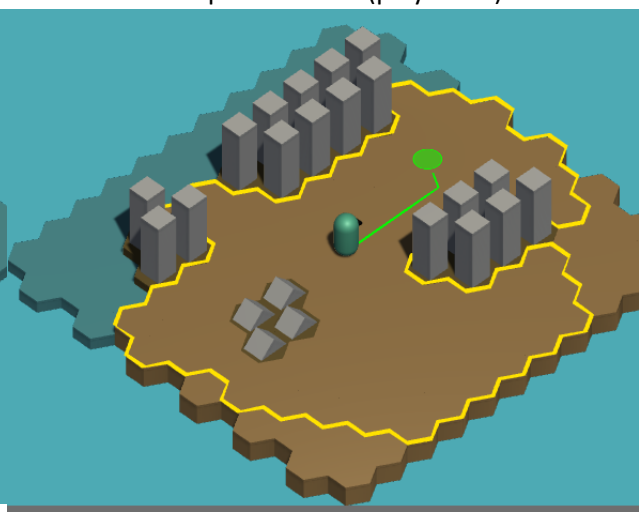
Package contains two example scenes



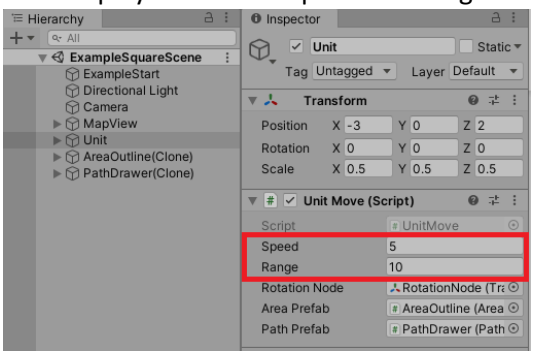
ExampleSquareScene (playmode)



ExampleHexScene (playmode)



- You can play with Unit's speed and range



- Unit moves by pressing up right mouse button inside area bounded yellow line
- Camera could be dragged by holding right mouse button and moving the mouse

Tips

- Create “Entity” classes for “Settings” classes during playmode. It prevents from modifying predefined map data and provides an opportunity to reset to default values

  MapEntity for MapSettings

  TileEntity for TileData

  It could be easily done by calling CreateEntity method which belongs to MapSettings class. Example could be observed at ExampleStart.cs

```cs
using UnityEngine;

namespace RedBjorn.ProtoTiles.Example
{
    public class ExampleStart : MonoBehaviour
    {
        public MapSettings Map;
        MapEntity MapEntity;

        void Start()
        {
            MapEntity = Map.CreateEntity();
            var unit = GameObject.FindObjectOfType<UnitMove>();
            if (unit)
            {
                unit.Init(MapEntity);
            }
        }
    }
}
```

- Most useful methods are located at MapEntity class

  Converting tile coordinates to world space coordinates and vice versa

```cs
/// <summary> Get world space position of center of tile entity
public Vector3 WorldPosition(TileEntity tile)...

/// <summary> Get world space position of center of tile bt it's coordinates
public Vector3 WorldPosition(Vector3Int tilePos)...

/// <summary> Get world space position of center of tile which is located at wor ...
public Vector3 TileCenter(Vector3 worldPos)...
```

  Calculating distance in tile space

```cs
public float Distance(TileEntity tileA, TileEntity tileB)...

/// <summary> Distance between two tiles located at corresponding world space po ...
public float Distance(Vector3 worldPosA, Vector3 worldPosB)...
```

  Calculating paths

```cs
/// <summary> Get path that consist of tile entities
public List<TileEntity> PathTiles(Vector3 from, Vector3 to, float range)...

/// <summary> Get path that consist of world space positions
public List<Vector3> PathPoints(Vector3 from, Vector3 to, float range)...
```

  Outline of walkable border

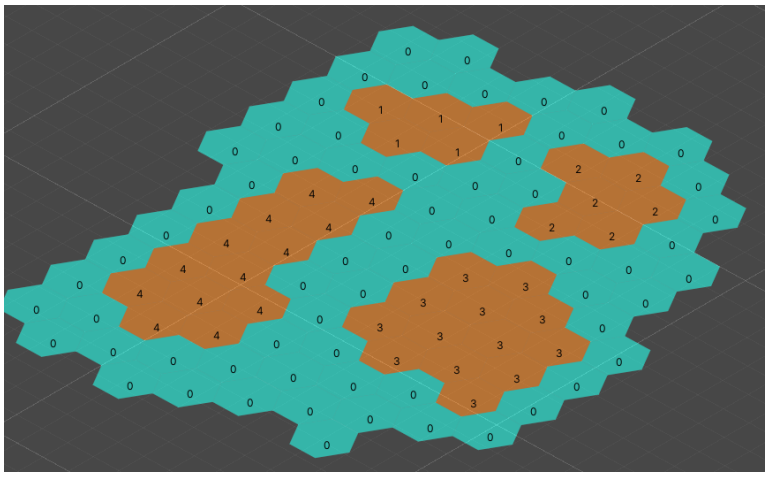
```cs
/// <summary> Get positions of border of walkable area
public List<Vector3> WalkableBorder(Vector3Int tilePosition, float range)...

/// <summary> Get positions of border of walkable area
public List<Vector3> WalkableBorder(Vector3 worldPosition, float range)...

/// <summary> Get positions of border of walkable area
public List<Vector3> WalkableBorder(TileEntity origin, float range)...
```

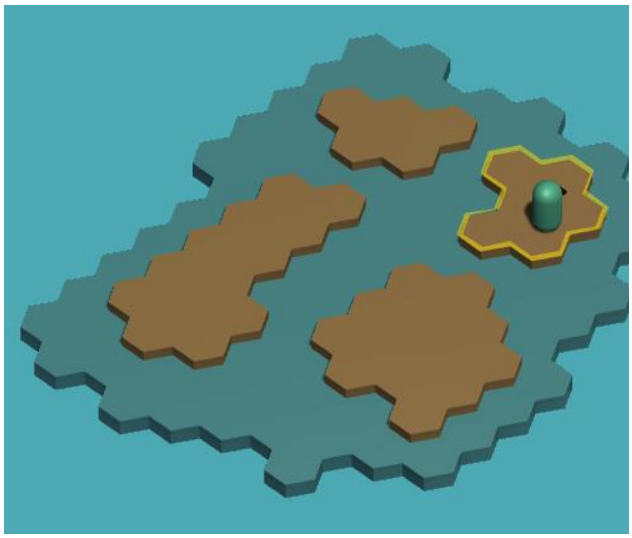
  Etc..

- Numbers which are located inside tiles during painting mode at Scene view indicate separate move zone areas
  0 index – predefined for NotMovable zone
  Indeces greater 0 - different movable zones



For example.

There is no walkable path from zone 2 to zone 1,3 and 4



There is no walkable path from zone 3 to zone 1,2 and 4