TexturePacker User Guide

Table of Contents

Introduction

This guide is for anyone who needs some help on using TexturePacker. In this guide I will show you how to export animation from Photoshop and put them into TexturePacker step by step.

I would like to thank Raven Mortimer, a wonderful artist and project partner who inspired me to write this guide. I also wanted to thank Matthew Doucette, Sean Morrow and Sam Robichaud – my instructor at NSCC Truro – for their advices and mentorship while I was writing this guide.

If you, readers, would like to give me any feedback on this guide, feel free to contact me at

Buu.nguyenquoc@gmail. You can also look for the newest updates of this guide on https://www.seromyr.com/download-archive

Getting Started

1. Download and install Adobe Photoshop

Get it at https://www.adobe.com/ca

Please note that the Render Video function mentioned in this guide will not working properly on Photoshop CC version 20.0.0 due to software's error. This issue has been fixed on version 20.0.1. It is my recommendation to use the newest version.

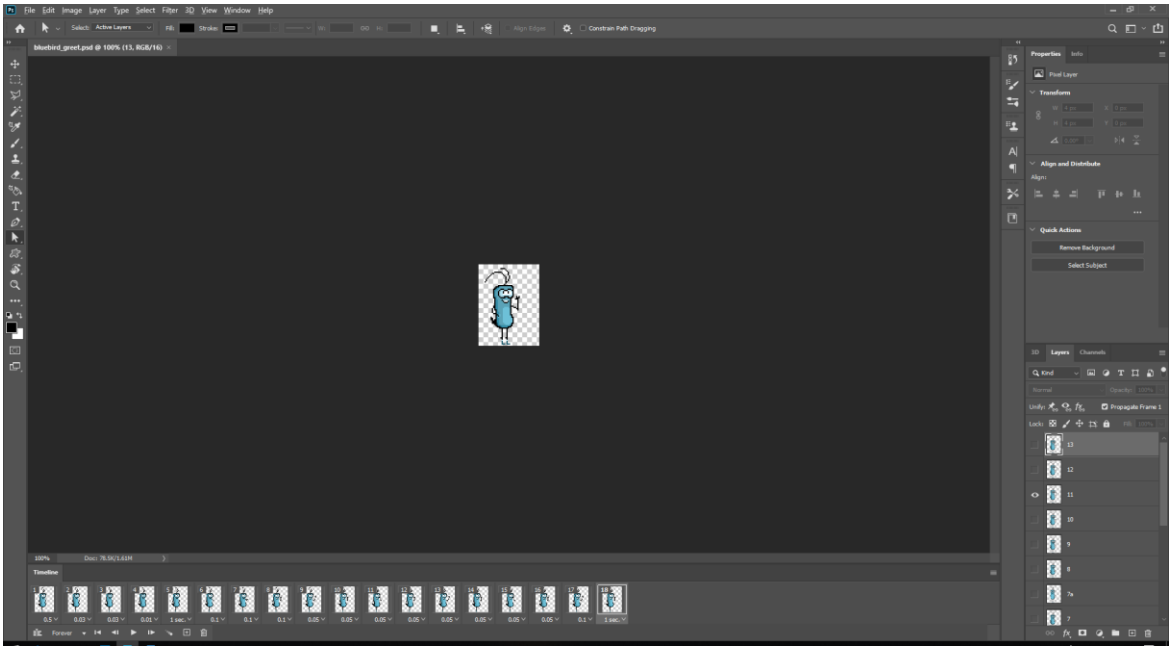
In this guide, I used Photoshop version 21.1.2.

2. Download and install TexturePacker

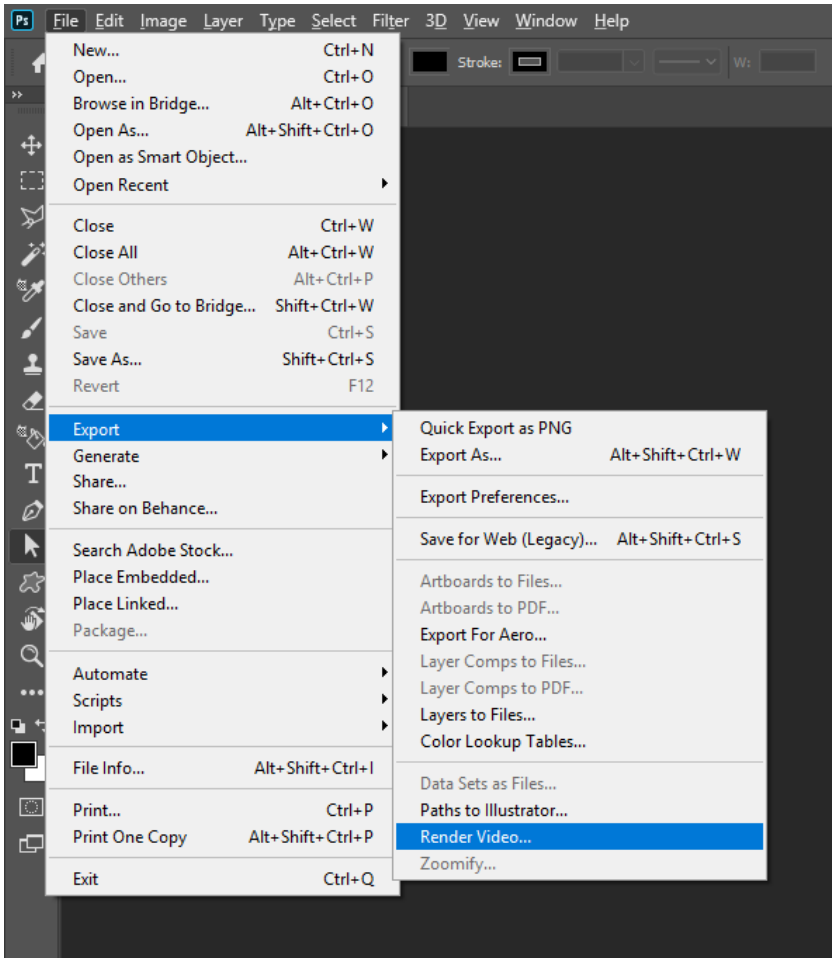
Get it at https://www.codeandweb.com/texturepacker

Export Animation from Photoshop

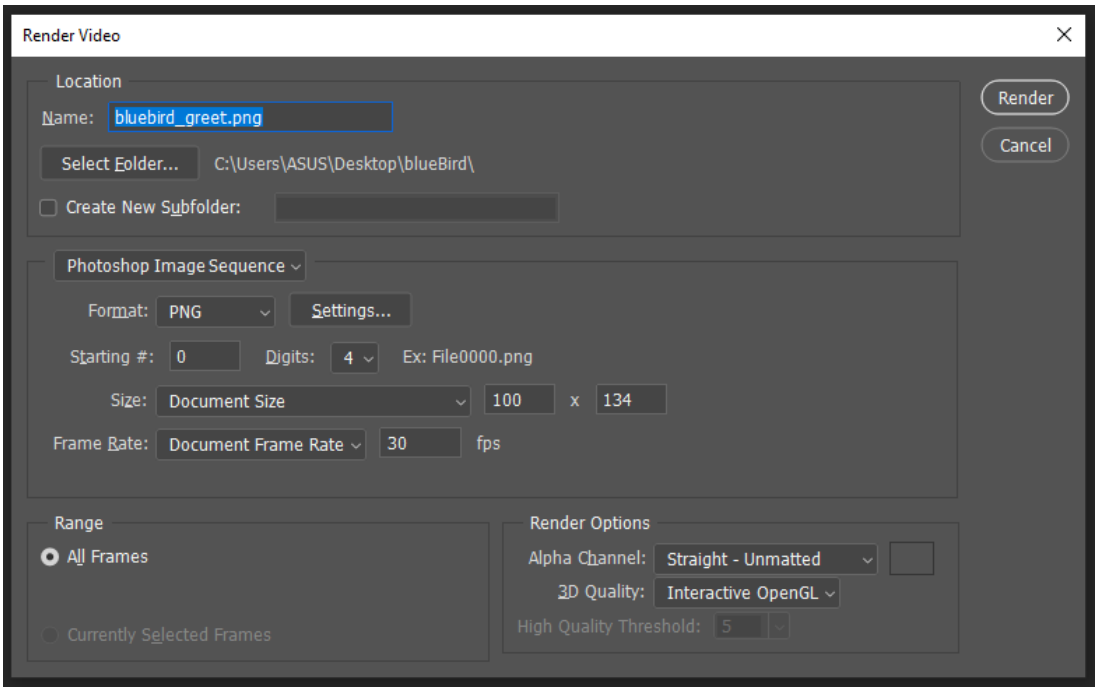
1. Open the animation file in Photoshop



2. Go to File > Export > Render Video



3. A Render Video window will pop up



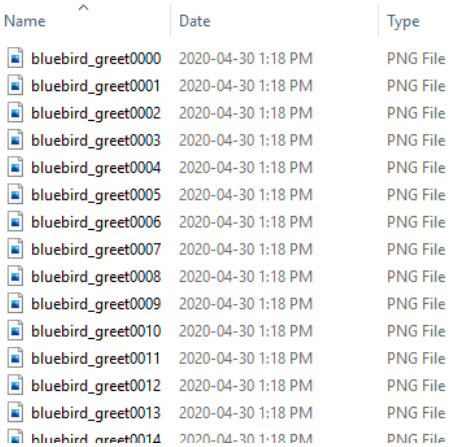
From top to bottom:

Name

- The output name suffix of the exporting files if your animation had more than 1 frame.

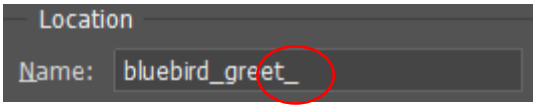
- Look at the name I had in the example: *bluebird_greet.png*
  "*bluebird*" is the name of the character.
  "*greet*" is the action of the current animation. In our project, it is likely that one character would have multiple actions.

- If I left as it is, Photoshop will export it to multiple files as shown in this image below:

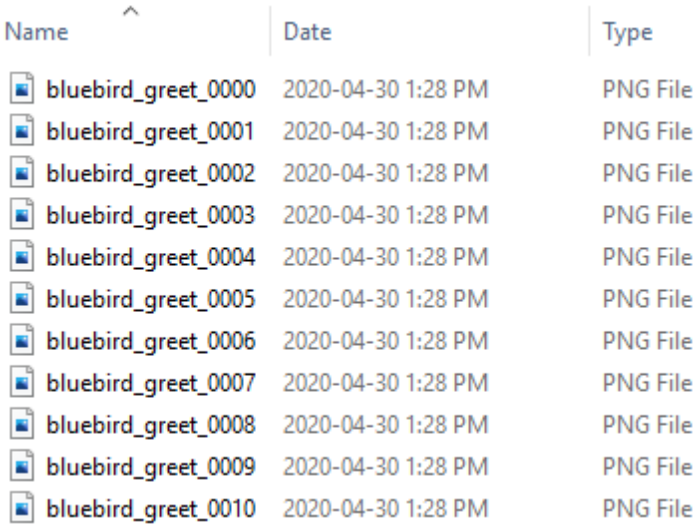


  You can see that it put 4-digit number right next to the name. That is okay but not quite well organized.

I suggest you put an extra underscore next to the file name which should look like this:



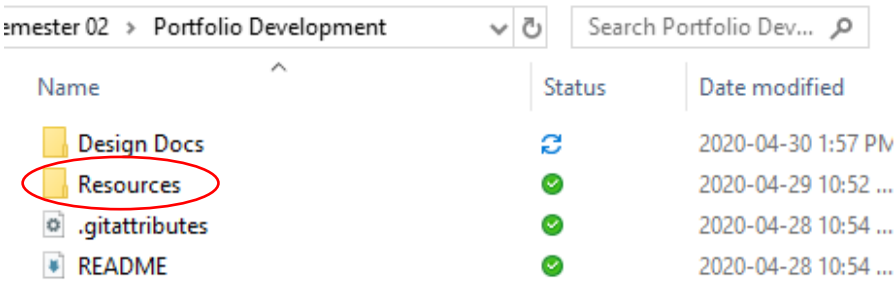
It would separate the name and the number, which is very easy to read and keep track.



This looks better, don't you agree?

Select Folder

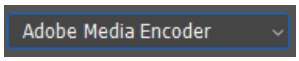
- This button allows you to change the location for exported files.
- I recommended you put all your exported animation in Resources folder, which where programmer would look for data implementation.



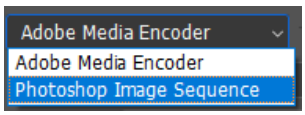
Create New Subfolder checkbox

- This button allows you to create a new subfolder and put all the exported files inside it.
- I recommend you create a subfolder for every animation of the same character. It is a tidier way to organize and would not accidentally override your other animation data during the process.
- Just simply check the checkbox and type in the name of the folder of your choice.

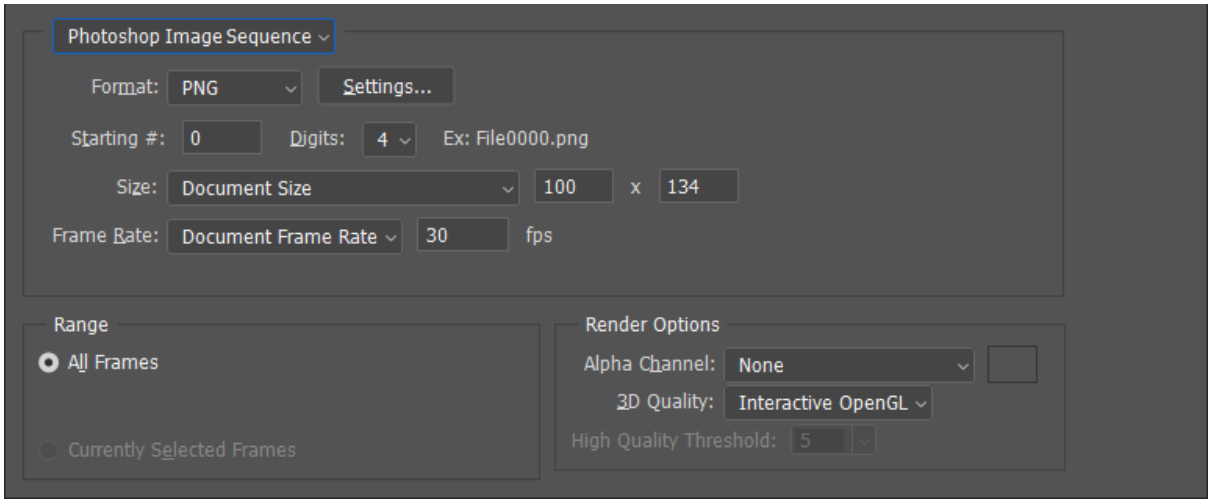
Encoder



- Photoshop probably uses this encoding method by default the first time you opened it. This is not what we are going to use.
- Click on it and select Photoshop Image Sequence.

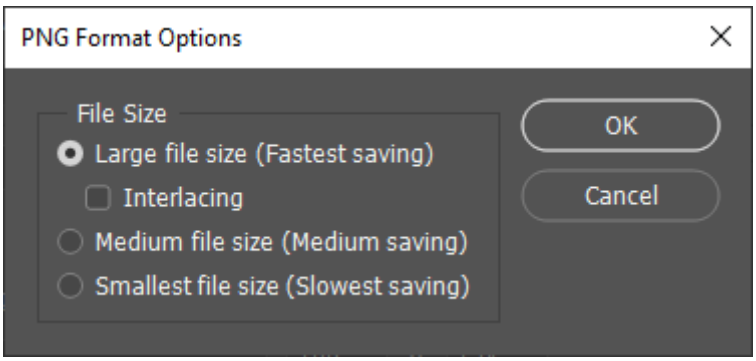


- You will see new settings for this encoding method. We will make some changes here.

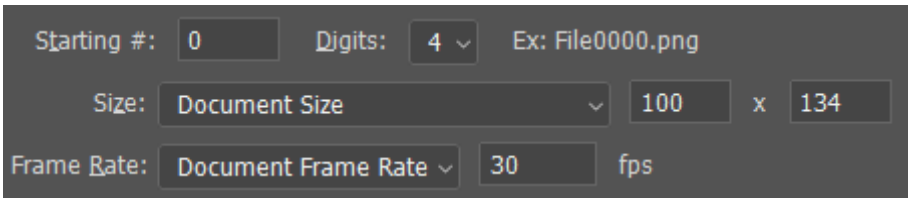


File format

- Select PNG
- In the Settings button next to it, choose Large file size to keep the best image quality.



Other formatting options

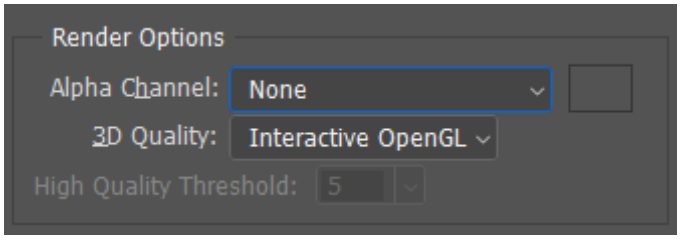


- Starting #: This affect the starting number in the naming prefix of the exported files. Just leave it be.
- Digits: As you have seen in the first demonstration above, the exported files have 4-digit number. If you were certain that your animation would never exceed 100 frames, you could change it down to 2-digit. I personally prefer 3-digit. It is your choice.
- Size: Just leave it be, it will use the dimensions of the document canvas.
- Frame Rate: Just leave it be, our game and most of other games out there are using 30 fps.

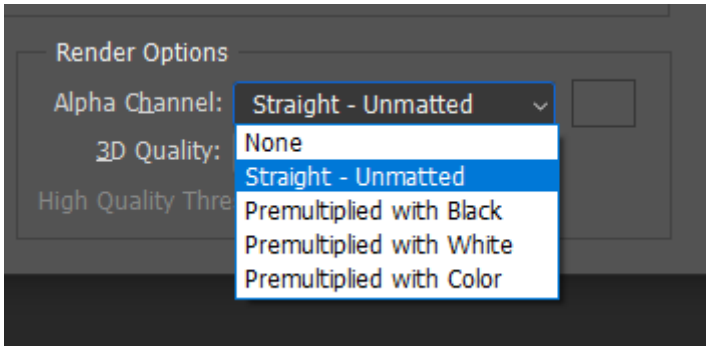
Range

- Do not touch it.

Render Options



- Alpha Chanel: Default is None, change it to Straight – Unmatted. It would allow exported images have transparent background.



- 3D Quality: Just leave it be.

Congratulations! You can hit Render and switch to TexturePacker now.

4. Summary

A few things to remember before hitting **Render** button.

1. Place an extra underscore at the end of the file name (if there was not any).
2. Export animation into Resources folder of the common working folders.
3. Use separate folder for each animation.
4. Make sure encoding method is Photoshop Image Sequence.
5. Format type is PNG with Large File Size for highest image quality.
6. Make sure the Alpha Chanel uses Straight – Unmatted for transparent background.

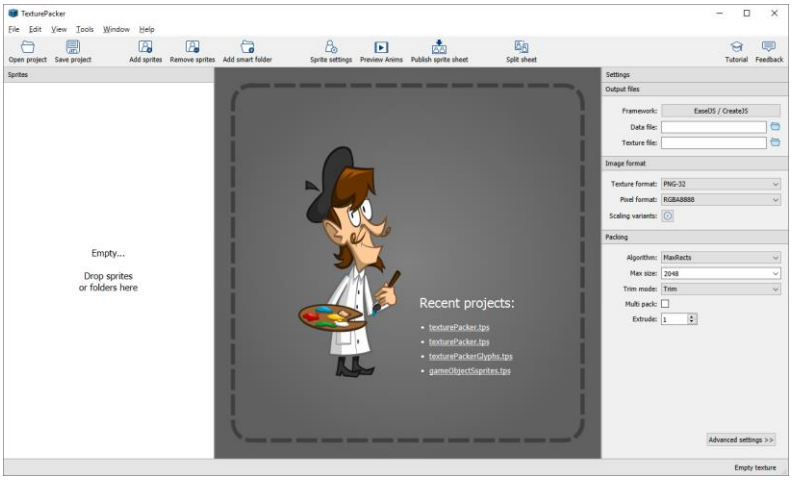
Working with TexturePacker

1. Creating a static asset

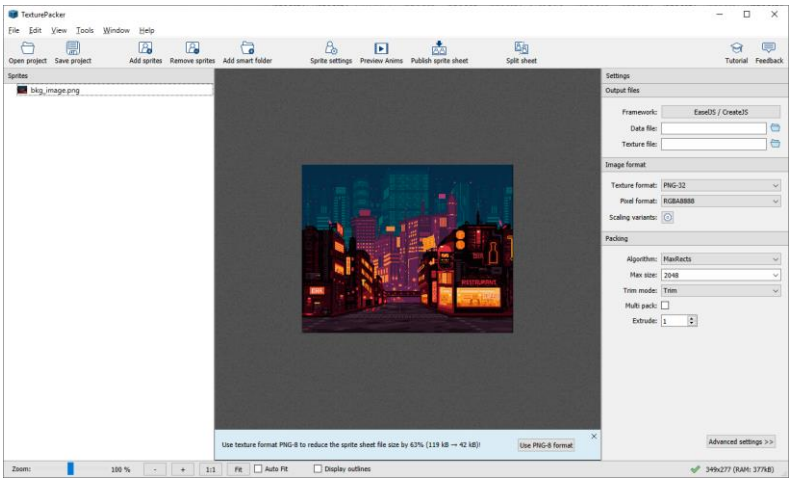
Static assets are sprites that has no animation, like background images, buttons, banners, splash arts.

In this example, I'm going to use a splash image taken from here. Thank you Ansimuz for this amazing piece of art.

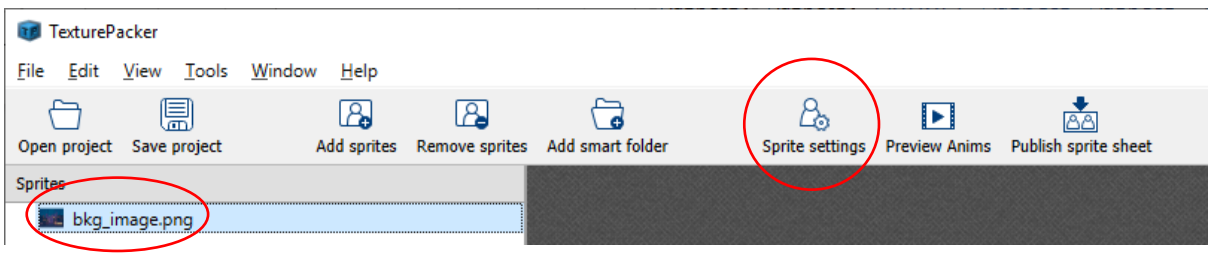
- Open TexturePacker.



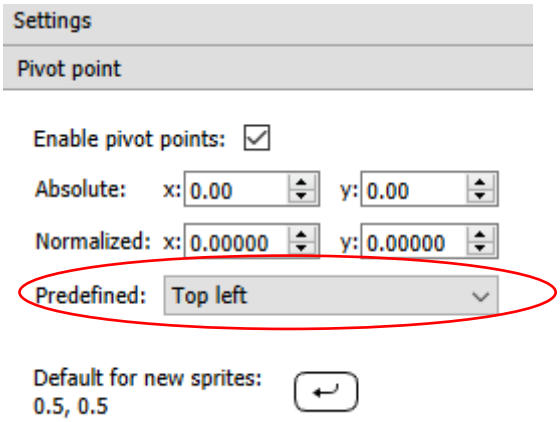
- Drag the image or image folder into TexturePacker.



- Select the image and click on Sprite Settings to change it Pivot Point (very important).



- By default, the pivot is at the center of the sprite. Change it to Top Left.

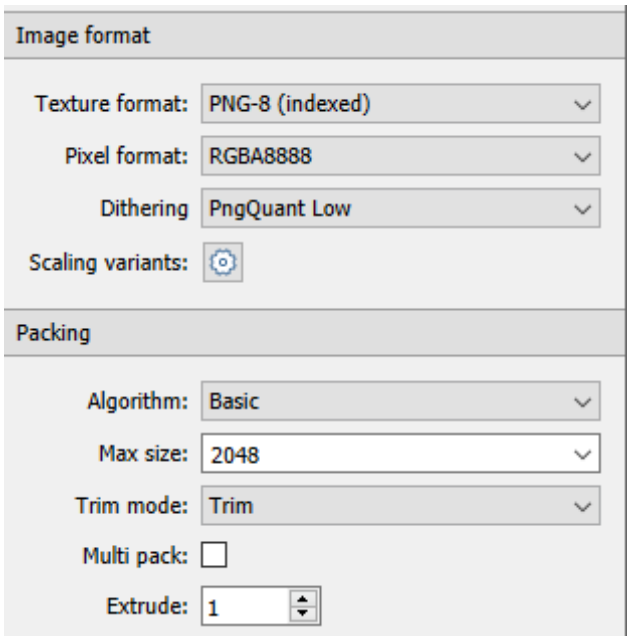


All the static assets' pivot should be Top Left, unless stated otherwise by programmer.

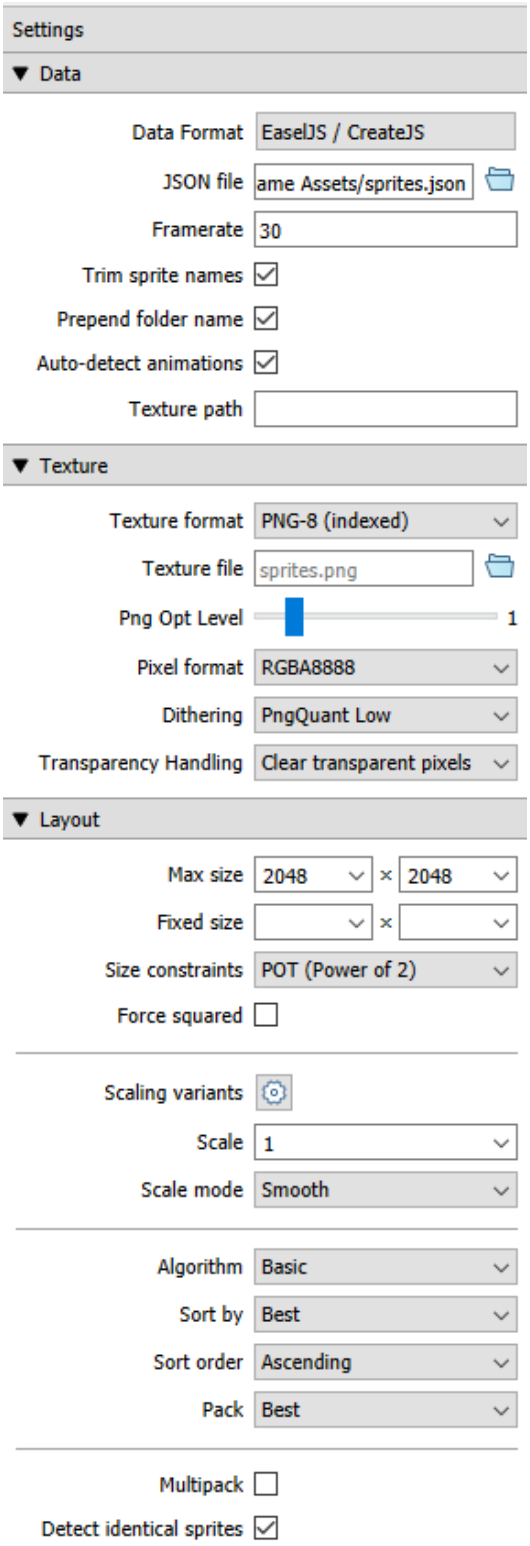
- Hit Back to sheet to make changes to other settings

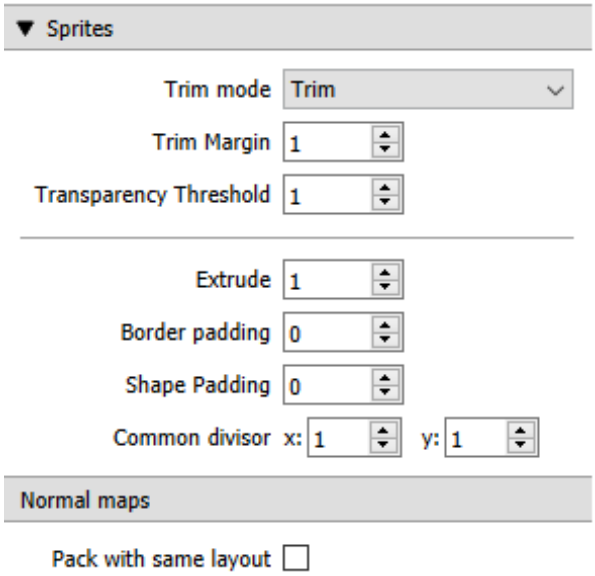
Now Look at the Settings Panel on the right side

- Change Framework to EaselJS / CreateJS (or anything else depend on your project).
- Set the output data file path and file name. I would recommend you put it inside the Resources folder. Save as type JSON file. TexturePacker would also create a new image file for this JSON file.
- Other settings should look like this image below

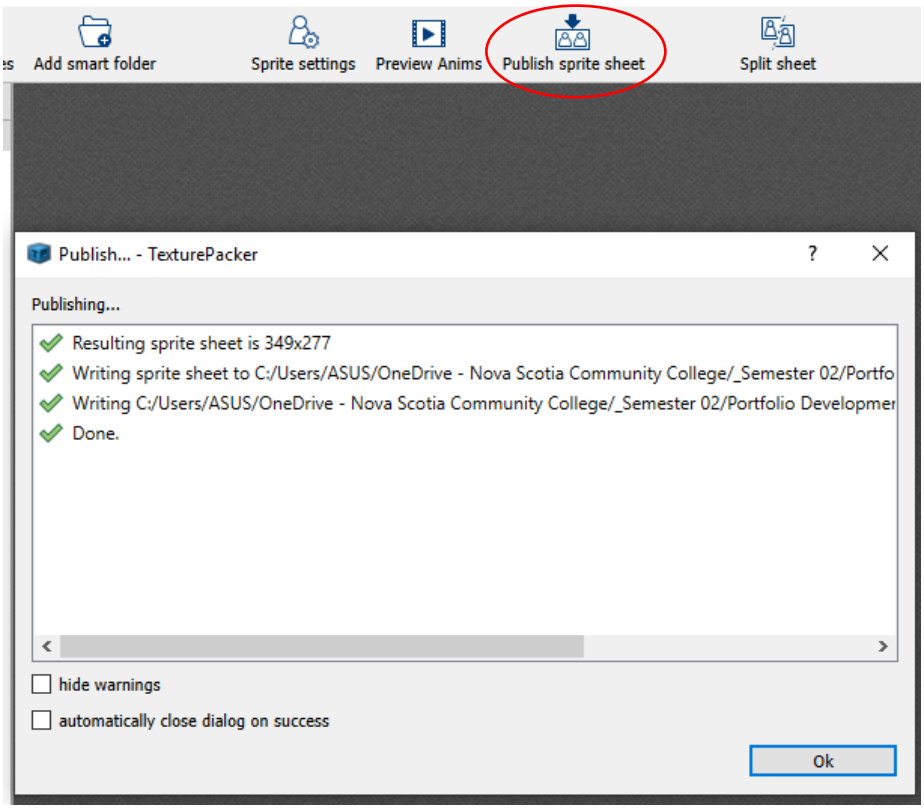


- Advanced settings should like these images below





- When everything is ready press the Publish Sprite Sheet button, a popup would appear and notify you that the export process has finished. It is time to send the exported JSON and PNG to your programmer or commit to GitHub.



- Do not forget to save your TexturePacker project.

2. Creating dynamic assets

Dynamic assets are sprites that has at least one animation, like a game character with walking, jumping animation.

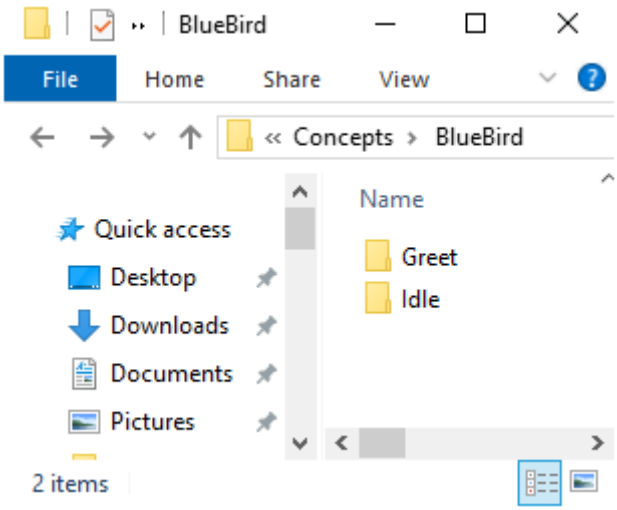
Casually, you could put all sprites of all artworks into one big JSON file. However, I would recommend you put them in separate JSON files. This helps a lot in preventing potential bugs that could slow your project down.

The process would be the same as creating static asset. In this section, I will demonstrate how to create a character sprite that has 2 different animations.

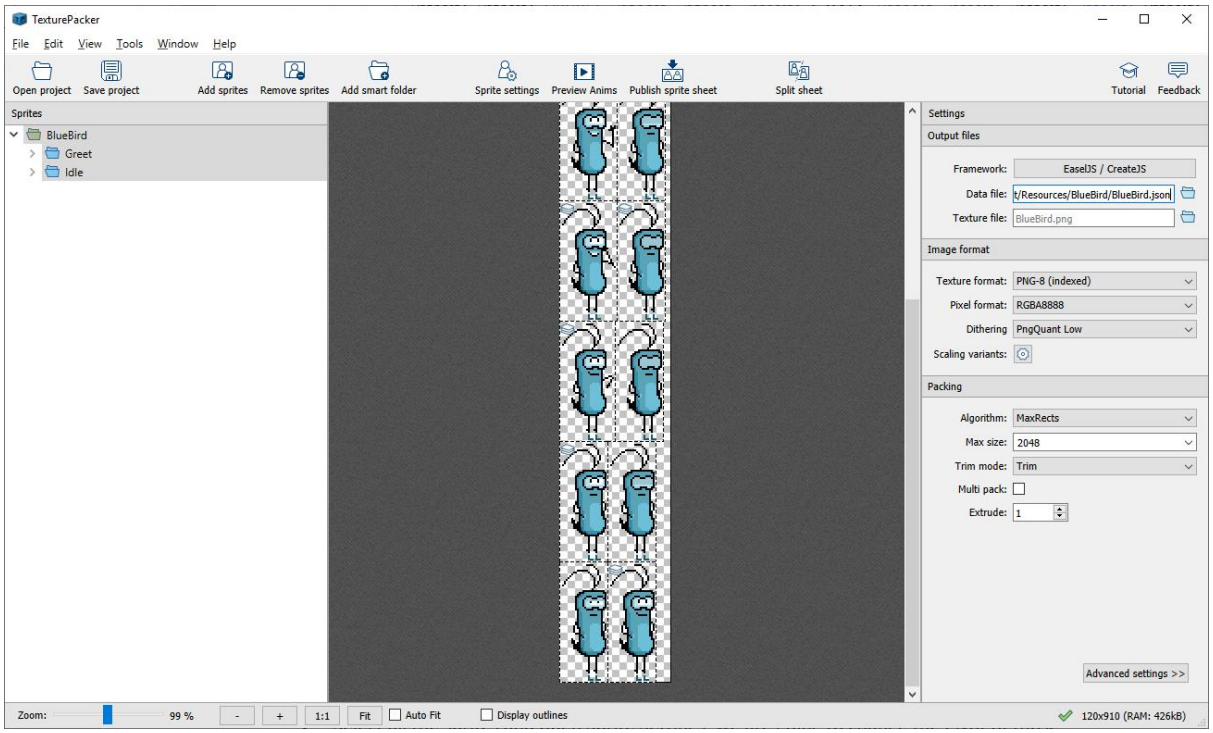
In this example, I am going to use my BlueBird character.

On a second thought, he does not really look like a bird. Welp! ¯_(ツ)_/¯

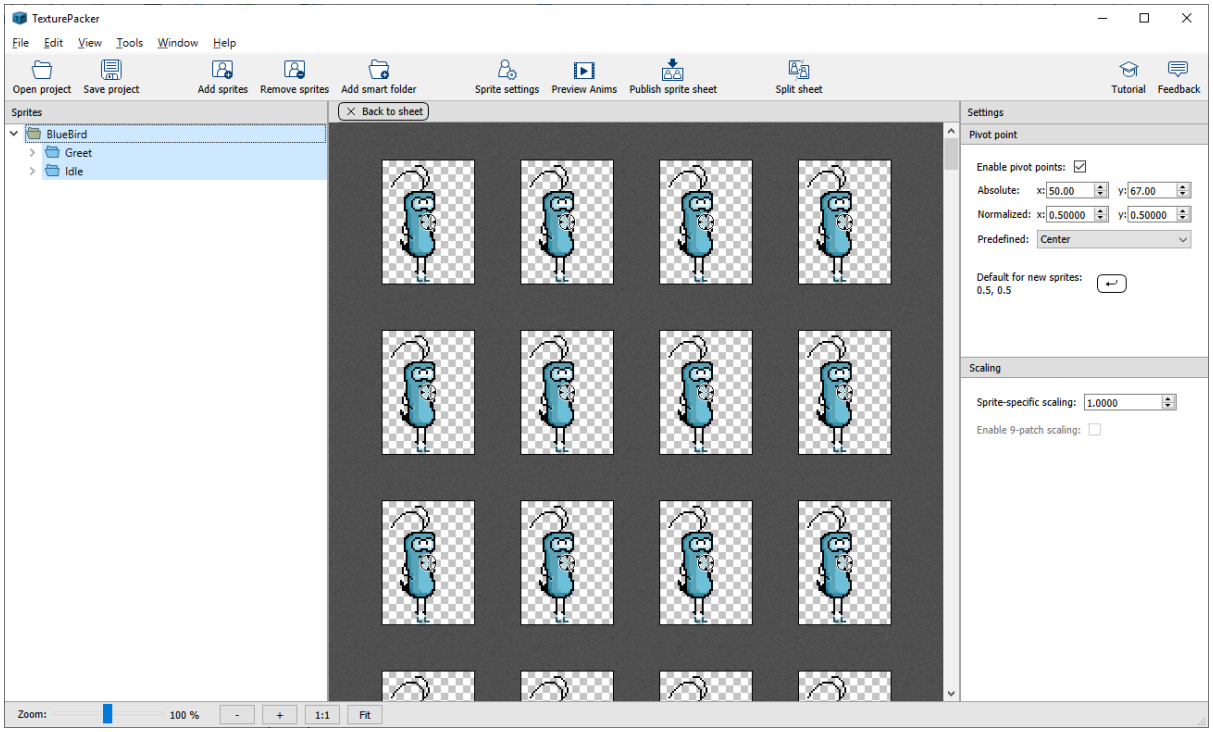
- I have two folders contain two different animation files exported from Photoshop using Render Video function.



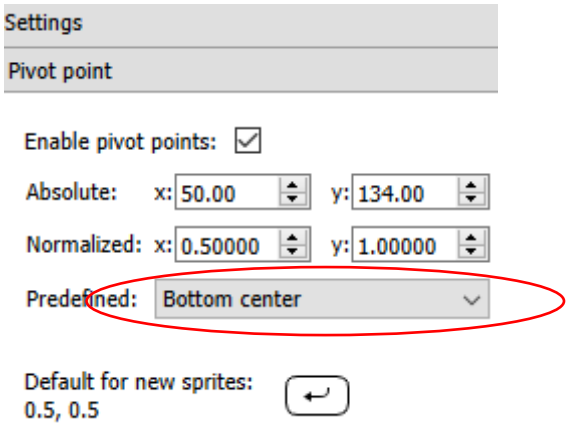
- Open TexturePacker and drag the parent BlueBird folder that holds the animations into it.



- Specify a new name and path for this sprite.
- Use the same settings described in Creating static assets section.
- Select all the sprites and open Sprite Settings, we are going to change the Pivot of those sprites a little bit different this time.



- Use Bottom Center for the Pivot Point



Explanation: This is an animated character that can be controlled by player. The feet are the most frequent things that collides with other objects during gameplay. Putting the pivot at the feet helps programmer a lot in coding. We could manually do it ourselves, but with the acknowledgement and help from artists would benefit all of us and make the work goes faster overall.

If you were unclear where you should set the pivot point of a certain sprite, just simply ask the programmer. If they are not available now and you need to finish your task anyways, set it to Top Left.

- You could hit Publish Sprite Sheet at this point.
- Do not forget to save your TexturePacker project file. You have been warned twice.

3. Summary

Things to remember before **publishing the sprite sheet**

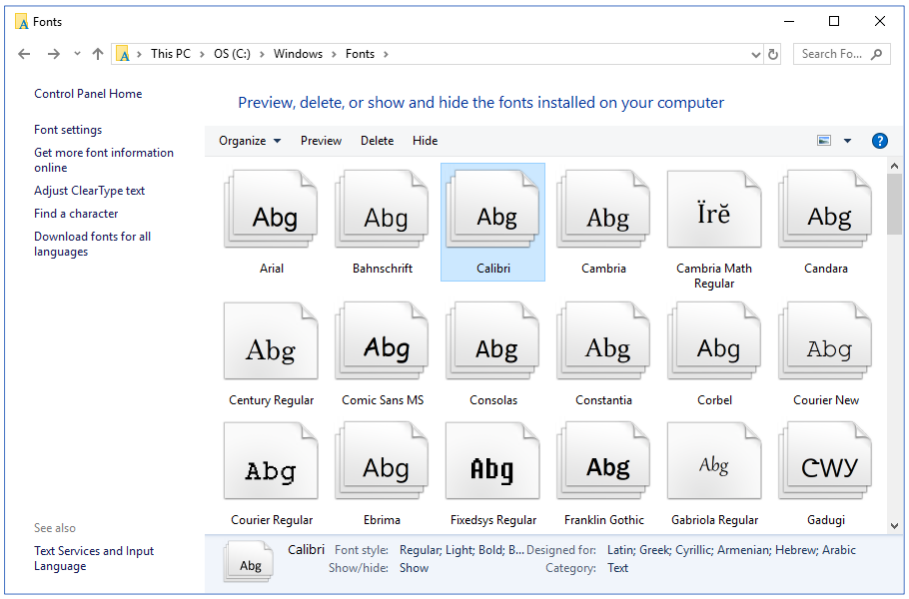
1. Make sure all settings and pivot points are set correctly.
2. Uses different JSON file for each character, do not make an All-In-One file.
3. If anything is unclear, give your programmer a nudge.
4. Save you TexturePacker project.

Creating Glyph assets aka. Bitmap Fonts for games

1. Preparations

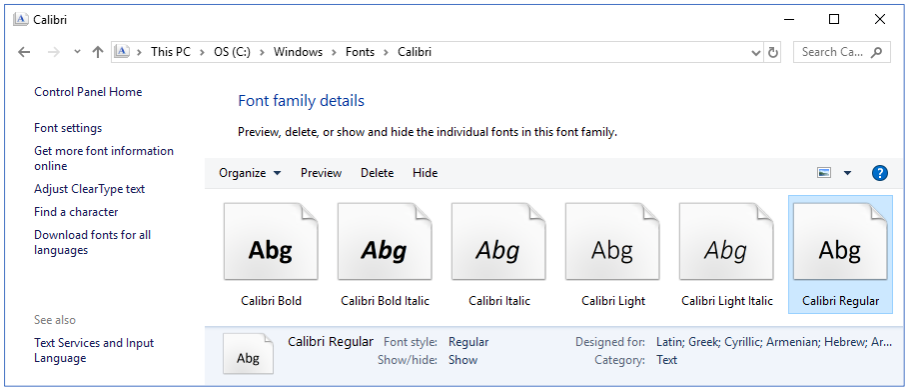
- Get the font of your liking on the internet, install it into your operating system. Or you could use any pre-installed font. For this instruction, I am going to use my all-time favorite Calibri.

- Get its postscript name. On Windows OS, open File Explorer (Window + E) (or right mouse click on the window icon on the taskbar). Go to Windows\Fonts. You would see something like this.

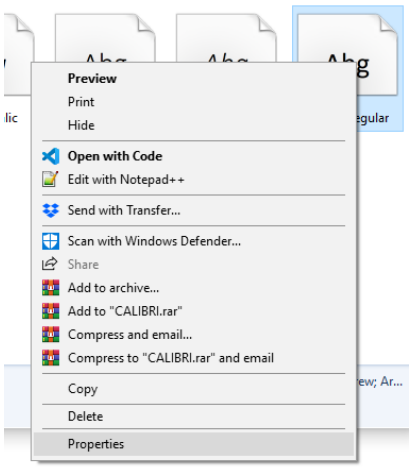


I do not have a Mac computer right now so I can not show you how to do that on MacOS. Please google yourself. Sorry for the inconvenience.

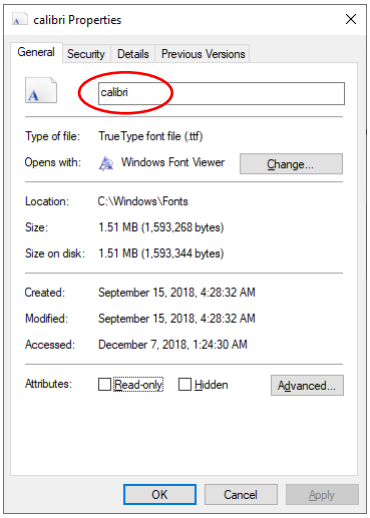
- Find the Calibri font. If it is a group, double click on it to view the family.



- Right click on one the family member and select Properties. I picked Calibri Regular.



- Copy / Write down the name appears in here.

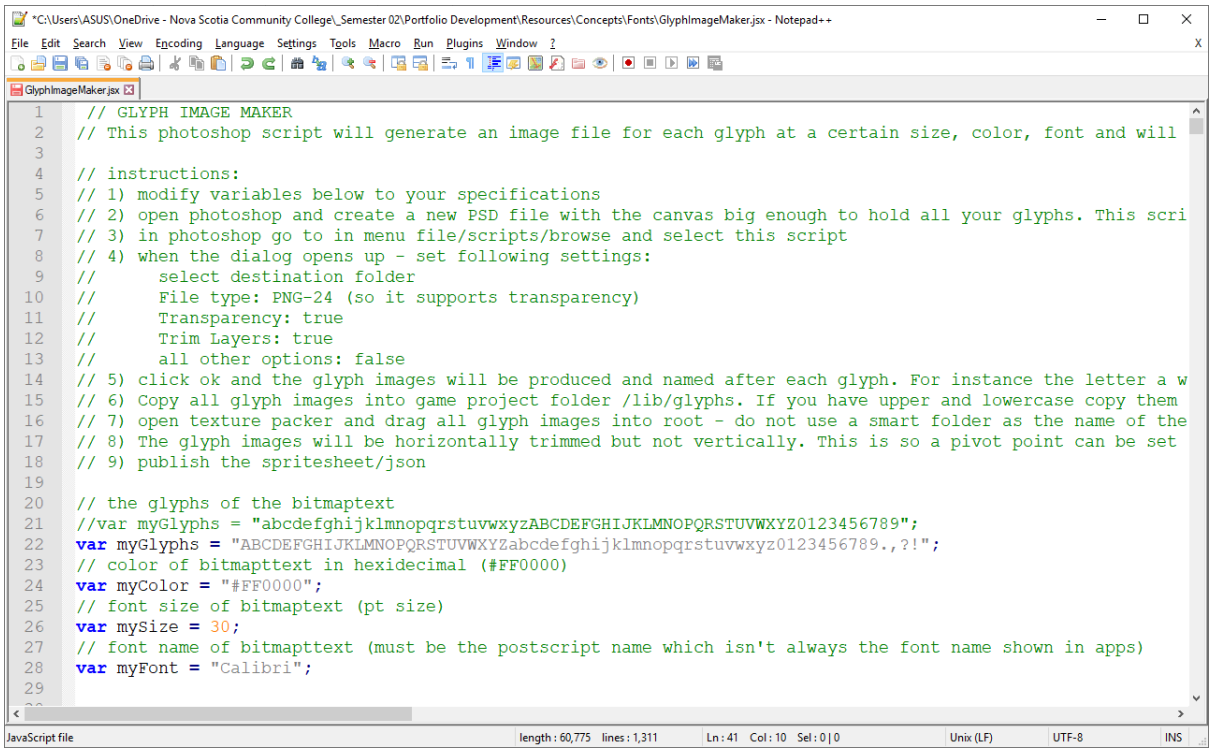


- You are going to need a Photoshop script to export font characters into PNG files. Get it here from the original author Adam Elsodaney - or contact Sean Morrow, I think he would gladly give you a copy of this script.
- From my experience with this custom script, I find has some flaws:
    - Terribly slow in case you wanted to get all the characters. It is a Photoshop thing, not your computer's.
    - Must start the script over if you made a small mistake which could cost you many hours. Very annoying!
    - Does not support Uppercase and Lowercase characters in the same batch.
    - Does not support special character such as: ? ! / \ you name it.

  After some time, I have come up with a fix for all those mentioned issues above.

2. The new way

- Open the script with a text editor. In my case, I used Notepad++.



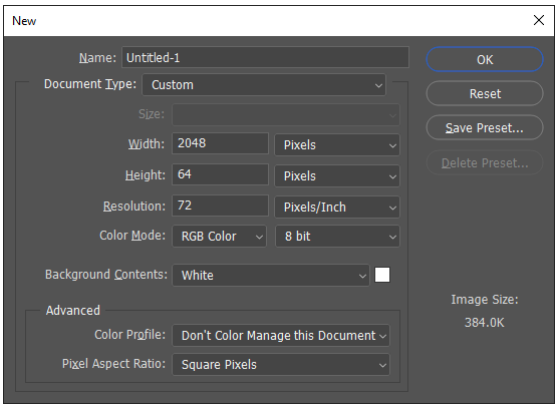
At var myGlyphs, put your characters of choice. I am going to add some uppercases, lowercases and special characters. For some weird flex, I always put uppercase characters before lowercases.

At var myColor, normally you would like a white color (#FFFFFF by the way). In this example I will use red.

At mySize: use your desired font size.

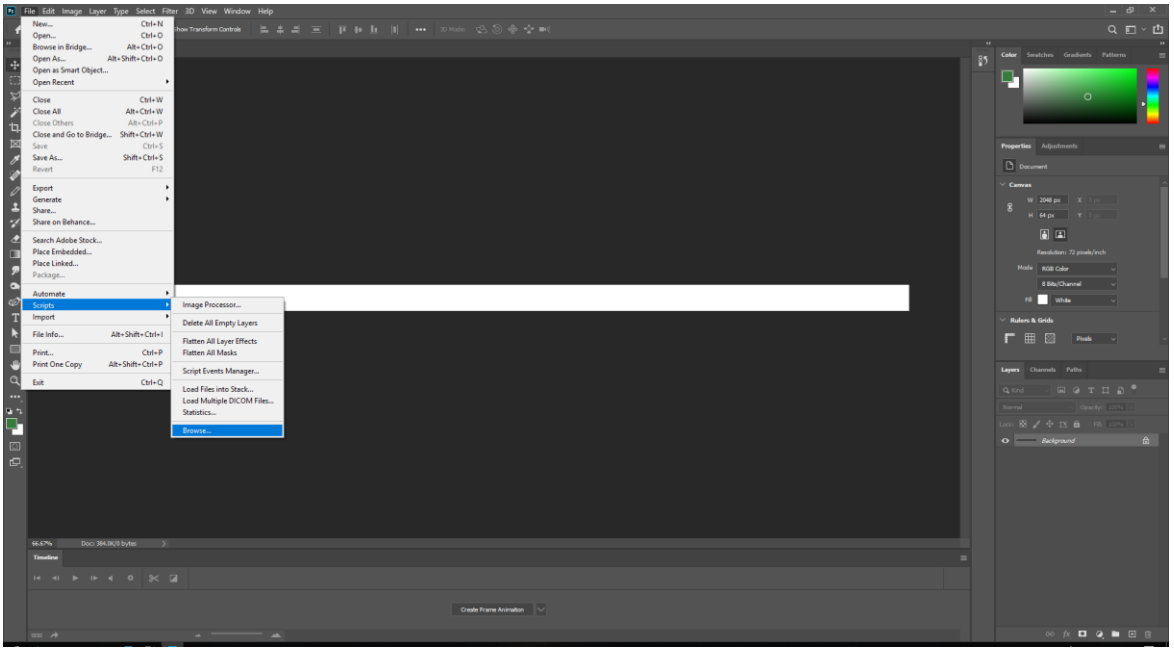
At myFont: put the postscript name here.

- Save and close the script.
- Open Photoshop and create a new long canvas. Something like this.

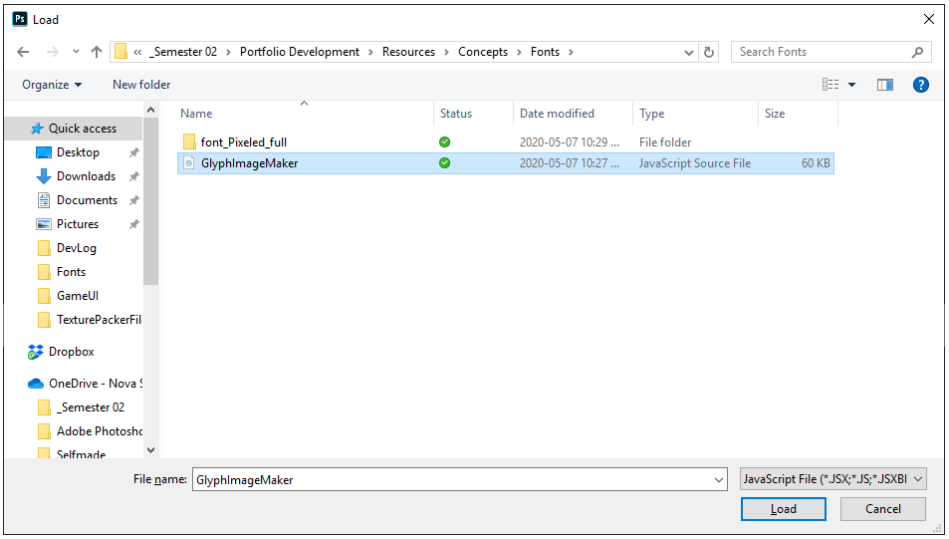


Because all your characters will be put on the same line so that you do not have to care about the height. You can always change the canvas height and width later so no worries.

- After creating the new file, go to File > Scripts > Browse…

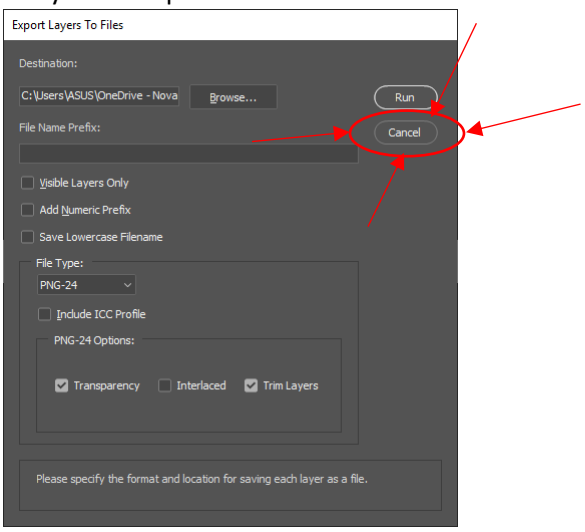


- A window will appear, look for the script location. Select and Load it.



- Wait a minute while the script is creating new layer for each character you defined in the script. Go on and treat yourself a cup of tea.

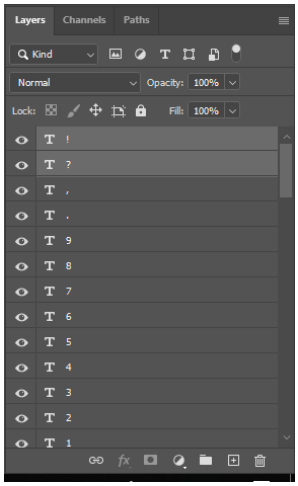
- After a while, a new window will appear. Cancel it. Yes! Right! Cancel it.
  We do not need the script beyond this point.



- Now it is time to make some adjustment for your canvas.
  In my example, the characters appeared to be put at the bottom left of the canvas, and the canvas seems too long. It does not really matter but let us make some adjustments to make it looks better.

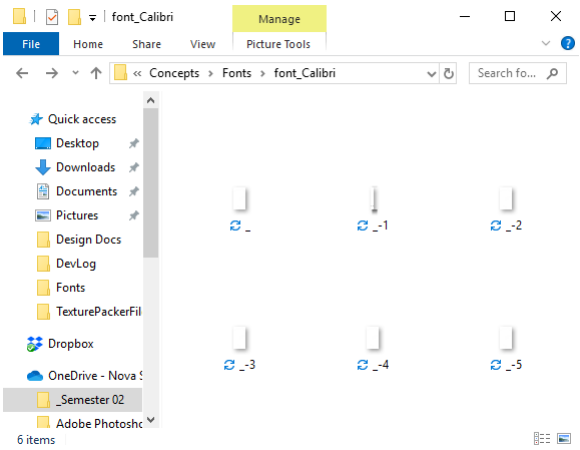
- I used crop to cut the redundant space, after that I aligned the characters group at the center.

- Save your file using PSD format so that you could make some change later.
- Look at the Layer window in Photoshop (in case have no idea what it is, press F7 multiple times) find the special character layers.



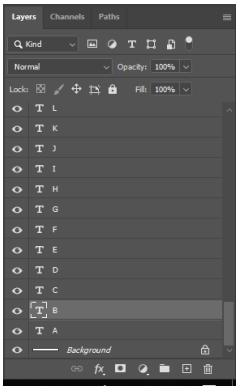
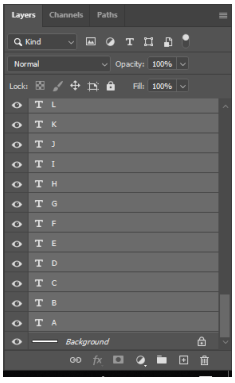
By some restrictions, most modern operating systems forbid us to put special characters in file and folder names. Photoshop knows that, so when you exported a layer contains special characters in it, with will rename the file to something else. In my case, ! and ? will be renamed to _.png and _-1.png. You could always rename them after exporting. However, let

us imagine that you have many special exported characters and all of them are in white color. Something like this:

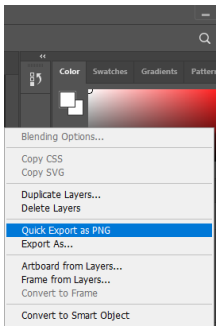


How to know which is which?

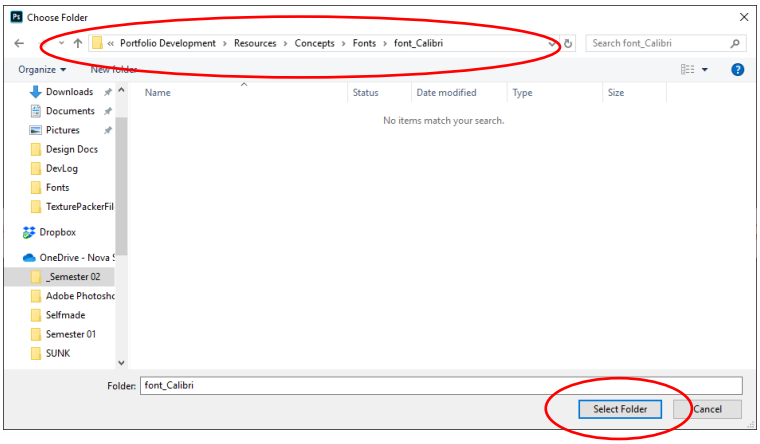
- To avoid that frustration, let rename all those special character layers with wordy names Some thing like this:
    - ? to “question”
    - ! to “exclamation”
    - Etc.
- Next, select all layers, except Background.

From here  to here 

- Right click on the selected layers choose Quick Export as PNG.

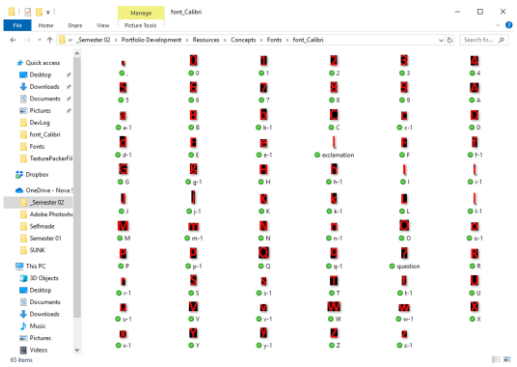


- A Choose Folder will appear, create a folder where you want to put all the exported file into. Some thing like what I did.



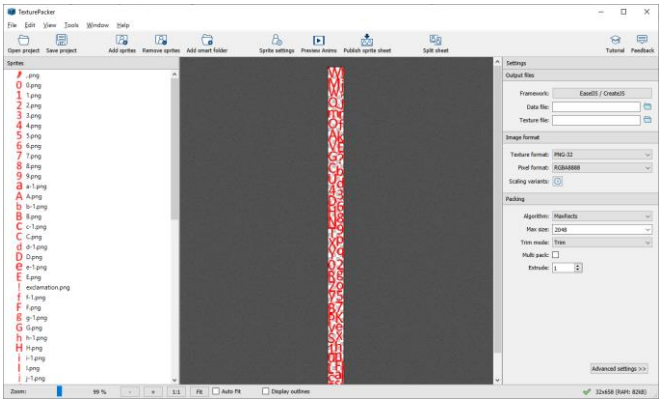
Hit Select Folder when you are done.

- Give Photoshop a few seconds to do the job. Also, give your cup of tea another sip. You can close Photoshop after this.
- As you can see, Photoshop has exported multiple PNG file for each character.



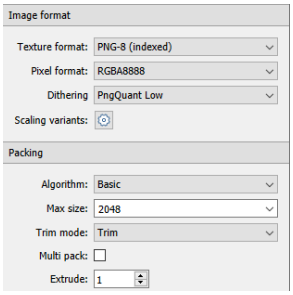
Do not panic if you saw black backgrounds on those files.

- Now, open TexturePacker and drag all those files onto it (do not drag the folder holding them).

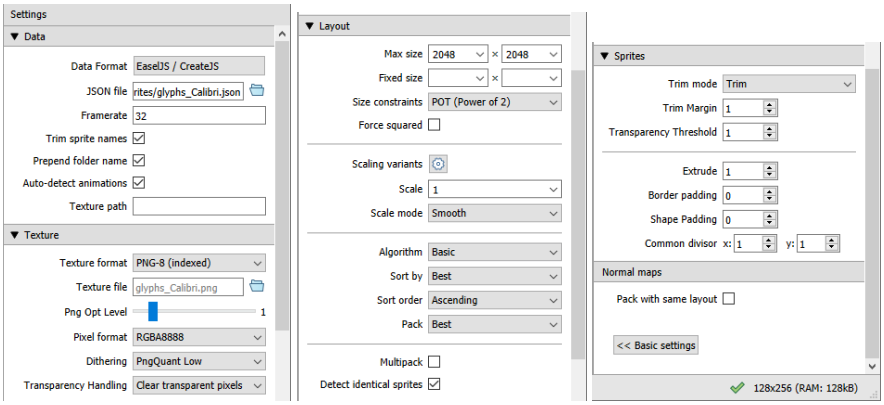


- You know the drill, same thing I talked in previous sections.
  - Select framework, specify data names and path.
  - Other settings should look like this

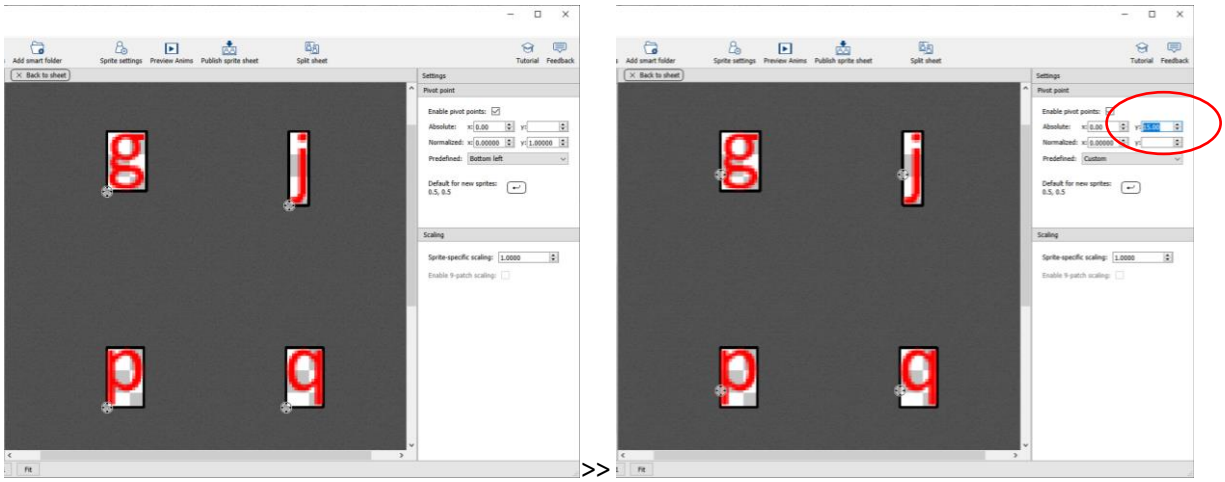
Basic settings



Advanced settings

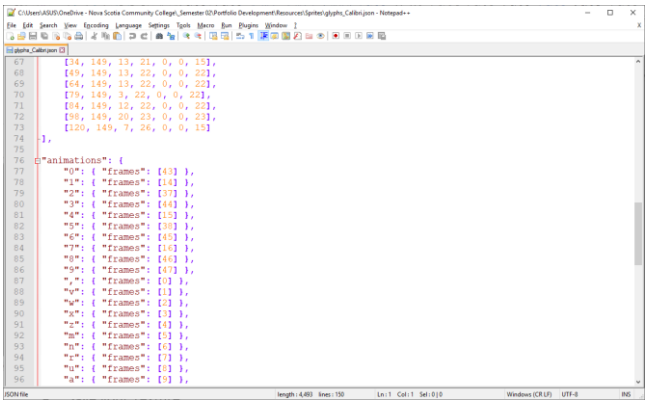


- Use Bottom Left pivot for all characters.
- You might notice that some lowercase characters such as **q y p g j** or **‘ , -** has a different character placement (depend on the font). You need to change their pivot point manually. Go ahead and select them. In Pivot point setting, adjust the Absolute y to suit their alignment within the font set. My images below are just examples. It may need a few tries until you found perfection.



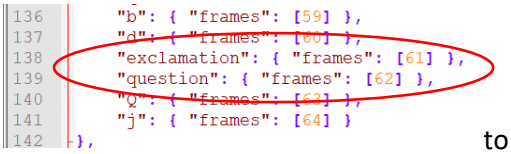
- Save your TexturePacker project and Publish Sprite Sheet.

- Last steps open the newly created JSON file using text editor, scroll down to the “animation” section.

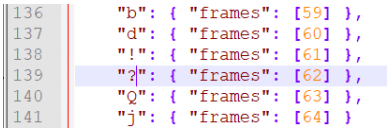


- Remember the special character files we had to rename before exporting in Photoshop? Find and replace them with the original character.

From



to



- Save this file. It is now ready to send to your teammate. As a programmer, I salute you for doing this. Much appreciated.

3. Summary

Important things of this section

1. You must use the font postscript name in the Photoshop script.
2. Rename the special characters in Photoshop before exporting them.
3. If you found some errors of any character, you can always go back and edit them in Photoshop, and export only them layers to save time.
4. Manually adjust pivot points of some lowercase or special characters depend on using font.
5. Change the special characters to their original names in JSON file.
6. Always save your files before closing them.

That is all for now. Thank you for reading my lengthy guide.

Happy making games.