Insane Systems Road GPS Navigator Guide

Content

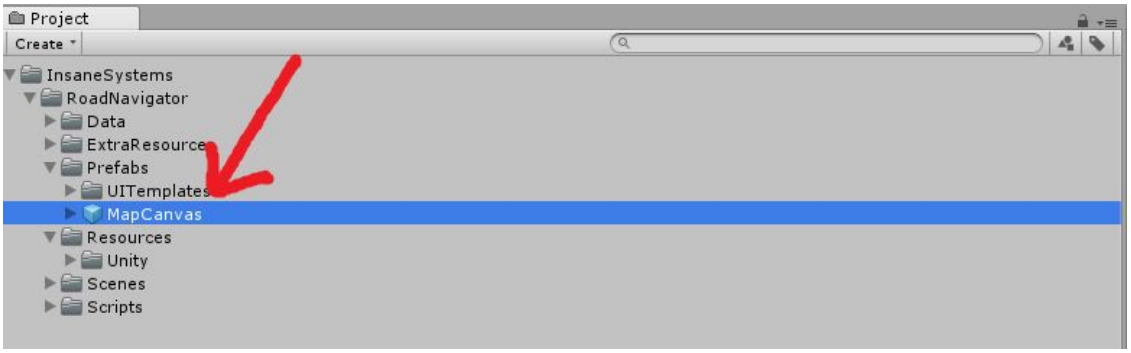
How to use

Before you start

Before you start to setup this asset for your levels, we highly recommend to finish all roads on your scene. Because changes on level (roads on level) also requires changes in Road Navigator settings.

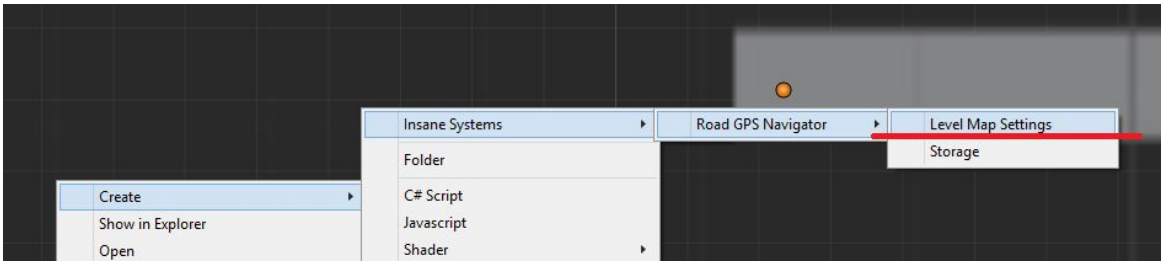
Setting up scene parameters

Setup your level with **Navigator** and **Map** is easy. First of all, you need to open folder Insane Systems -> RoadNavigator -> Prefabs.

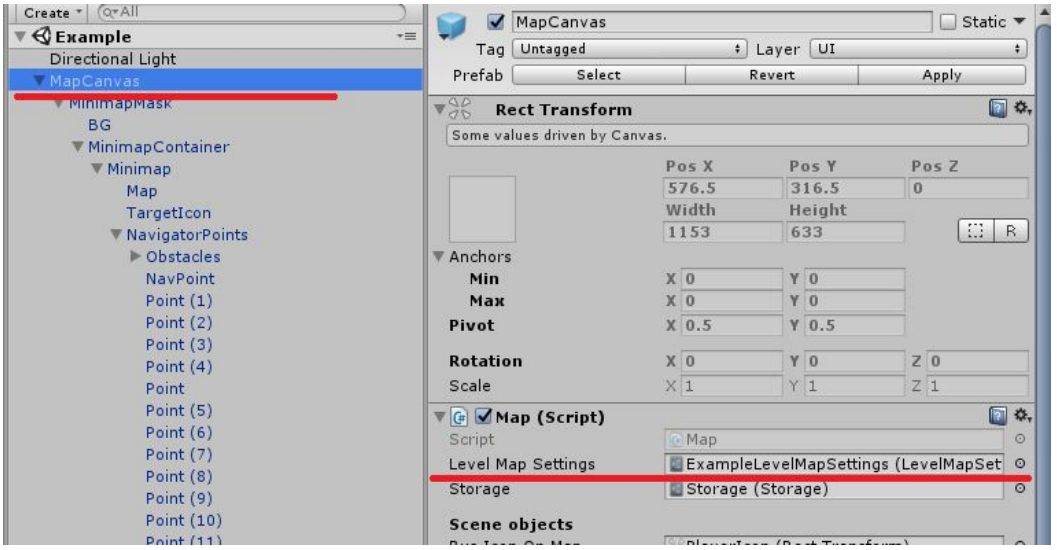


Now, bring **MapCanvas** to your Scene. This prefab already configured with base settings, you only need to setup it for your level.

After it done, you need to setup **Level Map Settings**. To do it, create new **LevelMapSettings** object by **Right Click** in **Project window** and selecting Insane Systems -> Road GPS Navigator -> Level Map Settings.

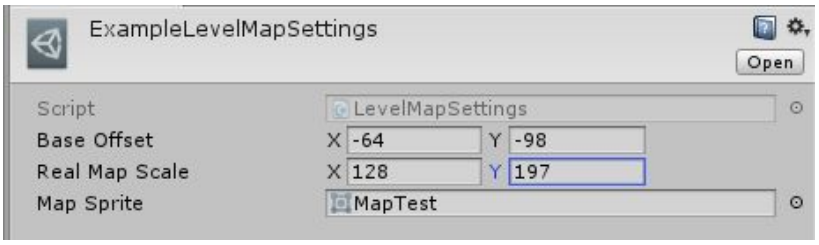


By default, this object placed in **Data/Levels** folder. Bring created object to scene **MapCanvas Map** component's **Level Map Settings** field:

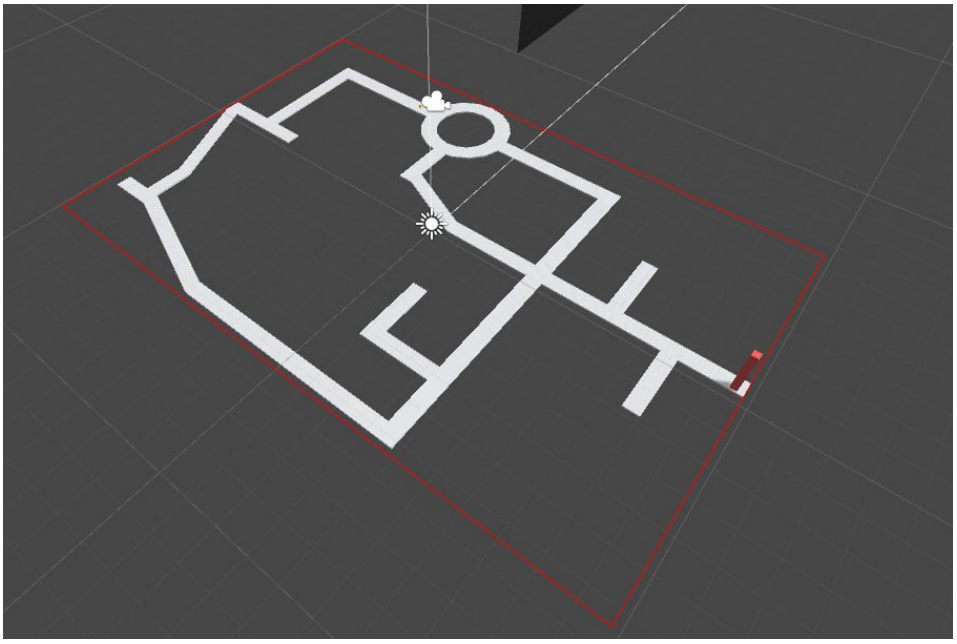


Setting up map

Now you need to edit created **LevelMapSettings** object parameters. Select it and you will see something like this:

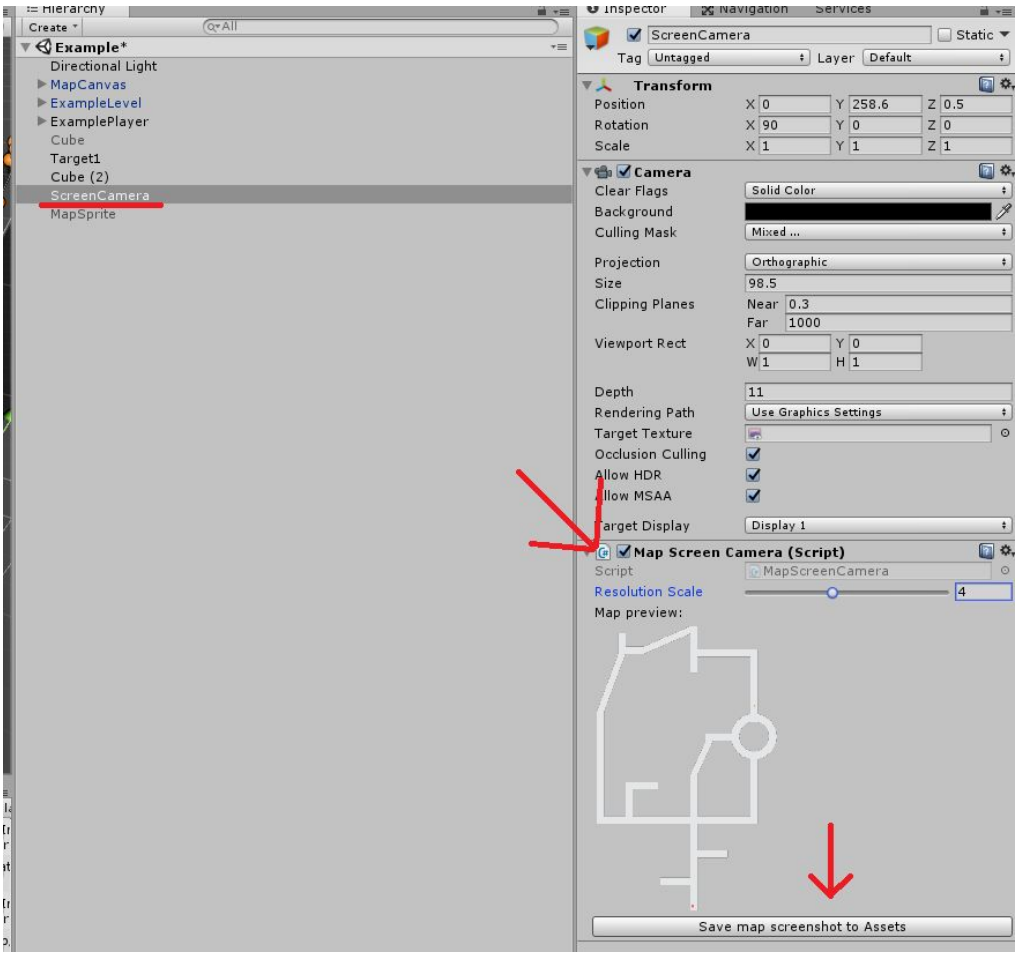


Edit this four number fields. When one of them changed, you will see **red rectangle** in **Scene** window. You need to fit your roads in this rectangle, like this:



Note that you need to fit all roads in this rect, but nothing more! It means, if you environment map is bigger than road map, put in rectangle only roads.

Now you need to make image from your map. It can be done by using our **ScreenCamera** Prefab with special component:



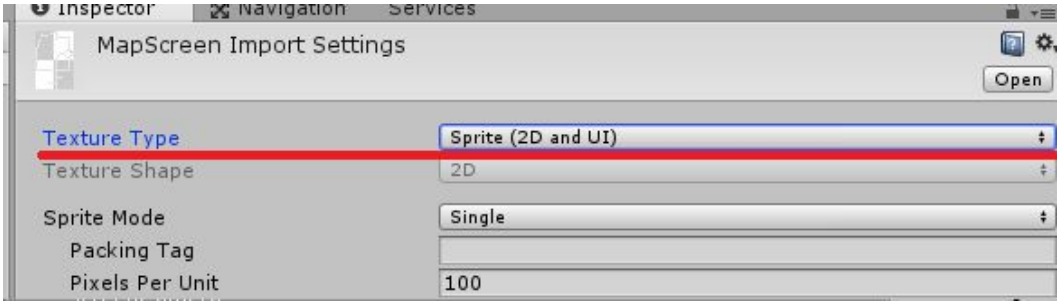
It automatically fits to parameters of **Level Map Settings** and always renders correct image of your map using Camera and RenderTexture. You can save this image by clicking **Save map screenshot to Assets**. Note that before capturing screenshot, you can disable not needed objects on map, leave only roads active.

Do it as you want, and finally, if you leave active only roads, and camera will see the skybox below roads, background of map will be transparent, so it can look good on map and navigator, try it. In our example it done this way.

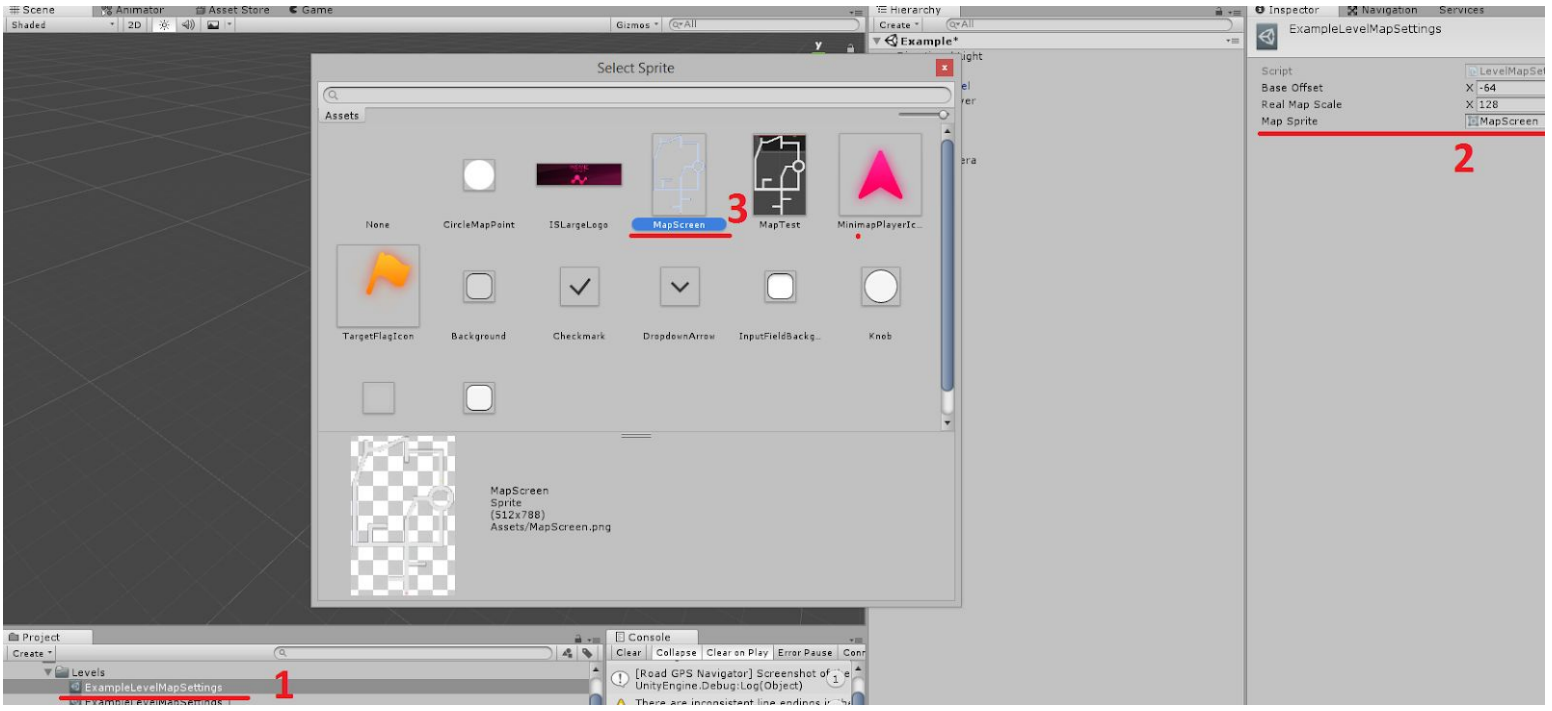
After map image is done, you can find It in project **Assets** folder, named **MapScreen**. You can use it as it is to finish setting up your level Navigator settings, or you can edit it in image editor, for example Photoshop (for changing visual style, etc). But in case you want to edit it, don't change image proportions, it will break work of Navigator and Map.

Finally, to setup this image as map and navigator background, do next steps:

1. Mark your image as **Sprite**:



2. Select your **Level Map Settings**, and set this image to field **Map Sprite**:



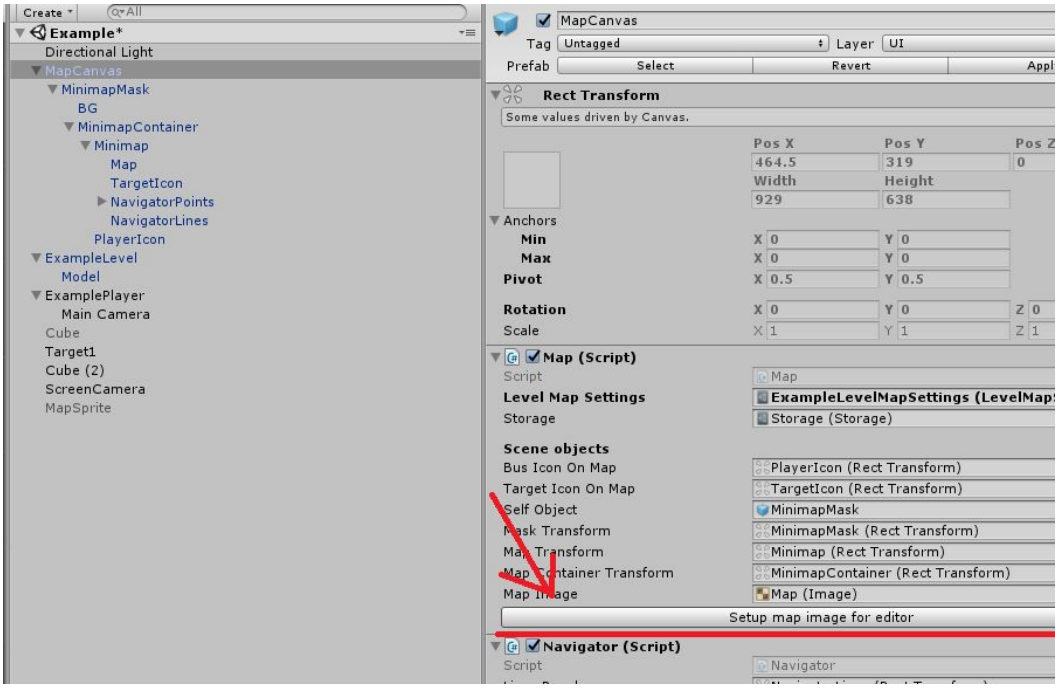
3. Run level and check, that map works and player position is correctly shown on map:

To show Map in game **press M**. (can be changed in asset settings).

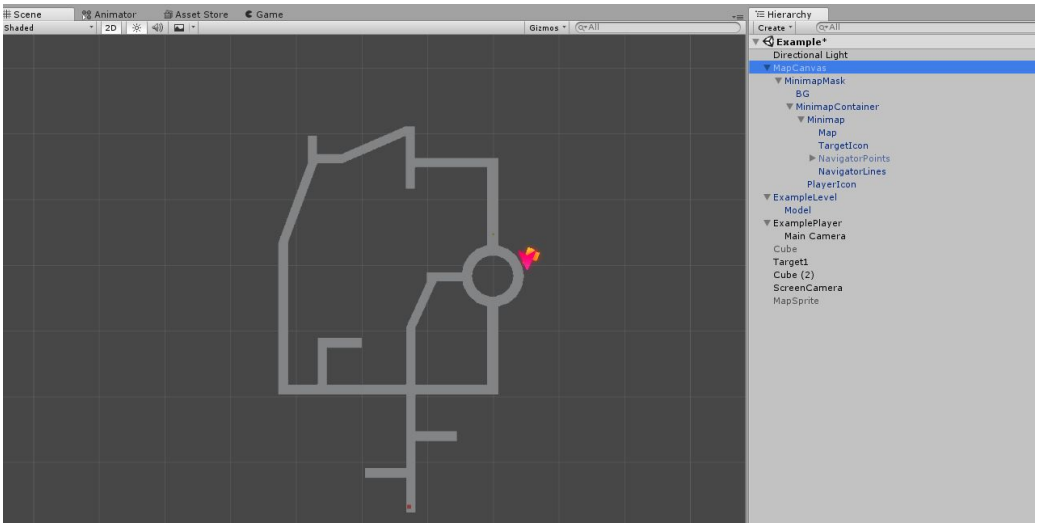
If all works fine, you did everything right! Now you can go to next step and setup **Navigator**.

Setting up roads

First of all, to see your new map image in editor, you need to hit this button:



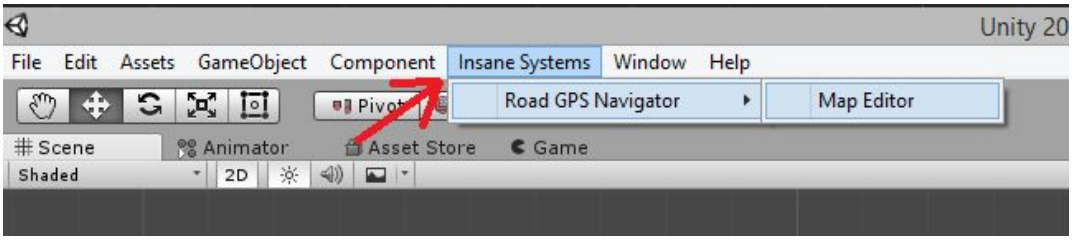
It will setup all **Map** and **Navigator** parameters to work with new map image and show your map image in editor:



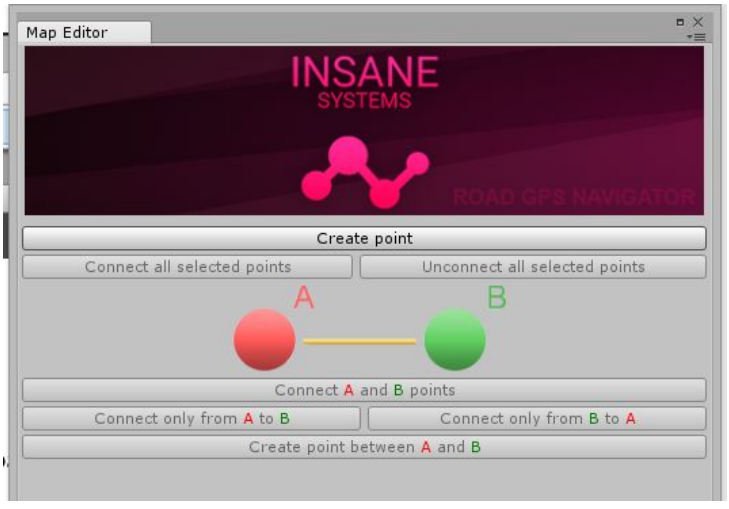
Note that you need do it every time you change image or its size. But if you're made screenshot of your final level, you'll need to do it once. ☺

Now you need to **setup roads on map**. It done by marking all crossroads on map and connecting it similar to your roads. For big maps it can take a long time, and for this reason we made **Map Editor**, which will save your time.

To open it, move cursor to Unity top menu, and select Insane Systems -> Road GPS Navigator -> Map Editor:

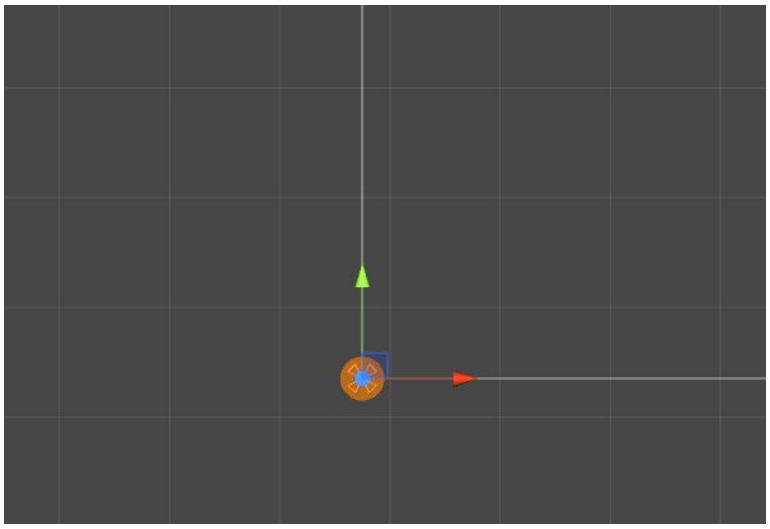


Will be opened this window:

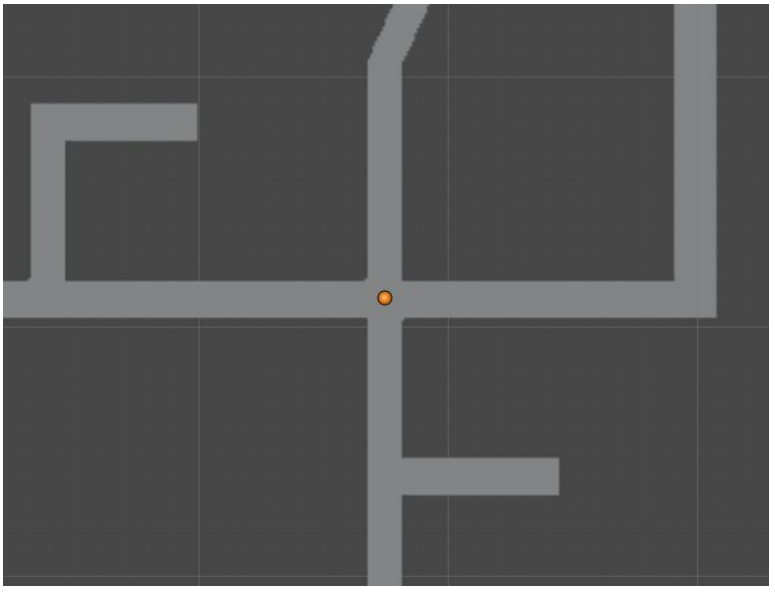


Yes, this is Map roads editor. It can look little different in your version, because sometimes we can add some new functionality.

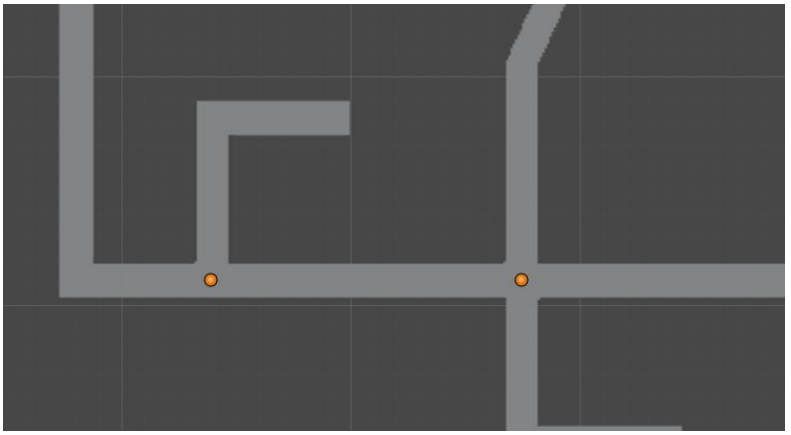
Now, let's start building roads. Better to do it looking at your map canvas in 2D mode of Scene (Map editor will suggest you to change it if you're using 3D mode when opened it). To start, hit **Create point** button. Result will be:



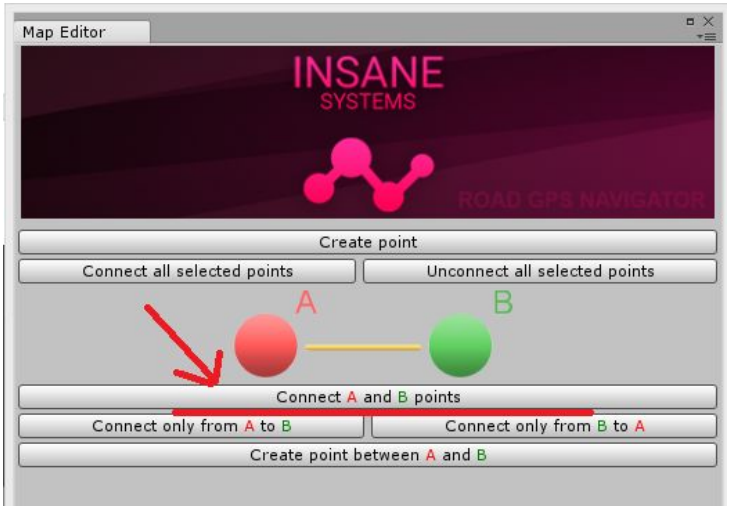
Point on **Scene** was created. This is **Navigator Crossroad Point**, described before. Now, move it to any crossroad on your map image:



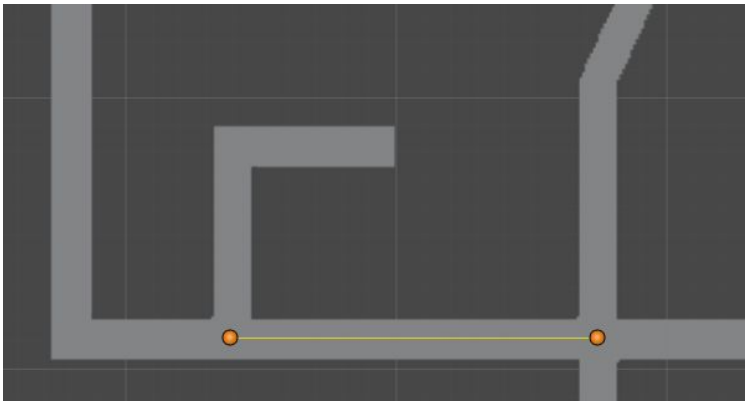
Now, create second point this way (or duplicate first point) and move to another crossroad:



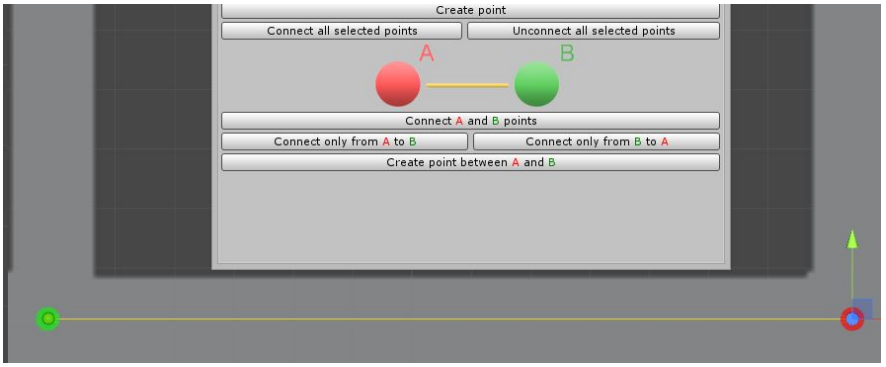
Finally, select two points and click **Connect A and B Points** in **Map Editor**:



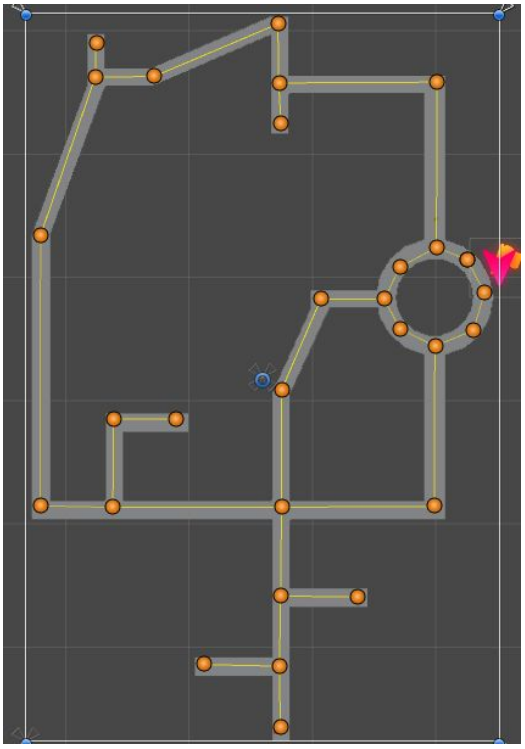
This action will create two-side connection between this crossroads (it means that now if player will be near one point, and his path will move along second point, navigator will draw line between them), you can see it in editor:



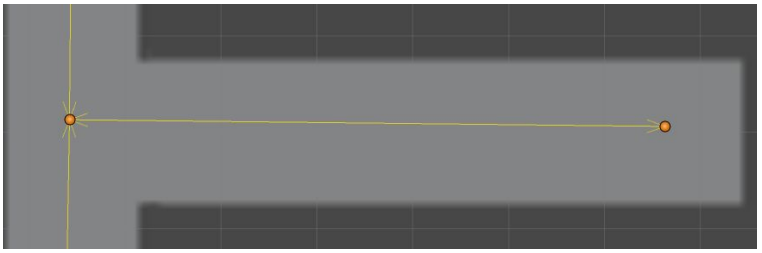
This is road with two-side movement. To create one-side movement between two points, hit **Connect only from A to B** or **B to A button**. While editing only two points, you will see help color markers on points (red and green):



You need to connect all roads on your map this way. Finally, it can look like this (to see all connections, select **MapCanvas**):



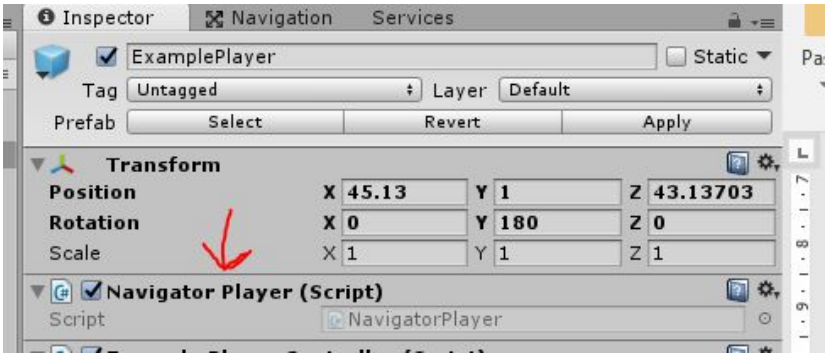
Also, with bigger zoom you can see directions of your roads, for example two-sided:



One-sided:

Setting up Navigator Player

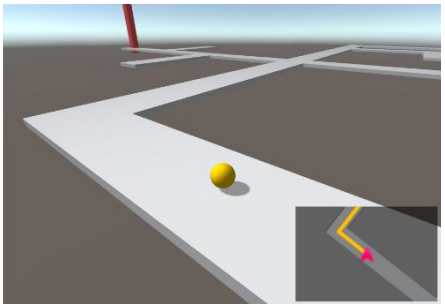
Navigator Player is a script, that indicates, what object should navigator follow. It can be your player, car, etc. You need add this component to your player object:



Now, you can run game to test your first navigator settings.

To show Navigator in game **press N**. (can be changed in asset settings).

Navigator should work like this (captured on Asset Example scene):

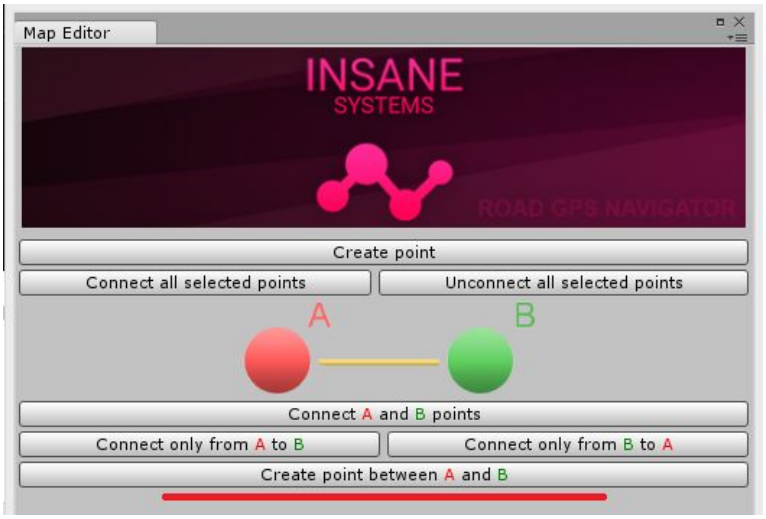


But it will not work without target, this will be described in next partitions.

Adding detail points

If your roads contain bends, you also can add in this positions Crossroad Points, just connected to 2 other points on this road:

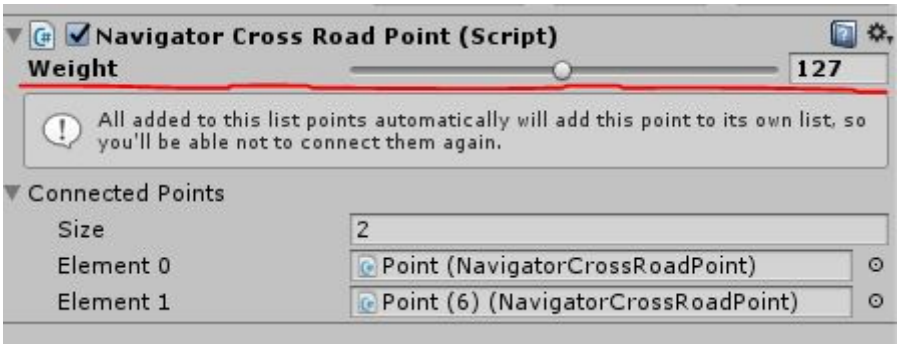
Best way to do it: select two side points and use **Create point between A and B** button in **Map Editor**:



More detailed points on map means that navigation line will be better quality. For example, it can be used for circle roads.

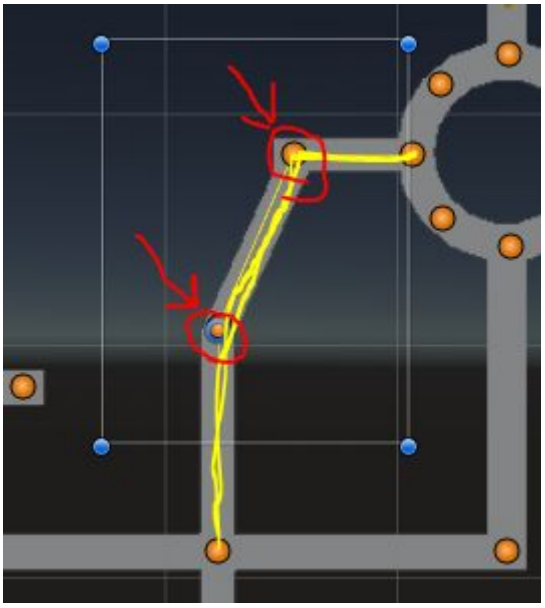
Path points weights

If you want to add priority to some road on your map, you can do it by reducing weight of points on this road. To do it, change this parameter:



How it works? Navigator selects 3 shortest paths from current player position, and calculates average weight of its point. Path with smaller average weight value will be selected to draw navigation line on it. Finally, what you need to do:

1. Select key points of priority road on map:



2. Decrease their weight to value 126 or lower (127 is default value).

But be careful – this feature can give you wrong results, if weights will be setted up not fully correct.

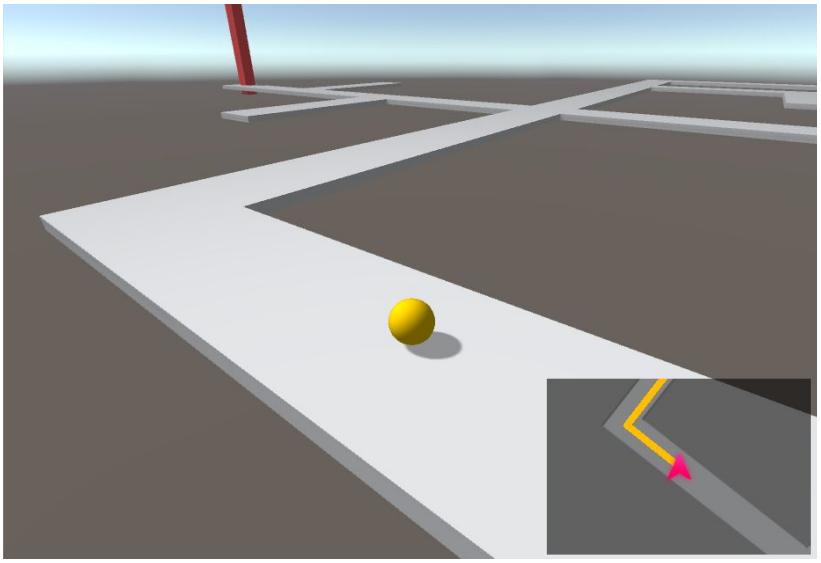
Setup navigator target from code

Now, when you finished with settings of all components, you will want to set targets to move on navigator from your code. It can be done very easy:

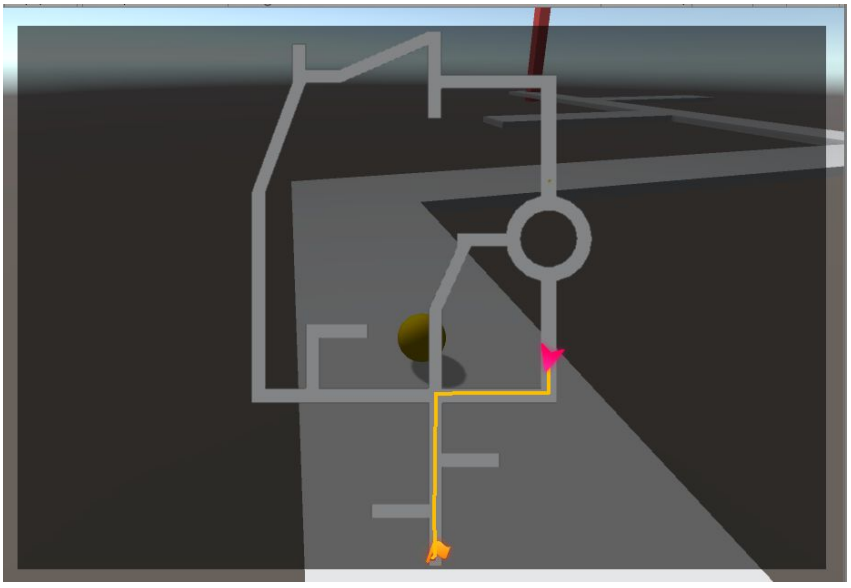
```
var navigator = FindObjectOfType<InsaneSystems.RoadNavigator.Navigator>();  // you need access to navigator component, you can do it any way you
                                                                             // want, I used example, which will work from any code.
var examplePosition = new Vector3(35, 15, 25); // setting up example point to draw line on navigator.

navigator.SetTargetPoint(examplePosition); // finally, setting this point as navigator target point

// or:

examplePosition = transform.position;
navigator.SetTargetPoint(examplePosition);
```

Note that you need to pass your object world position in this code, I mean same that shown in **Transform** component in **inspector**.

And final test of navigator:

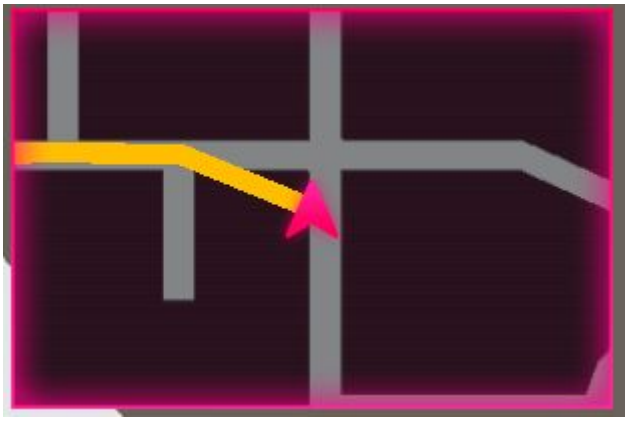


And how it looks on map:

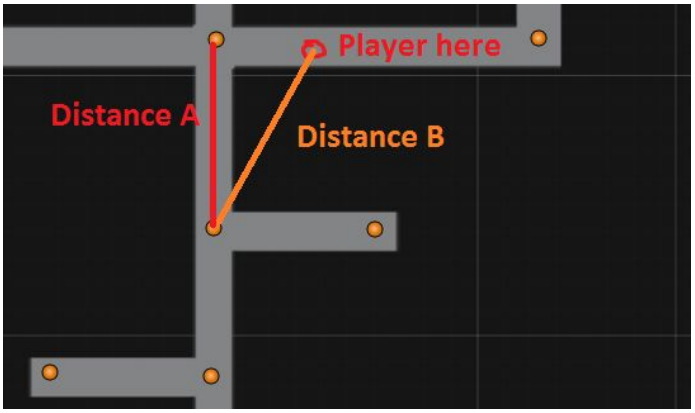


Obstacles

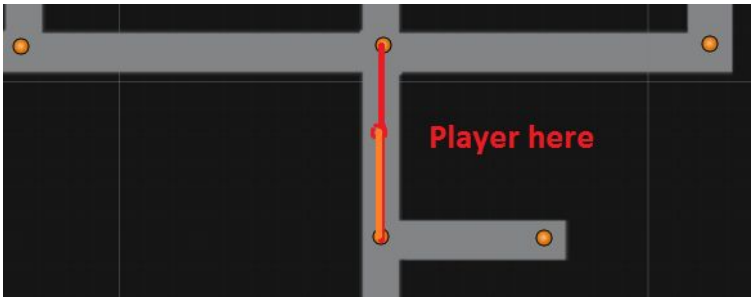
Sometimes you can notice troubles with navigator line, which looks like bug:



This happens because straight distance between player and next crossroad point is smaller, than distance between nearest to player crossroad point and next crossroad point:



Distance A bigger than Distance B (maybe this is not clear from example, but it tested and it right), so Navigator decides to draw line from player to next point, because this will be right in this situation:

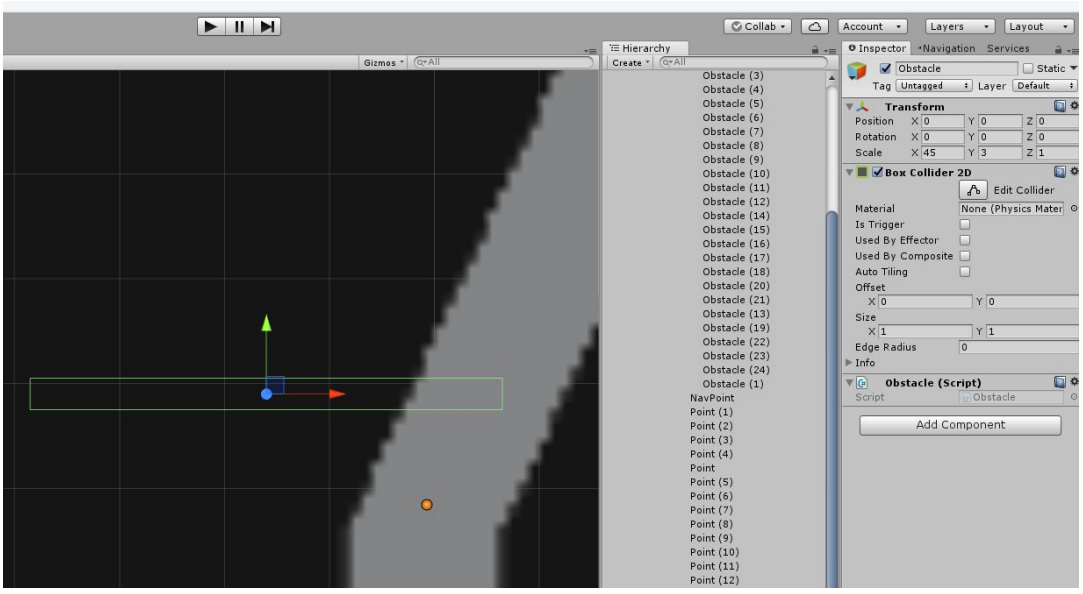


This is same situations, but first is wrong, but second is right. To solve this problem, we added obstacles system. **Obstacle** is **Box Collider 2D** that prevents wrong situations like this.

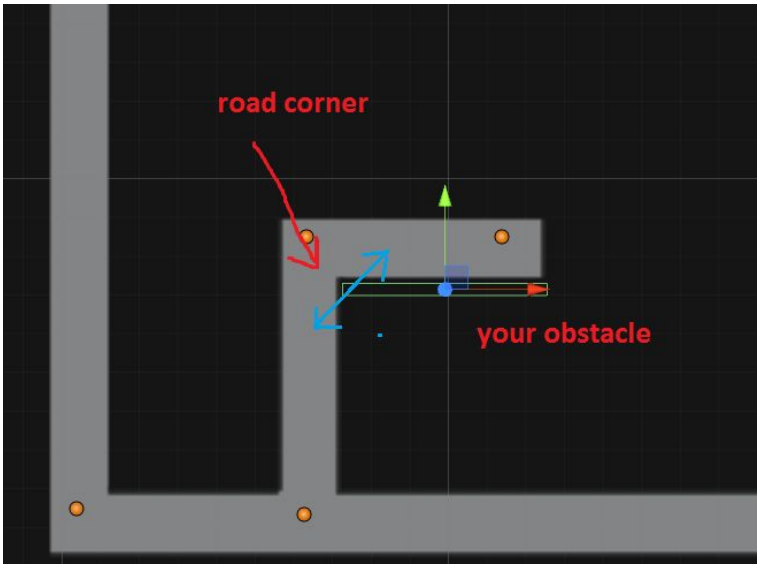
It is <u>not necessary</u> to add obstacles, but if you will setup them, Navigator will work better.

So, how you should do it? Very easy. Open **Map Editor** and hit **Create obstacle collider** button:

This action will create obstacle object on scene:

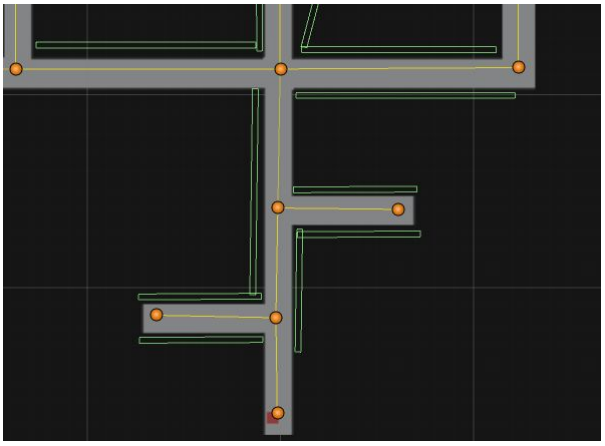


Now you can setup it by editing its **position** and **scale** transform parameters. Place obstacles on roads sides and near road corners like this:



Blue line on screenshot is possible line problems, which will be prevented. Keep it in mind and it will be easy to setup obstacles.

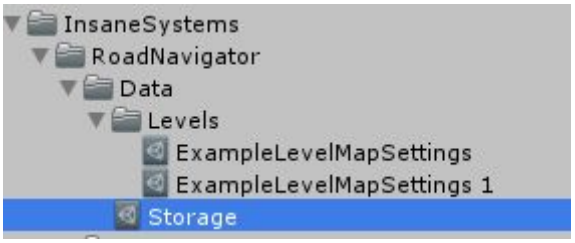
Final setup can look like this:



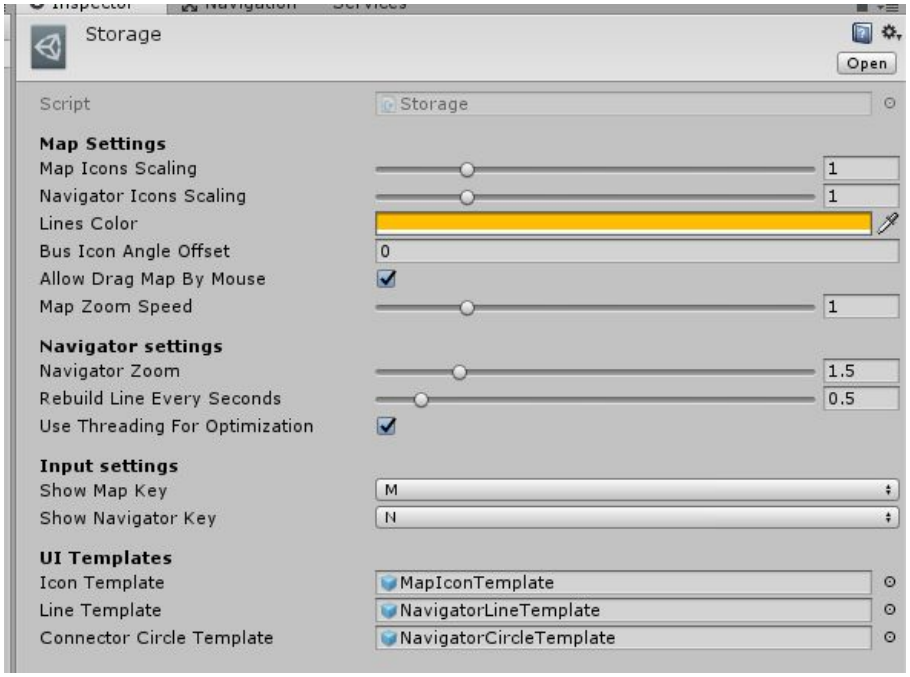
P.S.: we understand, that obstacle system looks like workaround and we will try to improve this system to reduce time, needed to get it working correctly.

Settings of Map and Navigator

You can find all asset settings in file named **Storage**:



It parameters looks like:

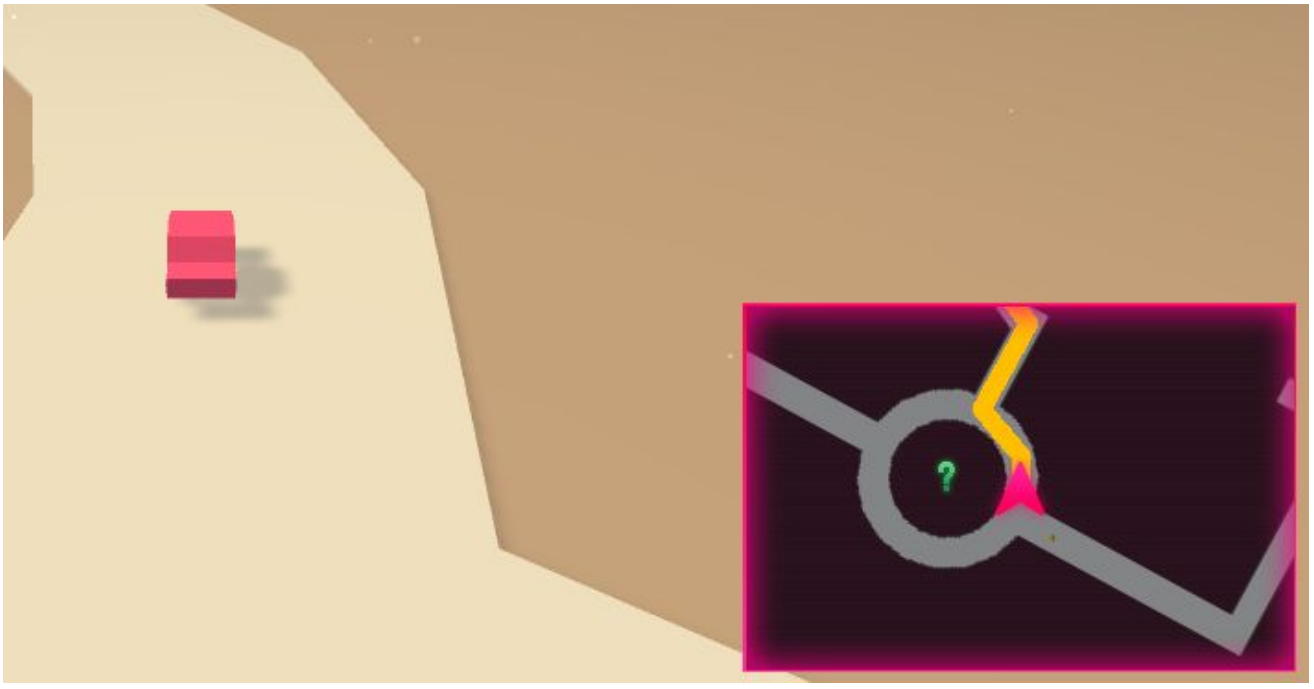


You can edit it as you want. In new updates we will extend these settings, Map Editor and other asset things.

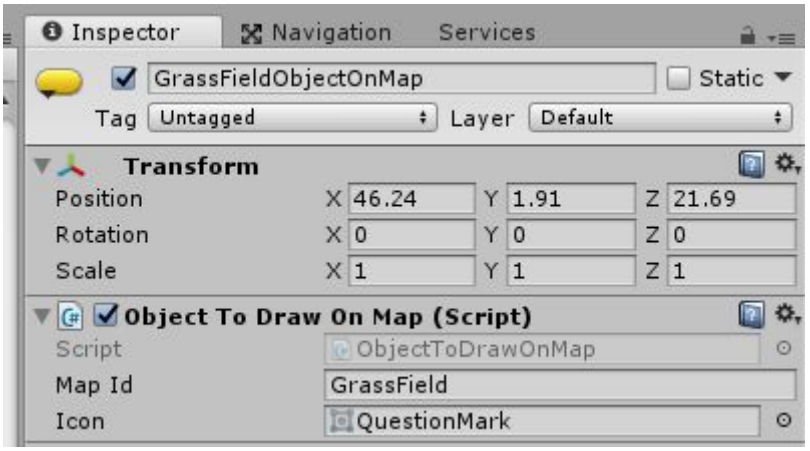
Adding custom icons to map

You can add some icons to your map, for example if your game have missions or some objects like Gas Station, Garage, etc.

How it will look (here it is question mark):



To do this, just add on any your object script named **ObjectToDrawOnMap**, and setup its parameters:



Code examples, how to add it from code and also way to remove it, if you no more want to see it on map:

```
Map minimap = Map.sceneInstance; // getting Map class instance on level

minimap.AddObjectToMap(icon, transform.position, "IconName");   // adding custom icon in position of this object with name IconName.
                                                                // Now it will be shown on map.

// and we can also remove this icon anytime:
minimap.RemoveObjectFromMap("IconName"); // just pass name of your icon, which was previously added.
```

Contacts

You can ask your questions or send your suggestions to us. ☺

To contact use email **godlikeaurora@gmail.com**

Actual version of documentation online:

https://docs.google.com/document/d/1CnE5ILyg6DRnI4pkEu5826_ALbSlk4cuDQbcF3YbRu0