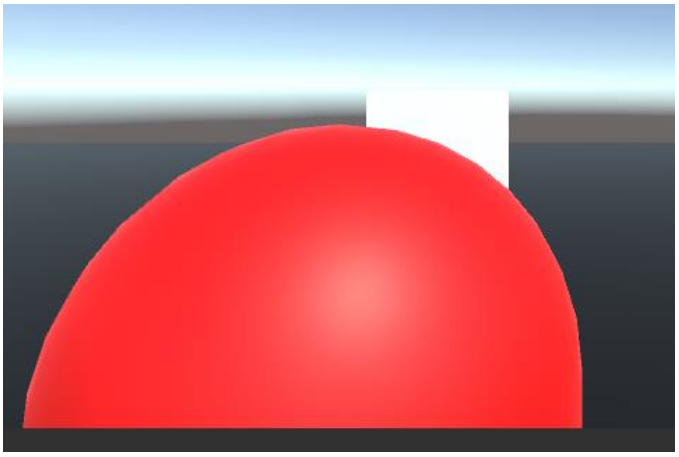
Minimap in Unity- two simple implementations

I'm using our simple scene setup from the past tutorials with some navmeshes and information setup. You can use anything, I'd recommend something you'd already created, it works best in 3D but the general gist works fine in 2D.

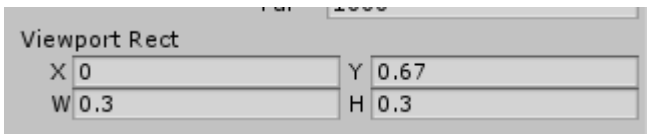


So, I've got a scene with an over the shoulder view of my player (the red capsule) for the main camera, there's nothing special about it, just set it up as you would normally. From here, you need to create a new camera. GameObject ->Camera from the top menu will do this.

We're going to make one camera act as the main view point and another as the view point of an "eye in the sky". Select the second camera, the one you've just created,

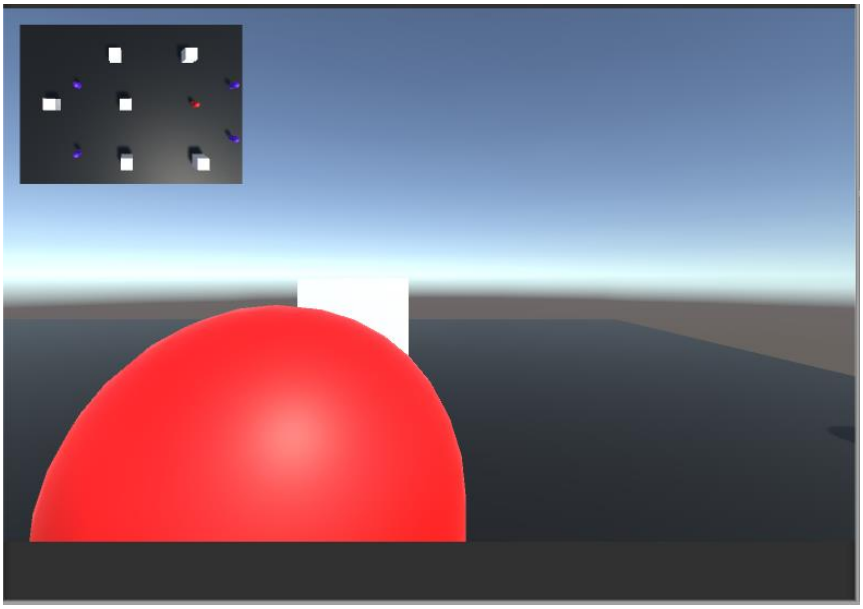
First let's get our camera looking down over the whole map, bring the camera up so it looks over the map and rotate it so it looks down. Something like the one to the left (which used 0,25,0 as position and 90,0,0 as rotation).

So now we have one camera that looks where we want it (in my case over the shoulder) and another that overlooks the whole map. For the minimap camera select and edit the Viewport Rect, this is the size and position of the camera's view in the world. We can use this to layout and show the camera view and resize and reposition it.



The viewport rect is from 0-1 where 1 is the whole width of the screen so 0.5 would be half the screen.

Experiment and play, the above settings gave me the below look, with the minimap in the top left.



Looks good but there's a few more things to change just to be sure, we set the depth of the camera to 1, this is the camera's position in the draw order. Cameras with a larger value will be drawn on top of cameras with a smaller value.

You set the clear flags setting on the minimap cam to depth only so it doesn't render a sky box or anything to cover the rest of the screen.

You remove the audio listener; a minimap doesn't need to listen for audio.

Give it a try. You could extend this further, drag it onto the player and zoom in a bit so it follows and gives an aerial view of the player. You can also assign a new layer to certain elements only rendering those by adjusting the culling mask.

You can also use the below script to track an object, just attach it to the camera and drag the target object onto the public field.

```
public GameObject target;

    void Update () {
        transform.position = new Vector3(target.transform.position.x, transform.position.y,
target.transform.position.z);
    }
```

Way 2 – Render Textures

Render textures are a cool thing, you probably know that textures are essentially pictures, representations of something. Well, render textures are images from the camera itself transferred as a texture onto objects. Bit of a rubbish explanation but it's cool, you'll see.

Let's start by creating a new render texture, in your project view right click -> create ->Render texture.

We'll tweak the camera we already have, restoring it back to the previous size in terms of the viewpoint rectangle. The settings are on the right.

I called my texture minimap and near the bottom of the box on the right you can see that I've dragged it into the Target Texture box. This means instead of rendering to the screen this camera is now rendering JUST to that texture.

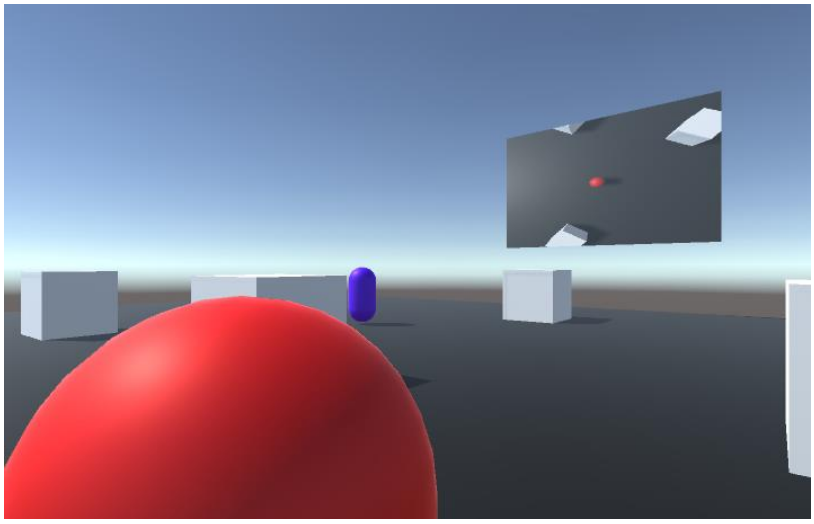
Add a canvas and a raw image to your project, positioning the raw image where you want your minimap to be. Then drag the texture you created into the Raw Image texture box.

The nice thing about this is you can easily add some styling, another image behind it to give a background, a frame, items around the map, any of your standard images or UI controls.

Give it a try.

What's the benefit of this? Well, if you create a plane, position it billboard style somewhere in your scene and drag the render texture onto it. You get a giant video billboard, can be used as all sorts of things, a giant kill screen, a giant spectator board. Use your brain.

There you go, short and sweet, a simple tutorial this time.



Challenges

Challenge 1

Add objects on other layers and make them showable only on one camera by adjusting the culling mask to decide what is being shown and what isn't

Challenge 2

Adjust your minimap so you can click somewhere and go to that point, this functions similar to an RTS and whatever you're moving would need a navmesh agent and script on it. Look at the navmesh tutorials and the RTS script.

Challenge 3

How could the camera effect be used to create a two player split screen game? Consider this and we'll come back to this later.