2D Arena Minimap

1. Introduction

The 2D Arena Minimap Asset was developed to help users setup a basic minimap for 2D games. The asset provides users with a simple way to set Minimap Markers for player, allies and enemies, as well as Unique markers and death markers.

2. Classes

The classes were made to be easily modified and understandable. The game utilizes four classes: **Manager, Object, Marker, and Camera**.

The **Minimap Manager** updates and initializes all markers and cameras. Manages the sprites and layers used in the minimap.

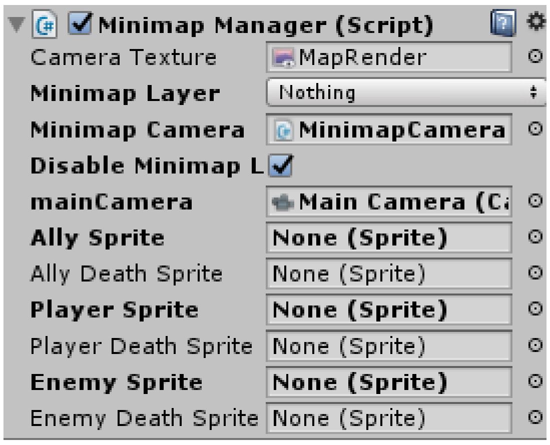
The **Minimap Object** sets the type of object, and the marker’s state (enabled, disabled, death).

The **Minimap Marker** is the actual marker/icon that will be used on the Minimap.

The **Minimap Camera** renders the Markers and other objects that will appear on the Minimap.

2.1. Minimap Manager

The **Minimap Manager** is the main part of the asset. The Manager contains a list of all the active markers on the object, sets the RenderTexture and CullingMask for the Minimap camera and the Main Camera.



Minimap Manager (Script)

Camera Texture	MapRender
Minimap Layer	Nothing
Minimap Camera	MinimapCamera
Disable Minimap L	✓
mainCamera	Main Camera (C
Ally Sprite	None (Sprite)
Ally Death Sprite	None (Sprite)
Player Sprite	None (Sprite)
Player Death Sprite	None (Sprite)
Enemy Sprite	None (Sprite)
Enemy Death Sprite	None (Sprite)

Fig. 01 – Minimap Manager Inspector

- **Camera Texture -** RenderTexture in which the minimap will be rendered. The asset already comes with a 256x256 RenderTexture, but feel free to change or modify it as you want.

- **Minimap Layer –** LayerMask that contains all the layers that will be shown on the minimap by setting the minimap camera's *LayerMask*. You can make your own layers and mark as many layers as you want.

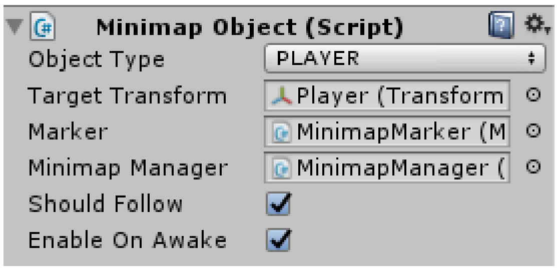
- **Minimap Camera –** Holds the camera that will be used to render the minimap, containing a **MinimapCamera** script.

- **Disable Minimap Layer on Main Camera –** By setting this bool as *True*, the **Minimap Manager** will try to disable the selected layers on the **Minimap Layer** from the Main Camera's culling mask. This way, the markers and other minimap objects will not be rendered on the main camera. You can also set the Main Camera layers manually, in case you have a layer that needs to appear on both cameras.

- **Sprites –** Set the sprites for each marker Type and state. For example, all markers with the *Ally* type will have the *Ally Sprite*, and will become an *Ally Death Sprite* on death.

2.2. Minimap Object

The **Minimap Object** component needs to be attached to every object that will generate a *Minimap Marker*. Through the *Minimap Object*, you can set the marker's type, enable or disable it, and set its state to "Death" sprite. In the Minimap Object, you can also set if the marker will follow the Object or stay static on the Minimap.



Minimap Object (Script)

Object Type	PLAYER
Target Transform	Player (Transform
Marker	MinimapMarker (M
Minimap Manager	MinimapManager (
Should Follow	✓
Enable On Awake	✓

Fig. 02 – Minimap Object Inspector

- **Object Type –** Sets the Marker's type, which in turn will set the correct sprite. If the Object Type is set to *Unique*, you can manually set a Marker Sprite on the Minimap Object's inspector, which can be useful to set itens, constructions, and others.

- **Target Transfom –** Sets the transform that the Marker will follow.

- **Marker –** The Marker's prefab that will be instantiated when the Object enables the marker for the first time. The Marker prefab needs the *Sprite Renderer* and the **Minimap Marker** components.

- **Minimap Manager –** Reference to the scene's *Minimap Manager*.

- **Should Follow –** If true, the Marker will follow the *Target Transform*'s position.

- **Enable On Awake –** If true, the Marker will ne instantiated as the game starts. If this *bool* is set as *false*, it is necessary to use the method *EnableMarker*() to instantiate the marker.

2.3. Minimap Marker

The **Minimap Marker** component is the marker itself. It's the object that will be visible on the minimap, utilizing a *Sprite Renderer*. It is controlled through the *Minimap Object* and *Minimap Manager* components.

2.4. Minimap Camera

The **Minimap Camera** component is used by the *Minimap Manager* to set the culling mask and the layers that must appear on the minimap. The component must be attached to a Camera, which will be used to render the objects that will appear on the minimap.

3. Setup Guide

I. Create a **Minimap Marker Prefab** and a **Render Texture**. The Prefab must contain a *Sprite Renderer* and *Minimap Marker* components.

II. Set the Marker Prefab’s layer. **It is recommended to create a “Minimap” layer for the Marker’s prefab to avoid errors.**

III. Add the **Minimap Camera** component to a Camera object. **Do not set add the Minimap Camera to your Main Camera: create a second Camera object to render the minimap.**

IV. On a new scene, attach a **Minimap Manager** component to an object.

V. On the **Minimap Manager**, set the **Camera Texture.** Select the render texture created previoiusly to be used to render the minimap.

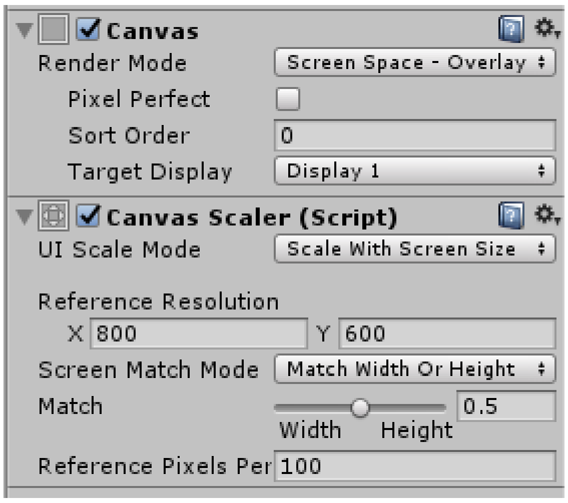
VI. Set the **Minimap Layer**. In this LayerMask, set the layers that will be rendered by the *Minimap Camera.*

VII. Set the **Minimap Camera** reference on the *Minimap Manager.*

VIII. Set if the Minimap Layer will be automatically disabled on the Main Camera. If set as true, all layers marked on the *Minimap Layer* will not be rendered by the main camera. **If you choose to set this *bool* as *true*, set the *MainCamera* reference on the Minimap Manager.**

IX. Set the sprites for each type. If a certain sprite type is not used (No enemies or allies), you can keep the sprite references as *null*.

X. Create a **Canvas**. This canvas will be used to render the Minimap. **Pay attention to the Canvas setup, to make sure the Minimap stays on the same position and size on all resolutions. Check the image below for an example of Canvas Setup:**


Canvas
Render Mode: Screen Space - Overlay
Pixel Perfect: ☐
Sort Order: 0
Target Display: Display 1

Canvas Scaler (Script)
UI Scale Mode: Scale With Screen Size
Reference Resolution: X 800 Y 600
Screen Match Mode: Match Width Or Height
Match: 0.5 (Width – Height)
Reference Pixels Per: 100

Fig. 03 – Minimap Canvas setup

XI. In the canvas, created a *Raw Image*. Set the previously created Render Texture as the Raw Image's *Texture,* as it you can see in the image below:

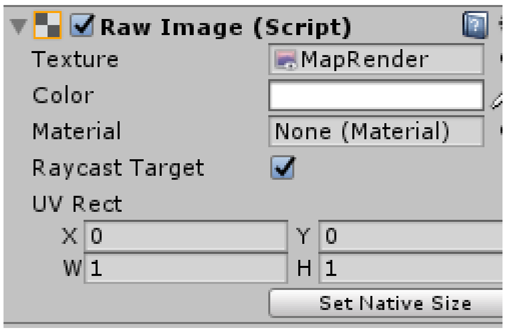

Raw Image (Script)
Texture: MapRender
Color:
Material: None (Material)
Raycast Target: ✓
UV Rect
X 0 Y 0
W 1 H 1

Fig. 04 – Raw Image in which the Minimap will be rendered

With this, the **Minimap Camera** will render all objects on the Raw Image of the canvas, creating a basic minimap for your game!

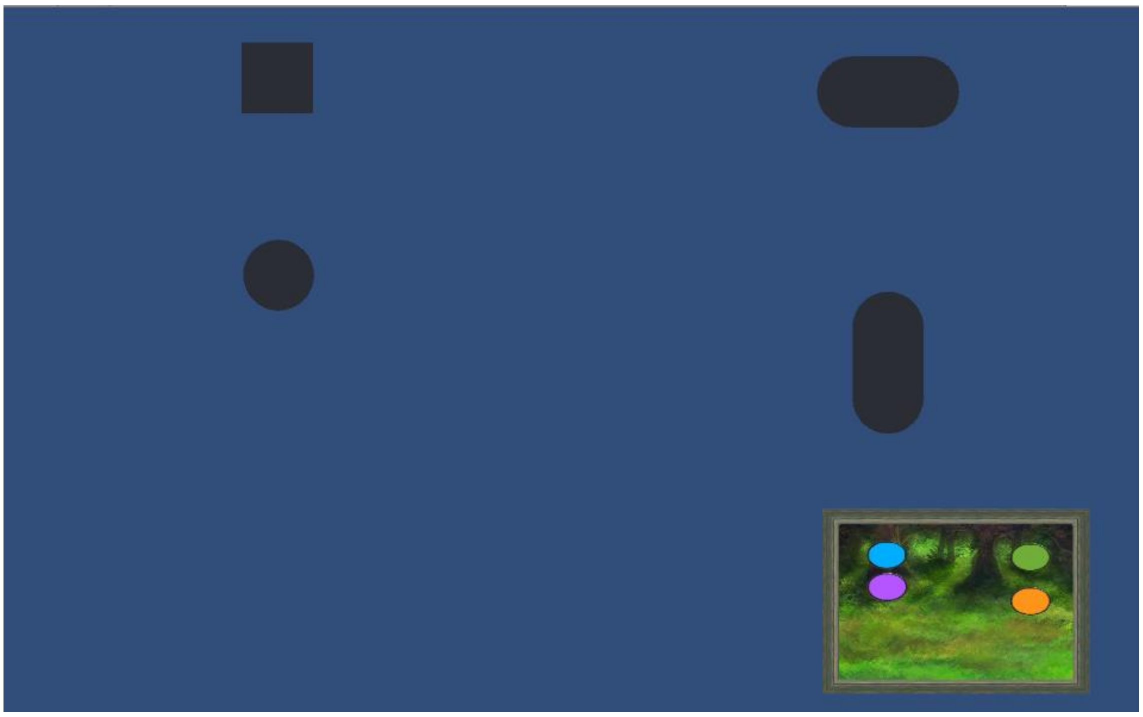


Fig. 05 – Basic scene with a Minimap Setup