Map and Radar Navigation System

Introduction

Thank you for buying this package. You can use this Map and Radar System in any kind of PC, Console, WebGL or Mobile games. Your games will automatically have a Radar and Map system.
It supports and has Map, Radar and Navigation functionalities.

Installation

After downloading this package, please import it in your project. It should be working without any error. A folder named “MapAndRadarSystem” should appear in your Asset level directory. Everything about this package is contained in that folder.

You can check SampleScenes (There are two sample scenes. These are SampleScene_Maze and SampleScene_Terrain) and find all the prefab and managers you need in the scene. Assign the player and your main camera and tadaaa. It is ready to use.

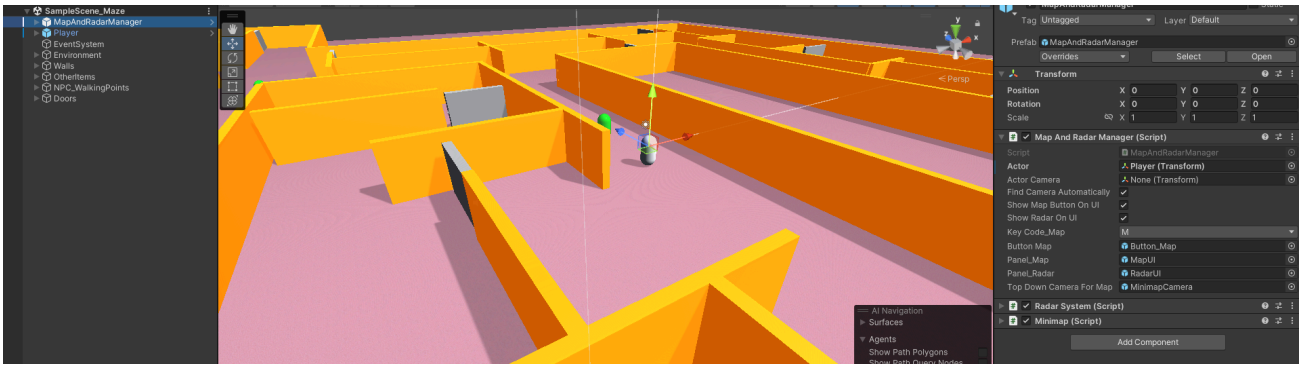
If you want to add MapandRadarSystem prefab into your own Scene, you can copy MapAndRadarManager gameobject from SampleScene (Or you can find it in Prefabs\ directory as well):

Let’s check it what it has in it:

- **MapAndRadarManager:** Manages the basic mechanics and configurations of both Map and Radar system.
- **GameCanvas:** All UI related objects are here. Canvas of the Package.
- **Button_Map:** Map Button UI on the scene. If you are developing mobile games, players can tap on it can open the map panel.
- **MapUI:** All UI elements of Map Panel are under this game object.
- **RadarUI:** All UI elements of Radar Panel are under this game object.

Configure your MapAndRadarManager

Map and Radar Manager script has all the configuration for you. By this script, you can configure your game’s map and radar details. Let’s check the details here.



- Actor: Assign your main character, player, vehicle or whatever you want here. Map camera and Radar camera will follow your actor and locate it center of map and radar UI.
- Actor Camera: Assign your actor camera here. The Manager will check its rotation and rotate the map and radar depending on your actor camera’s rotation values.
- FindCameraAutomatically: If you don’t assign your actor camera and tick this

feature, the Manager will search and try to find your main camera in order to use as actor camera.

- Show Map Button On UI: You can show or hide the map button on the UI.
- Show Radar On UI: You can show or hide the radar on the UI.
- Key Code Map: If you are developing a PC game, you can assign any keycode in order to show Map Panel on the UI during the runtime.

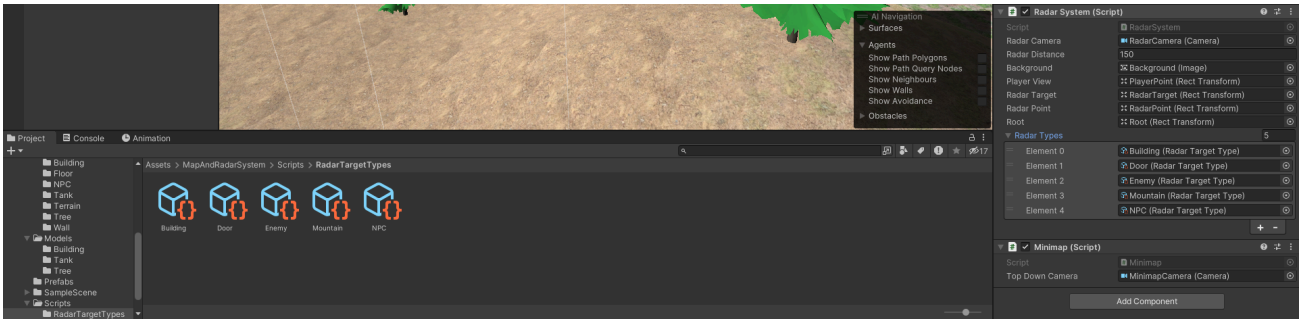
Radar System and RadarTargetTypes

The asset has got Radar Target Types. You can easily create your own RadarTargetTypes and assign them.
Radar System is very useful component in order to check the objects around us and the distance between them and our main actor. You can create your own Radar object by assigning RadarItem script on the object.

There are 5 ready to use types of objects which we can see on the Radar. These are:

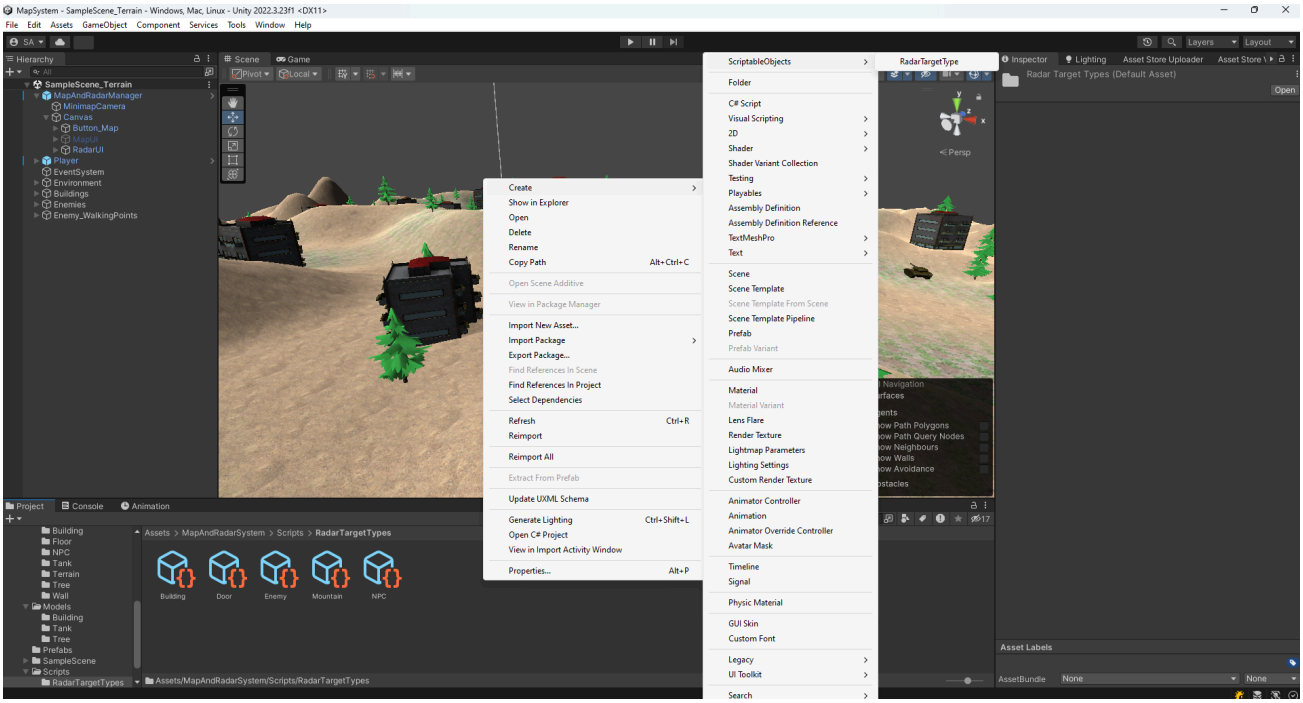
- Building
- Door
- Enemy
- Mountain
- NPC

You can configure the Radar Sensitivity Distance and the object types by this Radar System script on Radar System script.

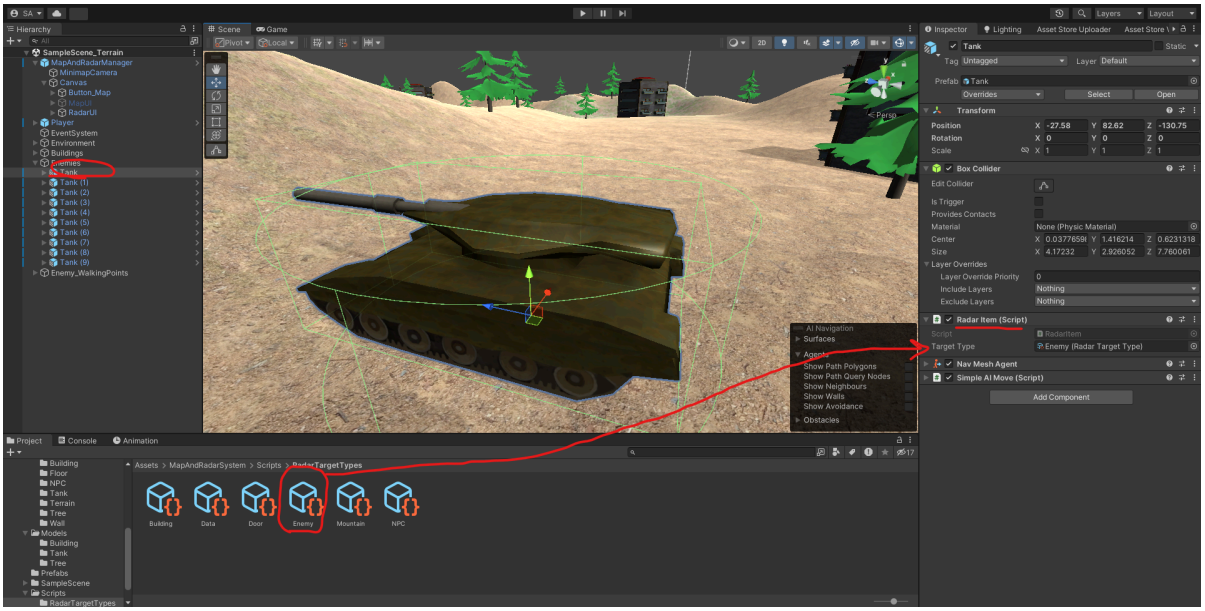


If you want to create a new RadarTargetType, please just right click on the project panel, go to Create, ScriptableObjects and click RadarTargetType.

After that; 1. Name your new RadarTargetType and Assign a Sprite (This sprite appears on the Radar). Your new RadarTargetType is ready. Now you need to assign it to “RadarTypes” list on the MapAndRadarManager object’s RadarSystem script. It is ready to use



Now, you need to add a “RadarItem” script on your game objects and assign their types in order to make them appear on the radar:



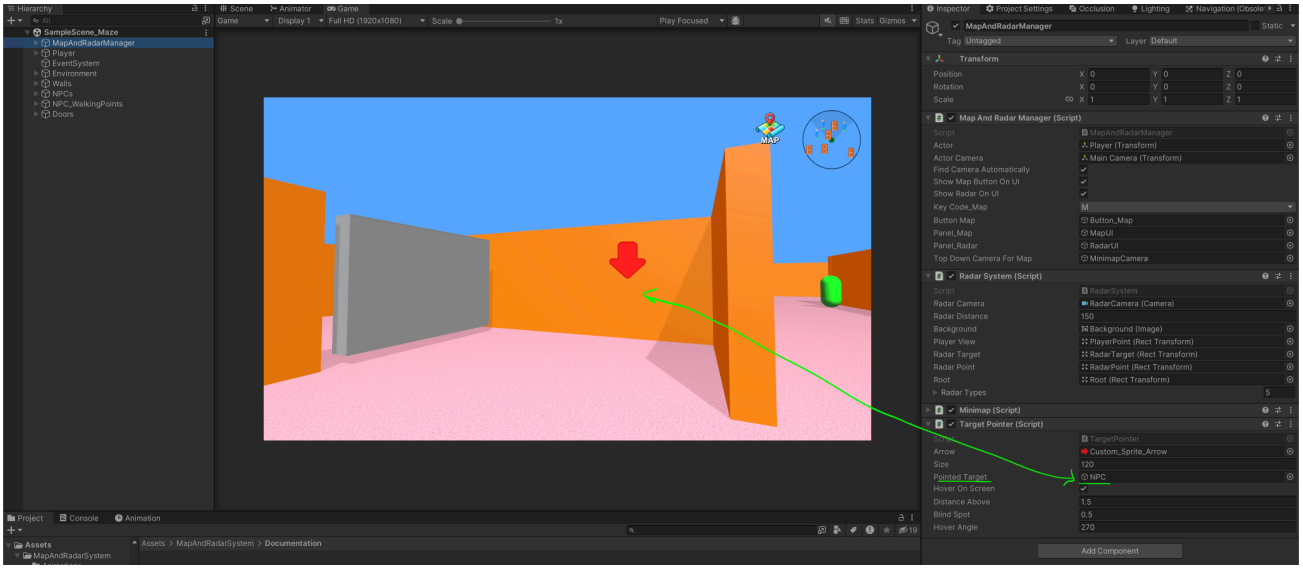
Target Pointer (Navigation System)

Sometimes, you may want to point and give a direction to the target on the UI. An arrow can point the target. This target pointer system is in the asset as well. If you want to point a gameobject, you can need to assign a gameobject to the PointedTarget variable on TargetPointer script. For example:

TargetPointer.Instance.PointedTarget = CarGameObject;

Or if you want to give up to point the target, you can just simply assign null to the same variable:

TargetPointer.Instance.PointedTarget = null;



If you need any help or If you have Questions, you can contact me via email queendeveloper95@gmail.com

Kind Regards, Queen Developer