

Table of Contents

I. Converting Card .pdfs into .Pngs

II. Importing the Cards Into the Game

III. Creating a New Scene

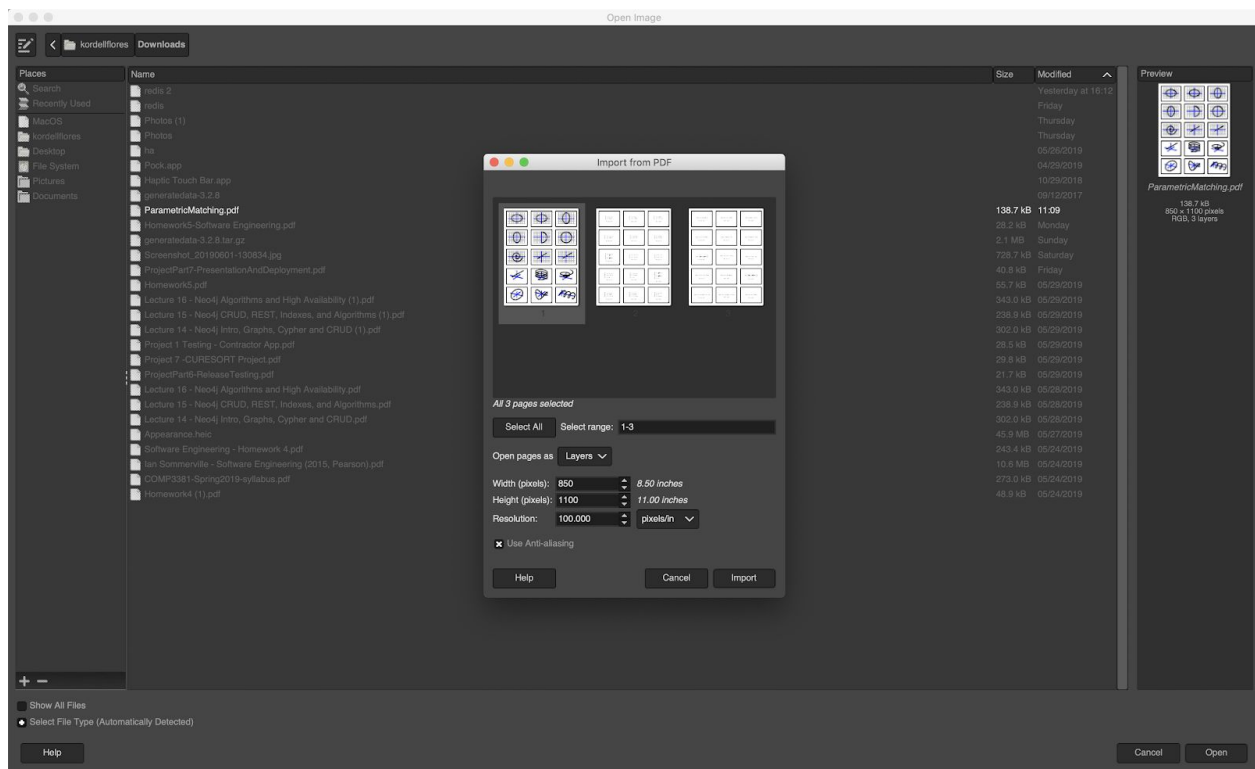
IV. Making a WebGL build

I. Converting Card .pdfs into .Pngs

You will need the following tools to make a new card set:

- Gimp (free)
- Unity (free)

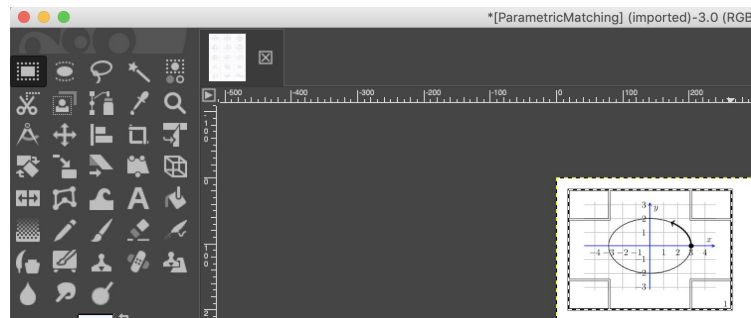
You will first need to get create a .pdf of the cards. The .pdfs we received held 3 pages, one page for each color of the cards (one page blue, one page green, one page red). Once you have that you will need to open Gimp and open the .pdf in Gimp. To do this, you will go to file -> open and find the .pdf file you have on your local machine. You should see the following screen:



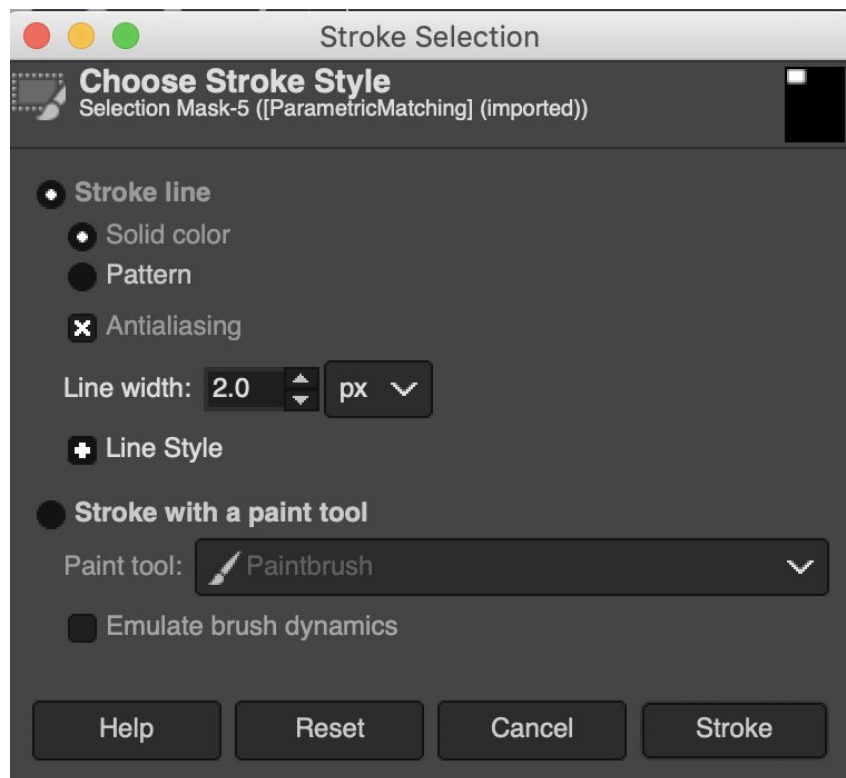
We only want to import one of the pages at one time so we will need to change the 'Selected Range' to page 1 and hit import.

You have now imported one of the card pages in the .pdf set of cards and can start getting rid of the transparency of the cards. To do this, you will first need to check if your cards have a border around them. We need this because Gimp will automatically separate parts of the image based on the lines. If your current image does not have borders, you will need to do the following steps(can skip if you already have borders):

-First, click the rectangle tool from the left side panel and draw a rectangle over one of the cards

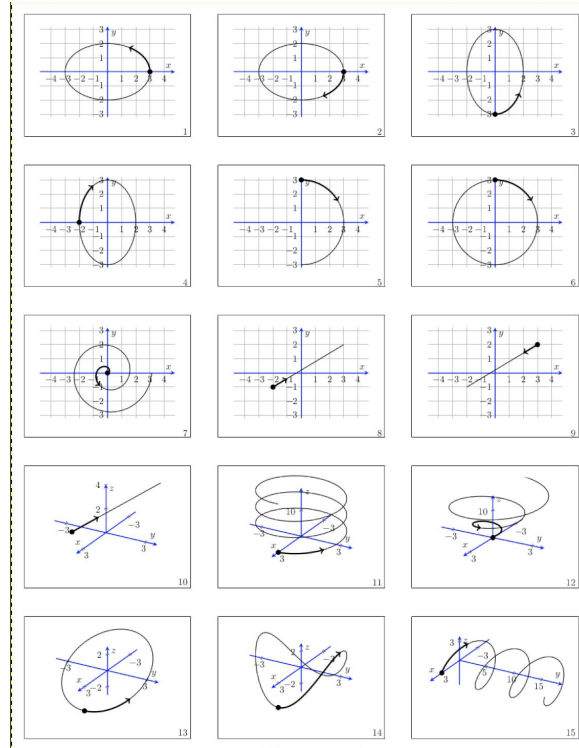


-Next, go to the edit->Stroke selection. You should see this screen:

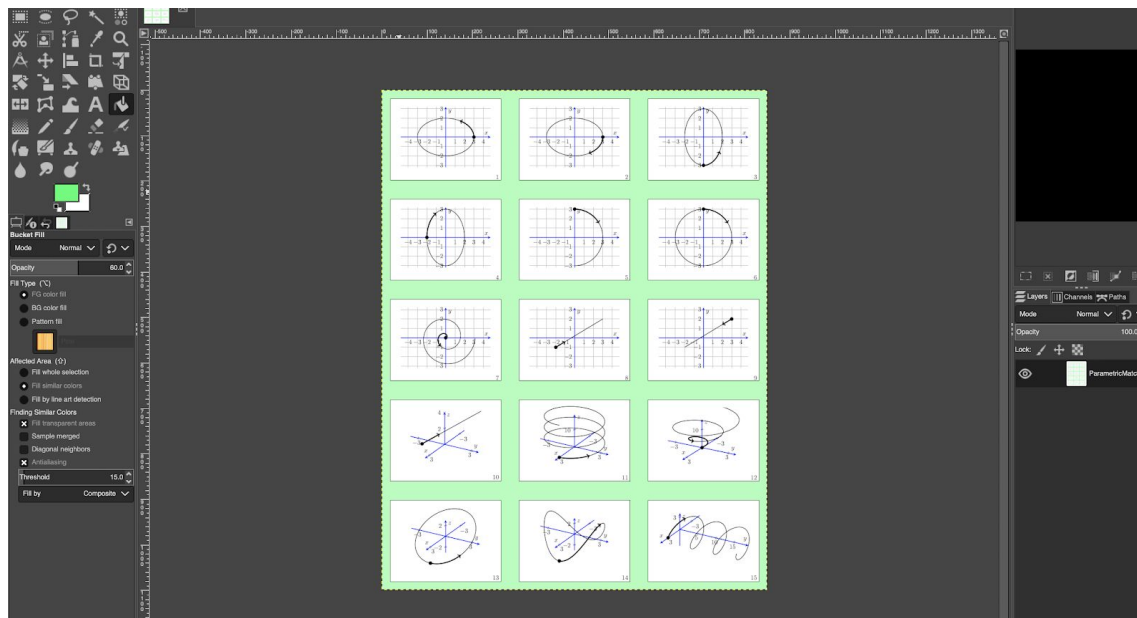


-You can change the line width to the width you would like then hit stroke. This will build a border around your card where the rectangle tool was.
-Repeat this for all the cards in your .pdf.

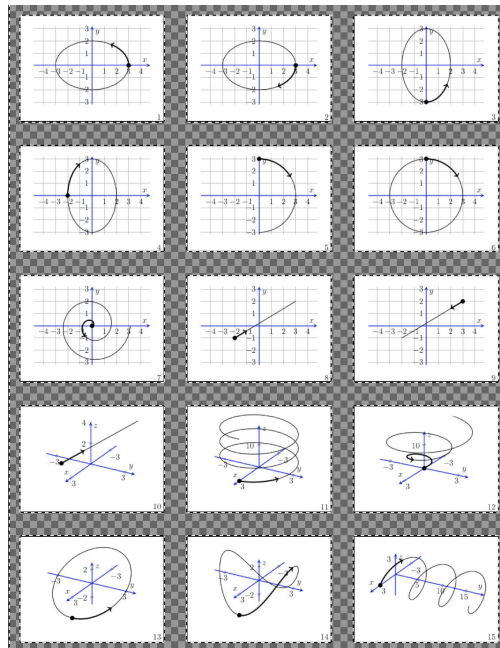
You should now have a .pdf that looks like the following:



Our next goal is to turn the background of the .pdf to a random color. We do this so we can select the random color and delete where that color is. In our case, we will select a green color and select outside of our card borders. You will need to select the paint bucket then change the color. Here is a screenshot of what your pdf should look like now:

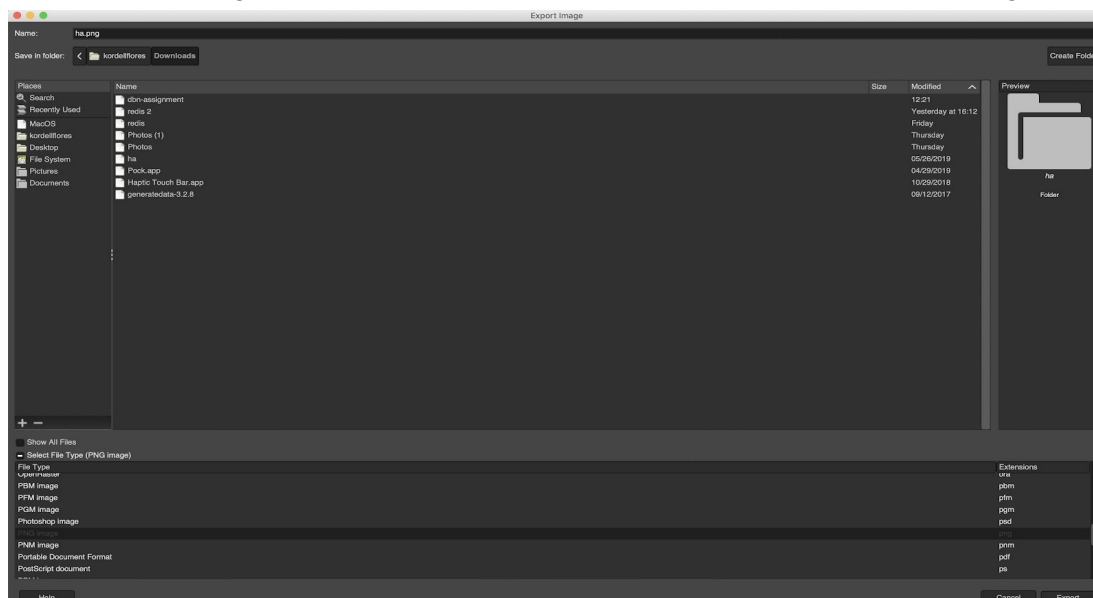


We can then separate the background from the cards and delete the backgrounds so we are only left with the cards. To do this, we will need to go to Select -> By Color and click the green background we made. Then click delete to delete the background leaving this image:



The final step in modifying our cards is to actually give the cards color. To do this, we again use the paint bucket, select the appropriate color, then select each card. Do keep in mind, that since lines separate portions of the card, you will need to fill in some whitespace.

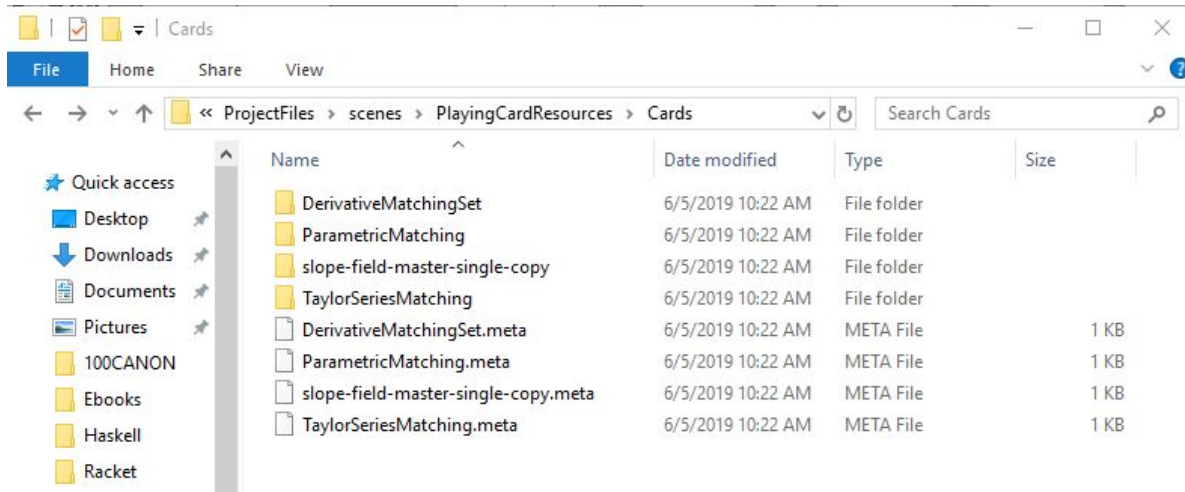
Lastly, you will need to export the files to .png files. You need to assure that they are .png because that format will keep the transparency. To export go to file -> Export As... and select a name and the .png format. You will now need to redo these steps for each page in each pdf.



II. Importing the Cards Into the Game

Adding new card sets is simple because of CalcMatch's card set standardization. To do so, you will need 3 png files - one with all the red cards on it, one with all the blue, and one with all the green.

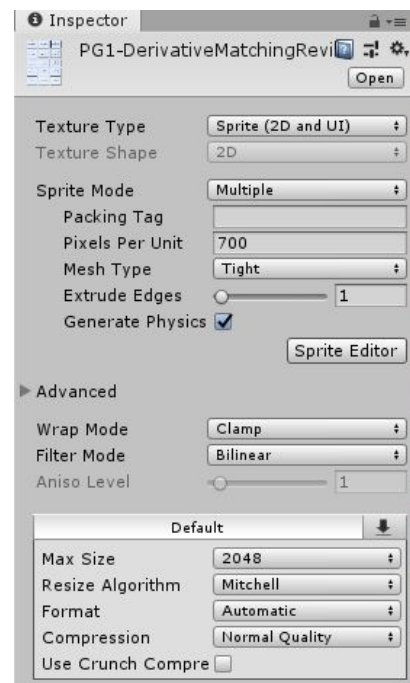
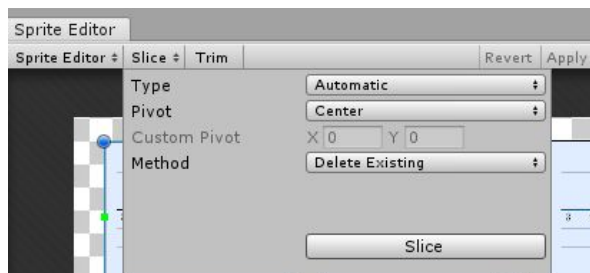
Once you have those files at the ready, navigate to `calcmatch/Assets/ProjectFiles/scenes/PlayingCardResources/Cards`, and create a folder with the desired name of the card sets. Put the three png files into the folder you created like so:



Our next step is to divide the three pngs into one png for each card (a lot of png's). Thankfully, Unity has a slicing tool which will help us do this trivially. To do so, open Unity and on the left hand side, navigate to

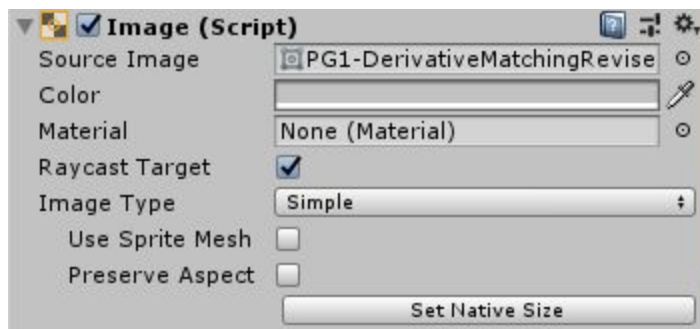
`Assets/ProjectFiles/scenes/PlayingCardResources/Cards/(the folder you created)`. Click on one of the png files in the folder and then make sure "Sprite Mode" is set to "Multiple", "Pixels Per Unit" is set to 700, and "Mesh Type" is set to "Full Rect".

Now select "Sprite Editor". Once in the sprite editor, press slice at the top, leave the default settings as they are, and then press slice. Once sliced, press Apply on the right hand side. The png has been separated into its individual cards.



Next, navigate to Assets/ProjectFiles/scenes/, and find DerivativeScene. Press control+D (or command+D on a mac) to duplicate the scene. Rename the duplicated scene to the desired name of your new scene.

Open the new scene. For more detailed information on how to create a new game scene refer to section 3. Navigate to BlueCanvas/BlueCardImage, and rename all of the buttons to have a different prefix. For example, “Der.Blue.3” could be changed to “Blah.Blue.3”, **as long as you use the same prefix for all cards and prefabs**. Once you decide on a prefix, stick to it. Make sure to keep the .Blue.3 part, it is necessary for the program to identify the cards. Once all the names have been changed, click on the first button (Example: Der.Blue.1). Under “Image (Script)”, change the source image to one of the new ones you just created after slicing. Make sure the first button corresponds to the first card, and so on. Repeat all of the steps in this paragraph for GreenCanvas/GreenCardImage and for RedCanvas/RedCardImage.



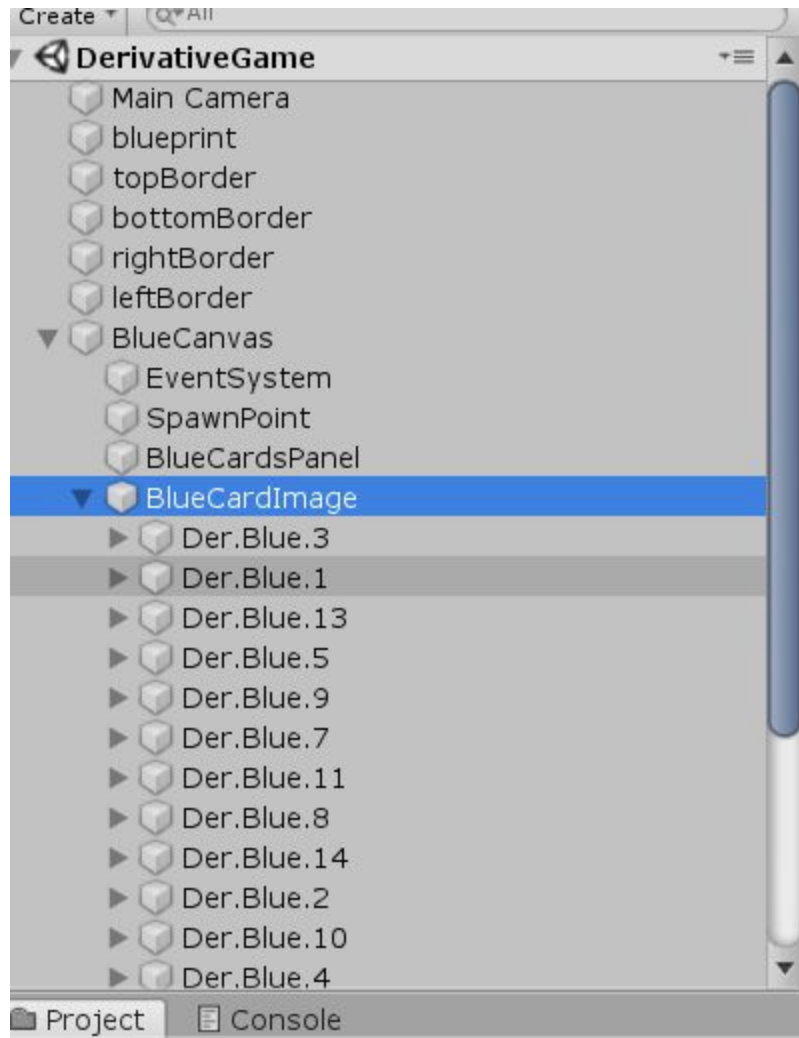
III. Creating a New Scene

In order to create a new game scene you will want to duplicate the DerivativeGame scene that exists in the unity project. This is easiest to do in the file browser.

DerivativeGame.unity	6/9/2019 5:08 PM	Unity scene file	446 KB
DerivativeGame.unity.meta	5/27/2019 4:46 PM	META File	1 KB
ParametricGame.unity	6/9/2019 5:08 PM	Unity scene file	446 KB
ParametricGame.unity.meta	5/28/2019 5:04 PM	META File	1 KB
PlayingCardResources.meta	5/27/2019 4:46 PM	META File	1 KB
prefabs.meta	5/27/2019 4:46 PM	META File	1 KB
scripts.meta	6/1/2019 2:36 PM	META File	1 KB
SlopeFieldGame.unity	6/9/2019 5:08 PM	Unity scene file	324 KB
SlopeFieldGame.unity.meta	5/28/2019 5:04 PM	META File	1 KB
TaylorSeriesGame.unity	6/9/2019 5:08 PM	Unity scene file	373 KB
TaylorSeriesGame.unity.meta	5/28/2019 5:04 PM	META File	1 KB

Simply copy the DerivativeGame.unity file and rename the duplicate to your desired game title. The metadata file will be generated for you. This directory can be found at:
...\\calcmatch\\CalcMatch\\Assets\\ProjectFiles\\scenes

Refer to the directions in section three for how to change the buttons in the scene to show your new cards visually. Note that every scene has a hierarchy of objects that can be displayed on the left side of the screen in most unity editors. It will look something like the following.

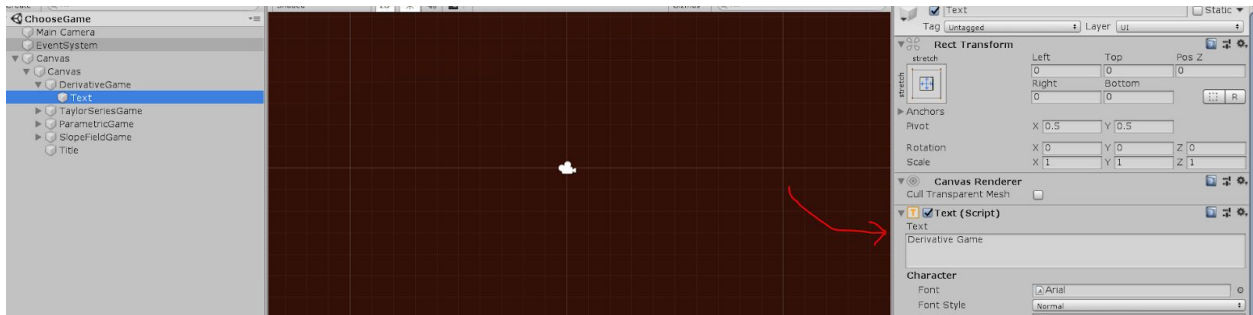


Pay particular attention to the names of the button objects under BlueCardImage, GreenCardImage, and PinkCardImage. These names must match the naming convention used when extracting the cards and saving them to the Resources folder as described in earlier sections. All of these must be changed individually.

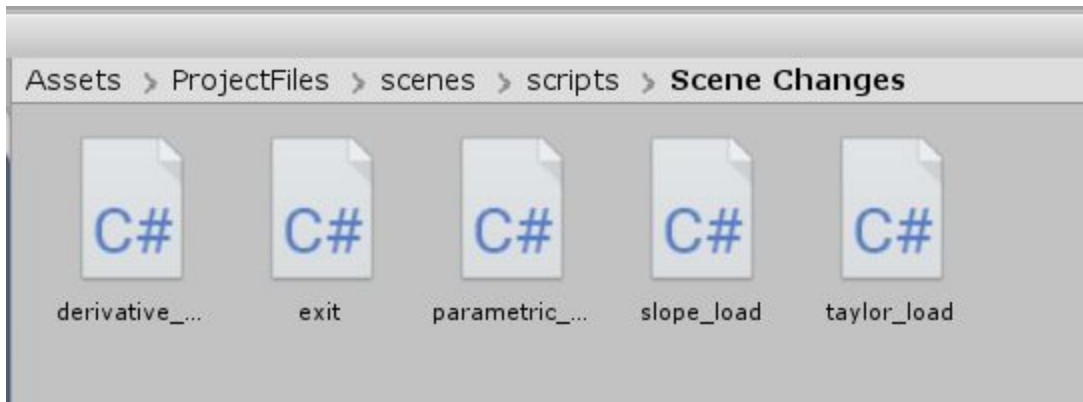
After you have duplicated the scene, renamed it, renamed all the buttons, and put the new card images over the card objects, you are almost done. The last thing is to add a way for players to access this new game scene. In Unity, open the ChooseGame Scene.



Copy the DerivativeGame object and rename it to whatever the new game is. Note that the object can be expanded and has a text object. You will want to open that and change the text to the name of your game.



After creating this new button and positioning it on the canvas where you would like, the last step is to link the button into the game flow. Open the following in the unity editor.



Create a new script that will be used to transition players to your new game scene. The contents may be copy-pasted from the derivative game scene with one exception. You will need to specify the name of your new scene in the call to:

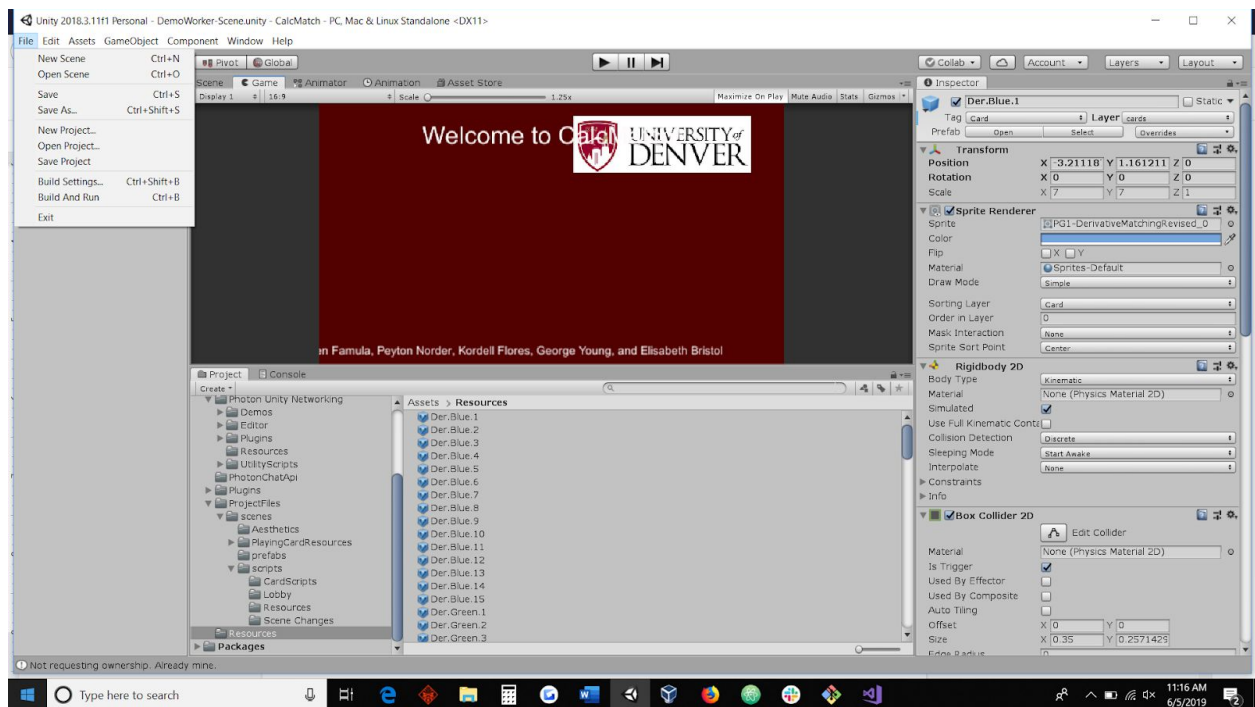
```
SceneManager.LoadScene("YourNewSceneNameHere");
```

Afterward, clicking your new button created above should take a player to the new game scene.

IV. Making a WebGL build

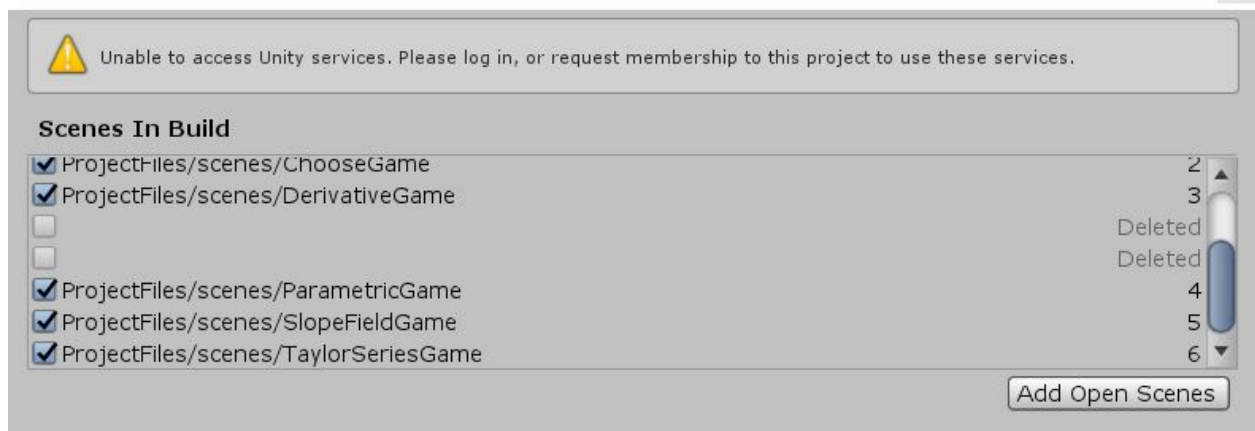
Once the project is ready to be built for distribution the following steps need to be taken.

First go to File in the Unity editor and select Build Settings

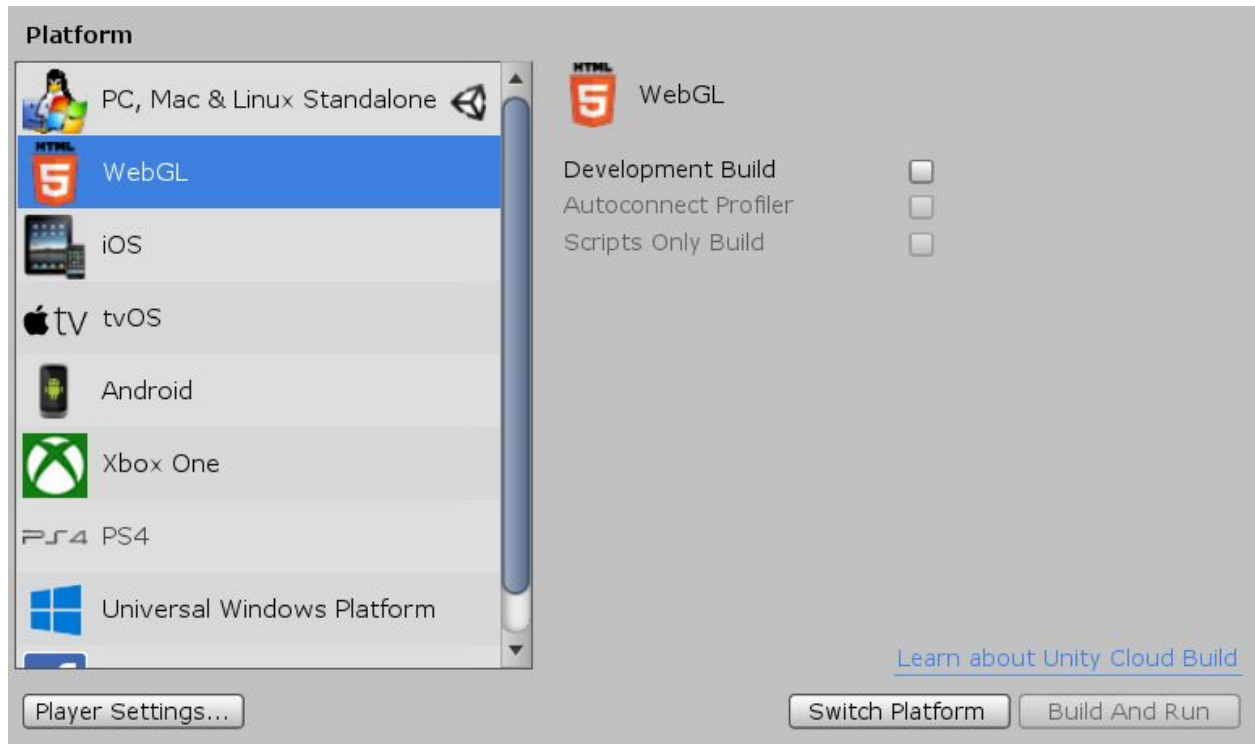


Once inside the Build Settings make sure all of the scenes you have created are in the current build, if the scene is not in the build go out of the editor select the scene to be added and then repeat the first step, once there click Add Open Scene and repeat this process until all desired scenes are added to the Build

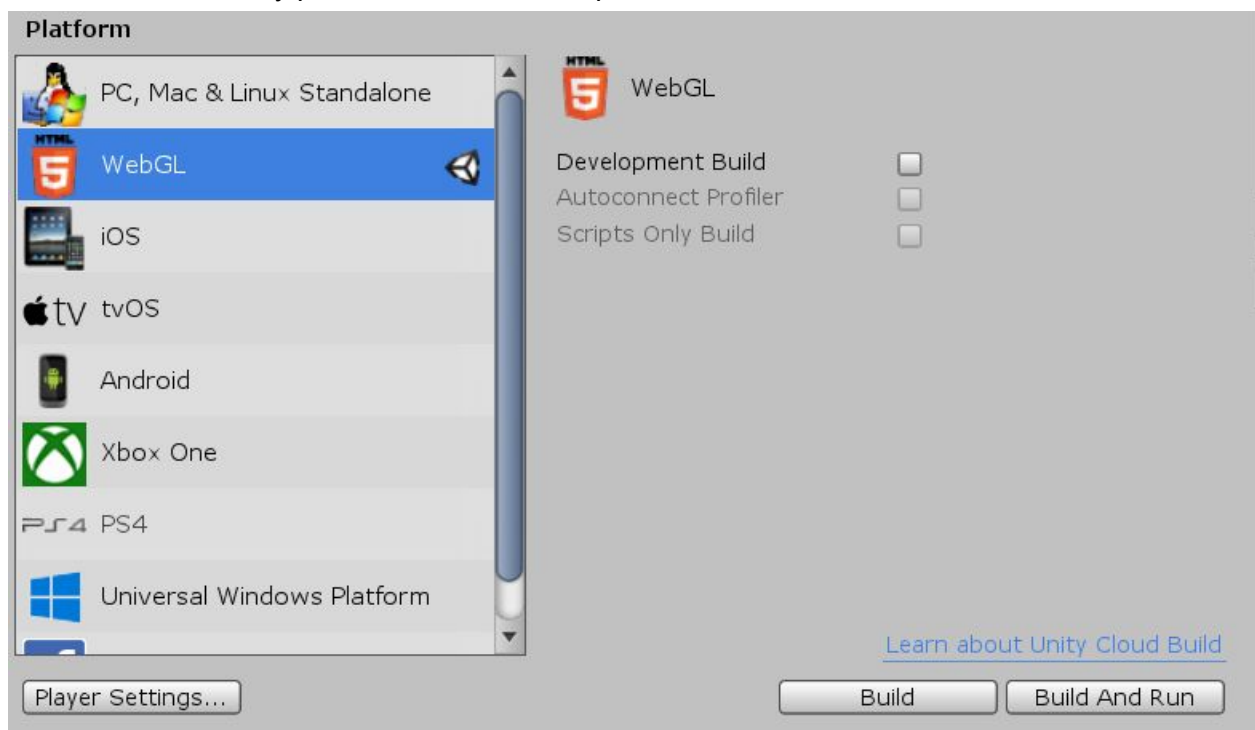
Build Settings



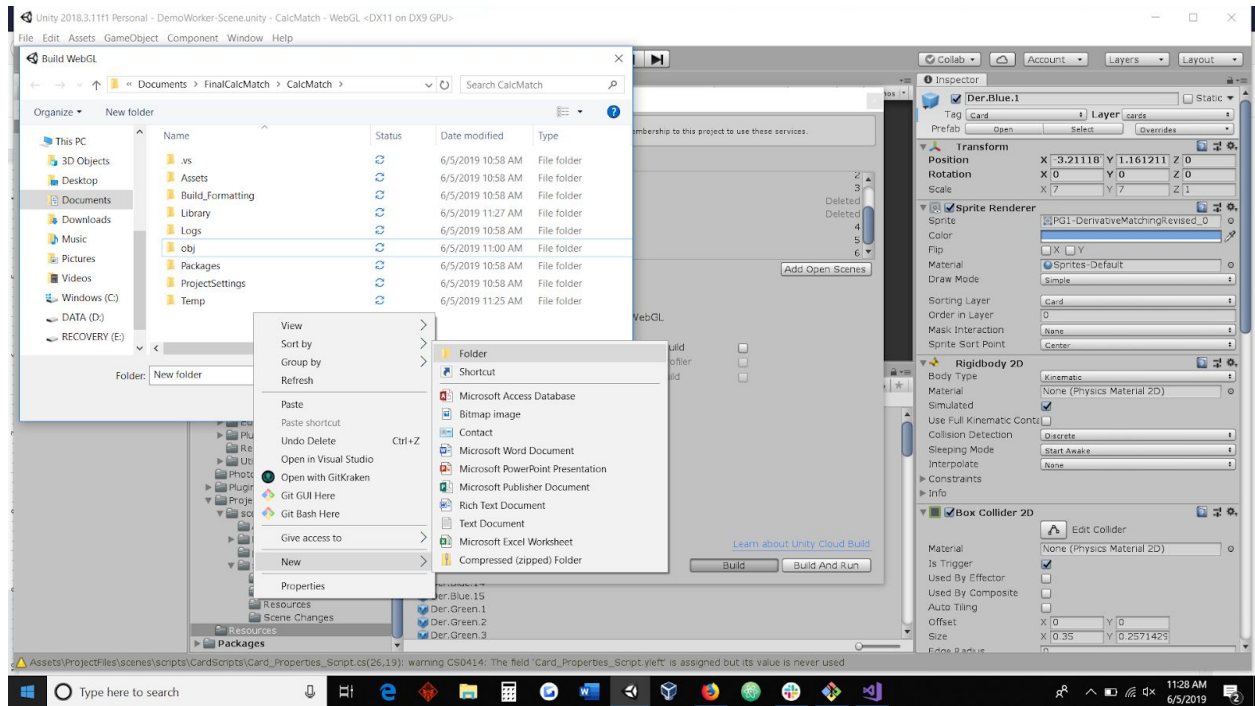
When all the the scenes are in the Build you are ready to select the Build type and actually build the project



Select WebGL as the Platform and Press Switch Platform, if the option to Switch Platform is not available do not worry proceed to the next step

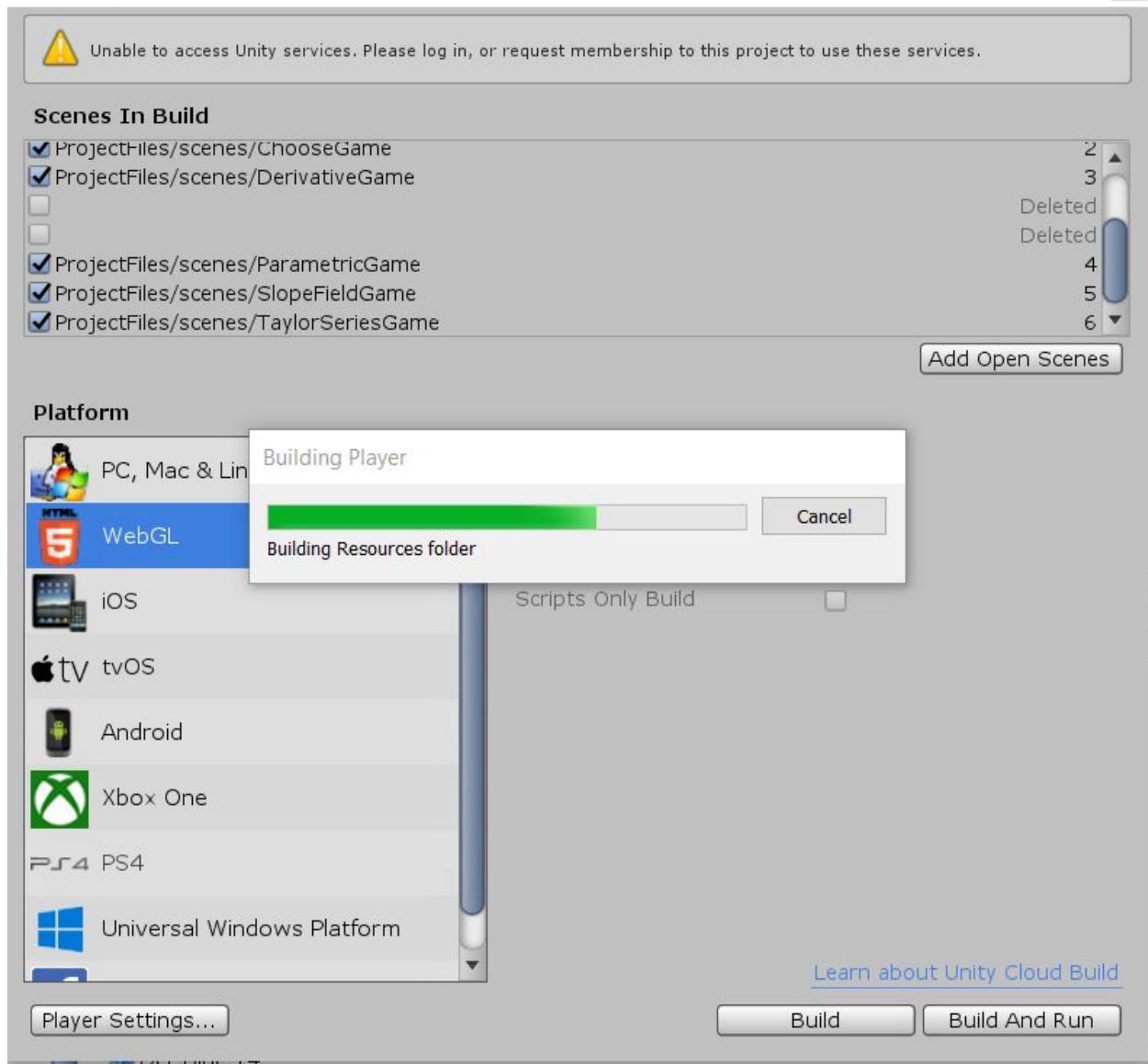


Once the platform is switched to WebGL click Build





























If there is a build folder select it and press select folder, if not create a new folder and name it build, naming is important so make sure the folder is named build.

Build Settings



This popup should show and will take a significant amount of time to complete around 5-10 minutes depending on your computer, at some point during the build you may not be able to open Unity or do anything inside or Unity, do not worry this is expected behavior.

Once the build is done a few files in the build need to be changed.

	.vs		6/5/2019 10:58 AM	File folder	
	Assets		6/5/2019 10:58 AM	File folder	
	build		6/5/2019 11:31 AM	File folder	
	Build_Formatting		6/5/2019 10:58 AM	File folder	
	Library		6/5/2019 11:32 AM	File folder	
	Logs		6/5/2019 10:58 AM	File folder	
	obj		6/5/2019 11:00 AM	File folder	
	Packages		6/5/2019 10:58 AM	File folder	
	ProjectSettings		6/5/2019 11:31 AM	File folder	
	Temp		6/5/2019 11:36 AM	File folder	
	Assembly-CSharp		6/5/2019 11:25 AM	Visual C# Project fi...	57 KB
	Assembly-CSharp-Editor		6/5/2019 11:25 AM	Visual C# Project fi...	46 KB
	Assembly-CSharp-firstpass		6/5/2019 11:25 AM	Visual C# Project fi...	41 KB

Click inside the Build_formatting folder.

	index		6/5/2019 10:58 AM	Firefox HTML Doc...	1 KB
	style		6/5/2019 10:58 AM	CSS File	2 KB

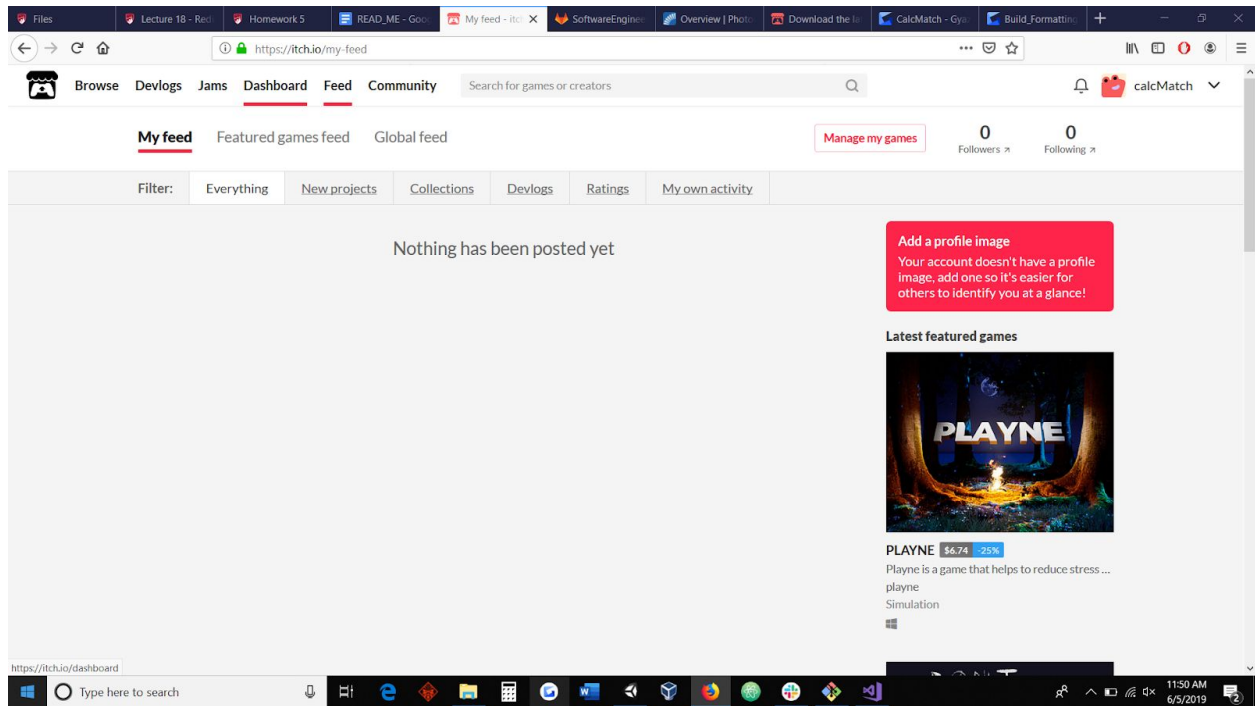
The index file needs to be put into the build folder replacing the file with the same name already in the file. The style file will go in the TemplateData folder, replacing the file that already exists. Once this is done you will zip the build folder in order to upload the game to itch.io

Go on <https://itch.io> and press the login button the credentials are as follows:

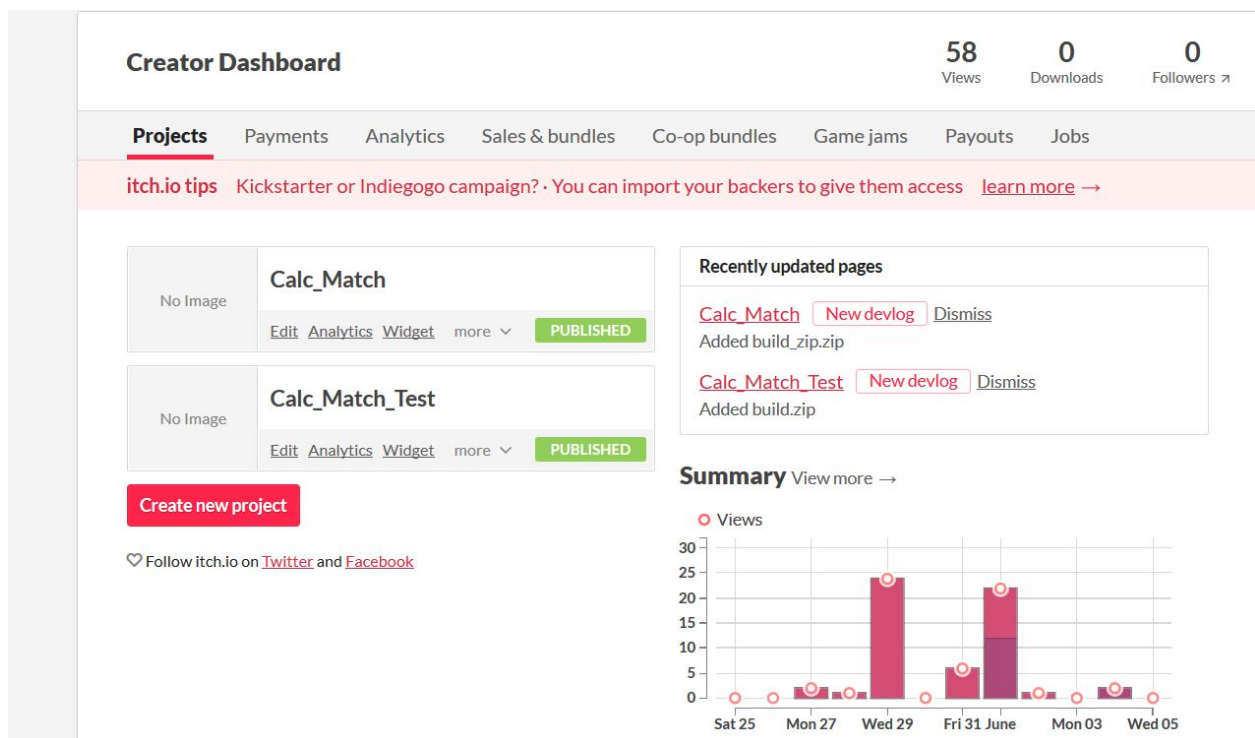
Username: calcmatch@gmail.com

Password: DU2019calc

Once logged in select the Dashboard option at the top of the screen



You should be given a screen like the one below with the option to create a new game or edit an existing one, select either option depending on your goal. Generally we created a test game to test changes which allowed for the old game to stay.



Add a title to your game and go down to the settings below and select HTML as the kind of project

Classification

What are you uploading?

Games — A piece of software you can play ▼

Kind of project

HTML — You have a ZIP or HTML file that will be played in the browser ▼

TIP *You can add additional downloadable files for any of the types above*

Release status

Released — Project is complete, but might receive some updates ▼

Go to the Uploads section and upload the zip file of the build, then change the embed options to click to launch in fullscreen. The embed settings can be changed but the current html and style files used earlier are optimized for the settings described above.

Uploads

Upload a ZIP file containing your game. There must be an `index.html` file in the ZIP.
Or upload a `.html` file that contains your entire game. [Learn more](#) →

Any additional files you upload will be made available for download. You can apply a minimum price to the project after uploading additional downloadable files.

TIP Use **butler** to upload game files: it only uploads what's changed, generates patches for the [itch.io app](#), and you can automate it. [Get started!](#)

Upload files

or

 Choose from Dropbox

[Add External file](#) ?

File size limit: 1 GB. [Contact us](#) if you need more space

Embed options

How should your project be run in your page?

Click to launch in fullscreen ▼

Frame options

- ☐ Mobile friendly — Your project can run on mobile phones (smaller resolution and touch support)
- ☐ Enable scrollbars — Enable scrollbars in the iframe that contains your project

The visibility settings can be changed depending on how public you want the game to be. Using restricted requires users to have the url of the game exactly and to have a password, we used this when developing the game. Once the game is deemed acceptable to be open to the public the Public setting was used, this allows for anyone to view the game and the game to come up when a search engine lookup is done.

Community

Build a community for your project by letting people post to your page.

- ☐ Disabled
- ☒ Comments — Add a nested comment thread to the bottom of the project page
- ☐ Discussion board — Add a dedicated community page with categories, threads, replies & more

Visibility & access

Use Draft to review your page before making it public. [Learn more about access modes](#)

- ☐ Draft — Only those who can edit the project can view the page
- ☒ Restricted — Only owners & authorized people can view the page
- ☐ Public — Anyone can view the page, **you can enable this after you've saved**

Restricted access settings

Only people who own the project can view the page. You can give access by generating a download key. The page will be unlisted in browse and search.

☐ Also allow a password to view page [?](#)

Save & view page

After that click Save and View Page and you have successfully uploaded a WebGL build to the internet.