**To Download additional support for external tools:**

<http://unity3d.com/get-unity/download?ref=personal>

**How To Connect Android To PC using Android Debug Bridge**

Download/install android studio and sdk.

Open android studio and click configure at the bottom and make sure that the software version is installed and under the sdk tab make sure that google usb driver is installed.

Open terminal and cd into android sdk (may be in a different directory than android studio)

* cd AppData\Local\Android\Sdk\platform-tools\
* There should be an executable called adb.exe

Run adb kill-server, then run adb start server.

On your phone enable developer mode, and then go to developer options and enable usb debugging

Plug in your phone into one of the mother board usbs. It may ask you to allow usb debug mode, hit OK

In the terminal type adb devices and see if youre device shows up. If so, well done.

**To Get Unity Remote To Work For Android**

Download/install Unity Remote on phone

Download/install latest java sdk

Download/install latest android studio/sdk

Follows steps to connect phone to PC using android debug bridge (shown above)

Open Unity, and under edit\preferences choose external tools, scroll down and put in paths for JDK and SDK. (If options to put in those paths don’t exist, you need to download and install android iOS support for Unity). Also in Unity, go to edit Project Settings\Editor and for Device choose any android device. (could change resolution as well).

Exit Unity, unplug phone, go to home screen on phone.

Plug in phone, allow usb debugging (if pop up shows up), then open Unity Remote on phone, then open Unity and choose your project, then hit play and it should work.

**To Build App on Android**

Make sure phone is in developer mode, has usb debugging on, and is connected through android debug tool (see steps above).

Make sure Android SDK and JDK are installed.

Make sure In Unity Preferences that SDK and JDK paths are correct.

Then go to File\Build Settings, choose android, then click on player settings.

-Under resolution and presentation, choose either landscape or portrait etc.

-Under Other Settings\Identification\Bundle Identifier type in com.”companyName”.”ProductName”

Now hit build and run on the Build Settings menu

This didn’t work for me first newest sdk so I had to download a previous version of sdk (25.2.5), install it, add that path in the Unity SDK path in Preferences\External Tools and then re-build and run. Itll ask you to update and just click yes. Then it should work

**How to Setup AdMob Plugin For Unity (better to look online for updates when the time comes)**

\*\*BEFORE IMPORTANT BACK UP YOUR PROJECT!!! Third party plugins can mess up your game.

Just followed this:

https://developers.google.com/admob/unity/start

**General Guideline for Game Creation Flow**

1. Open Unity and start a new project, choosing where to save it and whether it’s a 2d or 3d game.
2. Create Folders for all assets: fonts, sounds, sprites, scripts, scenes etc.
3. (Assuming all assets are already created) Copy all assets into the respective folders.
4. Setup sprites and sprite sheets
   1. Change to correct settings in inspector depending on it 2D or 3D
      1. Texture Type: for 2D put Sprite (2D and UI)
      2. For 2D uncheck Generate Mip Maps
   2. For image go to the Default tab in the inspector and choose none for compression.
   3. Max size should be set to a number above the biggest image dimension.
   4. For sprite sheet:
      1. You need to make Sprite Mode multiple.
      2. Click on Sprite Editor and slice sprites
         1. You may need to combine/deleted some of the boundaries since Unity is not perfect. You need to make sure that the boundaries are a little bit within the image, otherwise there will be zero alpha channel gaps in game if say you have identical backgrounds next to each other.
      3. When you’re happy with slicing, hit Apply.
      4. Name your sprite sheet components.
5. Create animations from sprites
   1. Click on the multiple images belonging to one animation and drag them into the world space.
   2. Save your animation.
   3. If you need to create another animation corresponding to the same object(s) then click on the dropdown menu with the animations name on it and select Create New Clip. Then drag your desired sprite onto the animation as a keyframe (you may need to reshuffle your windows so that both the animations and sprites windows can be seen).
   4. Set the correct animation as the default one in the Animator.
   5. Make transitions between the animations if needed.
   6. Create the necessary parameters needed to control transitions.
      1. Can be found on the left of the Animator window. Click on parameters. Then click on the plus sign and choose the type of parameter you would like.
   7. Once the parameters are made, you need to flick on the transition arrows, and on the inspector on the right create the logic for each arrow.
6. Add necessary colliders to objects and shape them accordingly
7. Add necessary Rigid Bodies to objects.
   1. Set correct Body Type
      1. Dynamic for ambient forces and applied forces.
      2. Kinematic for applied forces.
      3. Static for no forces acting on it.
8. Make prefabs.
   1. Once objects are ready with their components (besides any scripts) drag them into a prefabs folder. Now you will be able to pull that prepared object anytime you want.
      1. NOTE: if any changes are made to the game object in the scene, you need to hit apply in the inspector for the prefab to change as well.
9. Add all possible Tags, Sortings Layers, and Layers
   1. Tags are used simple as a referencing tool.
   2. Sorting Layers determines rendering order.
   3. Layers determines what physically interacts with what.
      1. Can control these interactions in Edit/Physics2D.
10. Next instructions depend on your particular game…GoodLuck!!

***Important Unity Functions***

**GetComponent<>() :** will get a particular component from the game object that the script is attached to.

* private Rigibbody2D myRigidBody;
* myRigidBody = GetComponet<RigidBody2d>();

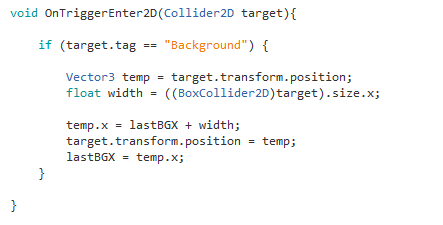
**[SerializeField] :** will show field in attached game object in Unity workspace even if the variable is private. You can just attach the whole game object in the serialized field and it will find the respective component for you. NOTE: this way of loading variables is quicker than using GetComponent in the Awake function.

* [SerializeField]
* private Rigidbody2D myRigidBody;

**GameObject.FindGameObjectWithTag() :** can use this function to find a particular game object that has been tagged with a string. Can use *GameObject.FIndGameObejctsWithTag()* to find all game objects that have been tagged with a particular string.

* private Button flapButton;
* flapButton = GameObject.FindGameObjectWIthTag(“FlapButton”);

**OnTriggerEnter2D(Collider2D target){} :** used to perform actions for during triggering of colliders



**OnCollisionEnter2D(Collision2D target){} :** used to perform actions for the collisions of two objects, when neither is a trigger (think of physically impenetrable contact). Make sure if you get the tag of target you do target.gameObject.tag as shown below.

