Playmaker Platforming Starter Kit

Documentation

Thank you for purchasing the Playmaker Platforming Starter Kit! Please make sure you have Playmaker installed, as this kit will not function without it.

Feel free to contact us if you have any questions or suggestions either by using the contact form on **3dsauce.com**, or by email: **3dsauce@gmail.com**

Contents

1 - Getting Started

Once you have installed both Playmaker and this kit. You can begin by loading the **"DemoLevel"** scene. The gameobject **"PlatformingController"** has fully commented logic within it's FSM inside the Playmaker Editor.

If you run in to any issues or irregular behavior with this kit, the following may help you. Project Settings and Global Variables should install themselves when you import the package for the first time, but in some rare cases this does not work as intended. For these cases you will need to navigate to the resources folder and extract the “ProjectSettings.zip” into the root of your project overwriting the old ones when prompted. Also select the “PlatformerGlobals” and select “Import Globals” in the inpector window. Following these two steps should remedy any issues you may have had with the installation.

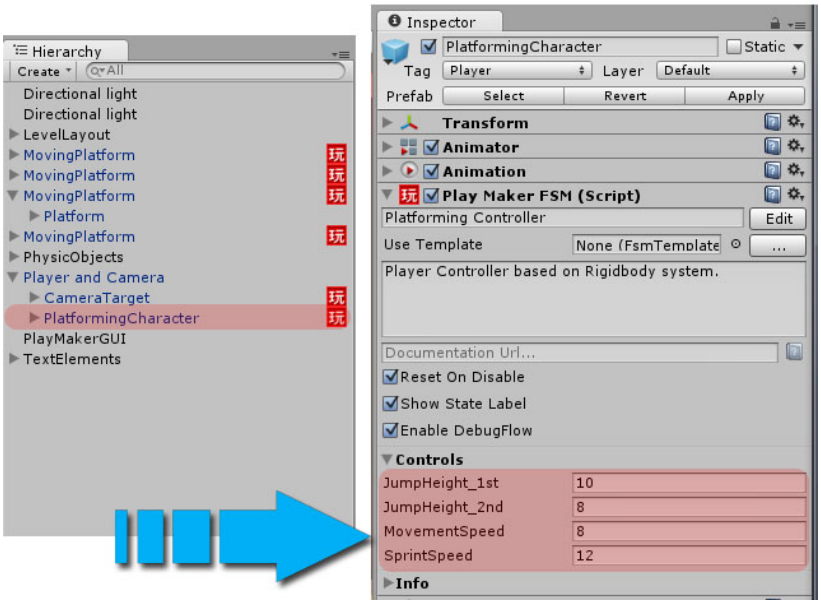
For assistance in putting your own game together and customizing the feel of the controller, please refer to the following guidelines.

2 - Customization

To tweak your player's Movement Speed, Sprint Speed and Jump Height:

Select the "PlatformingCharacter" scene object from the hierarchy tab.

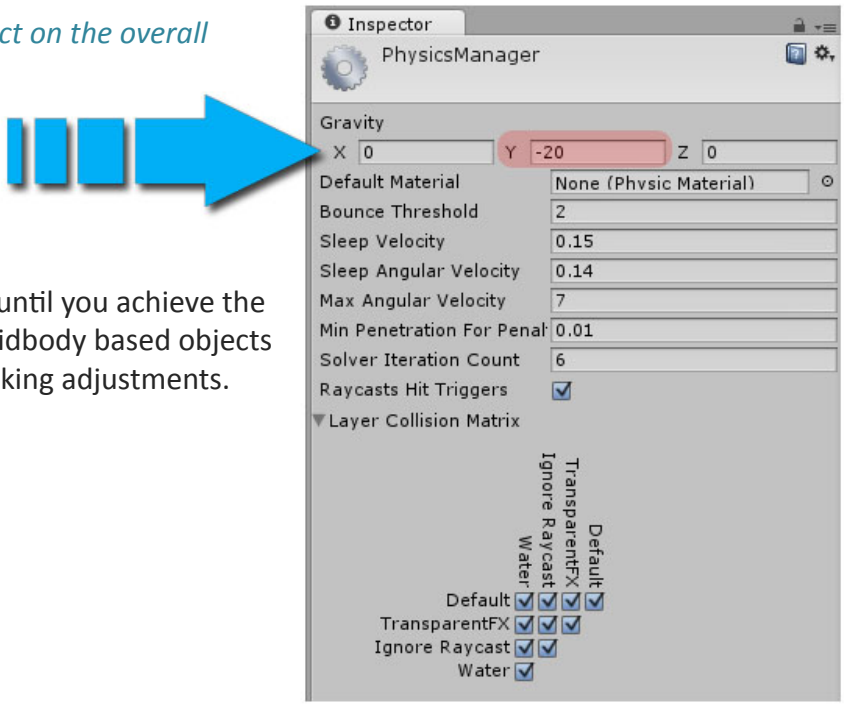
You will find a number of variables exposed and ready to edit in the inspector tab as shown here.



To change the game's gravity:

*(**Note** - This setting will have a huge impact on the overall feel of your game, use sparingly)*

1. **Go to - Edit/Project Settings/Physics** to access the physics settings within the inspector tab.



2. **Adjust the gravity value on the Y-Axis** until you achieve the desired feel. This also affects all other Rigidbody based objects in your scene, keep this in mind when making adjustments.

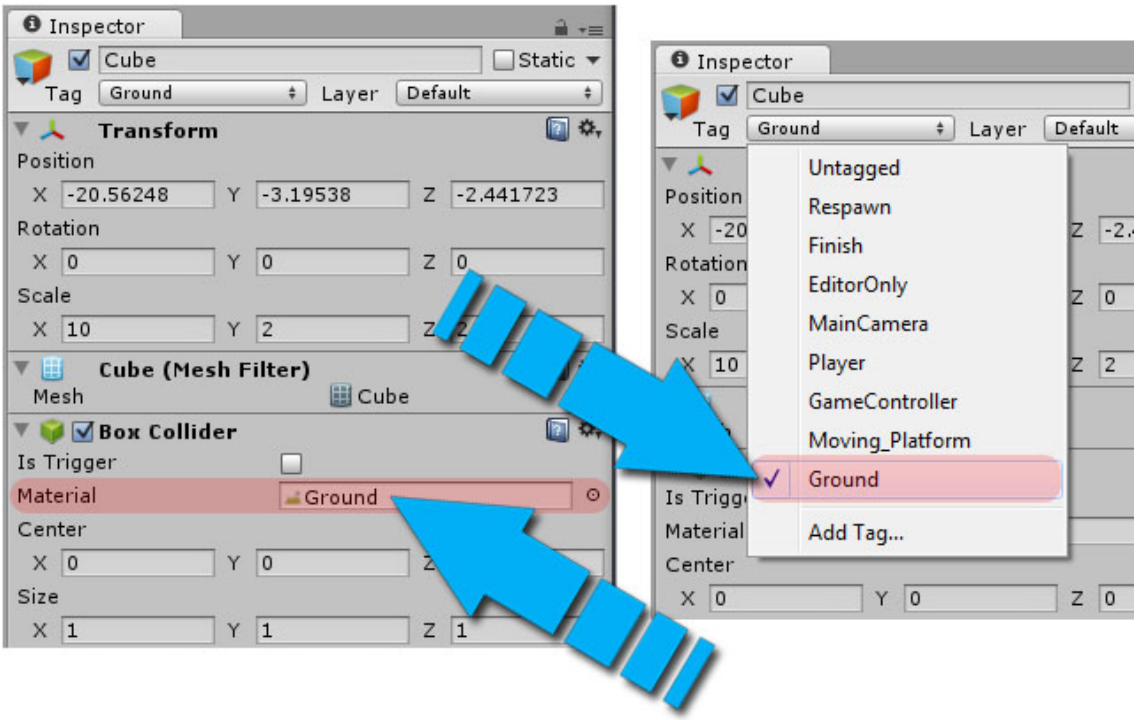
3 - Level Construction

To create the basic ground object:

1. Start by creating a mesh in the scene, it can be one of your own, or simply a basic cube from Unity.

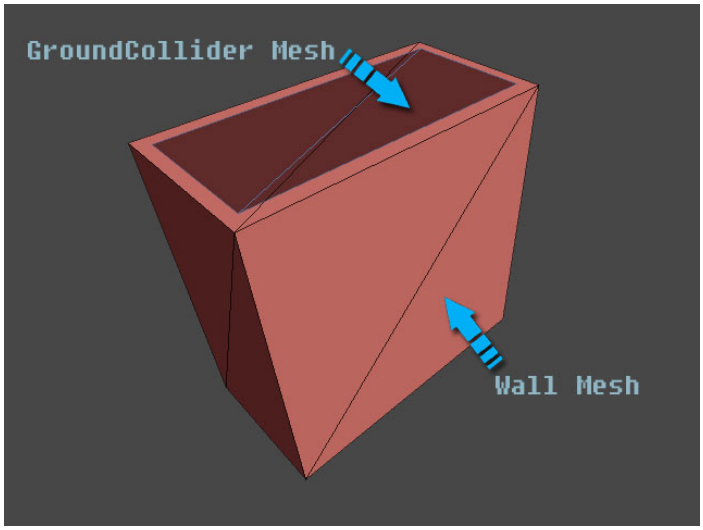
2. **Ensure that you have tagged the ground mesh "Ground"** as shown below.

3. **From the "Physic Materials" folder, Drag the "Ground" material into the Material slot of your ground mesh's Box Collider.** This will ensure rigidbodys interact with the ground in a desired manner.



To create the wall object:

*(**Note** - Don't use a ground object as a wall, it would allow the player to exploit unlimited jumps.)*



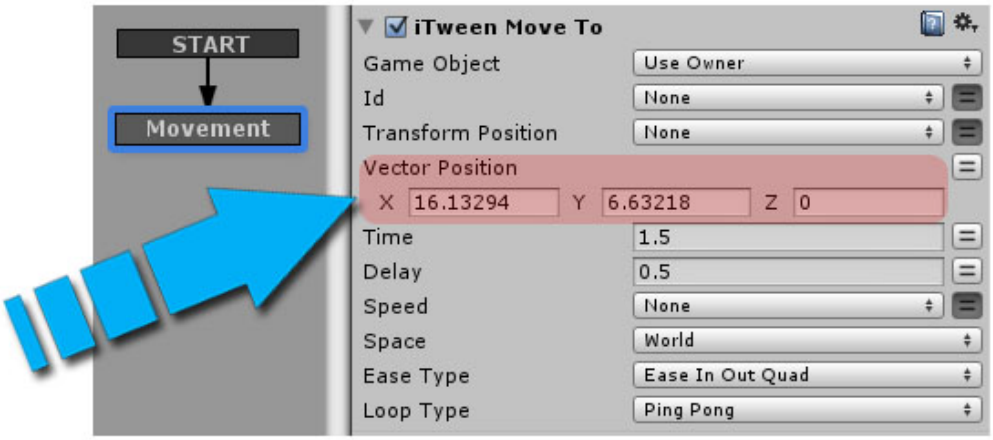
1. Start with your own mesh or a Unity cube. You can leave this mesh's tag as the default "Untagged."
*(**Note** - You will also use this wall setup whenever the player is able to hit their head on this mesh from below, again this will prevent an infinite jump loop from occuring)*

2. *(If the player needs to walk on top of the wall mesh, follow this step, if not you can leave it as is.)*
From the "Prefabs" folder, Drag the "GroundCollider" prefeb into the scene. Position it Just above the mesh, and scale it to match the walkable area on top of the wall. Parent it to the wall mesh.

To create a moving platform:

Place the "MovingPlatform" prefab into the scene. The platform will move back and forth between the position you placed it in, and the position defined by the "iTween Move To" action in the platform's FSM.

*(**Note** - You will notice the parent is merely an empty game object with a scale of 0,0,0. This will prevent any player scaling issues when they land on, and leave the platform. The next child is the actual visible mesh itself, with no tag defined. The final child is basically the GroundCollider on top that will be used to reset jumps properly. All of these elements are required for smooth operation.)*



To add the controller and camera to a scene:

1. Simply Drag the "PlatformingController" prefab and the "CameraTarget" prefab into the scene. Move the "CameraTarget" to the approximate coordinates of the "PlatformingController."

2. Delete the Main Camera from your scene, as there is already one included inside of the "CameraTarget" prefab. Make sure the "PlatformingController" is placed above a ground object to avoid falling into the infinite abyss of Unity space.

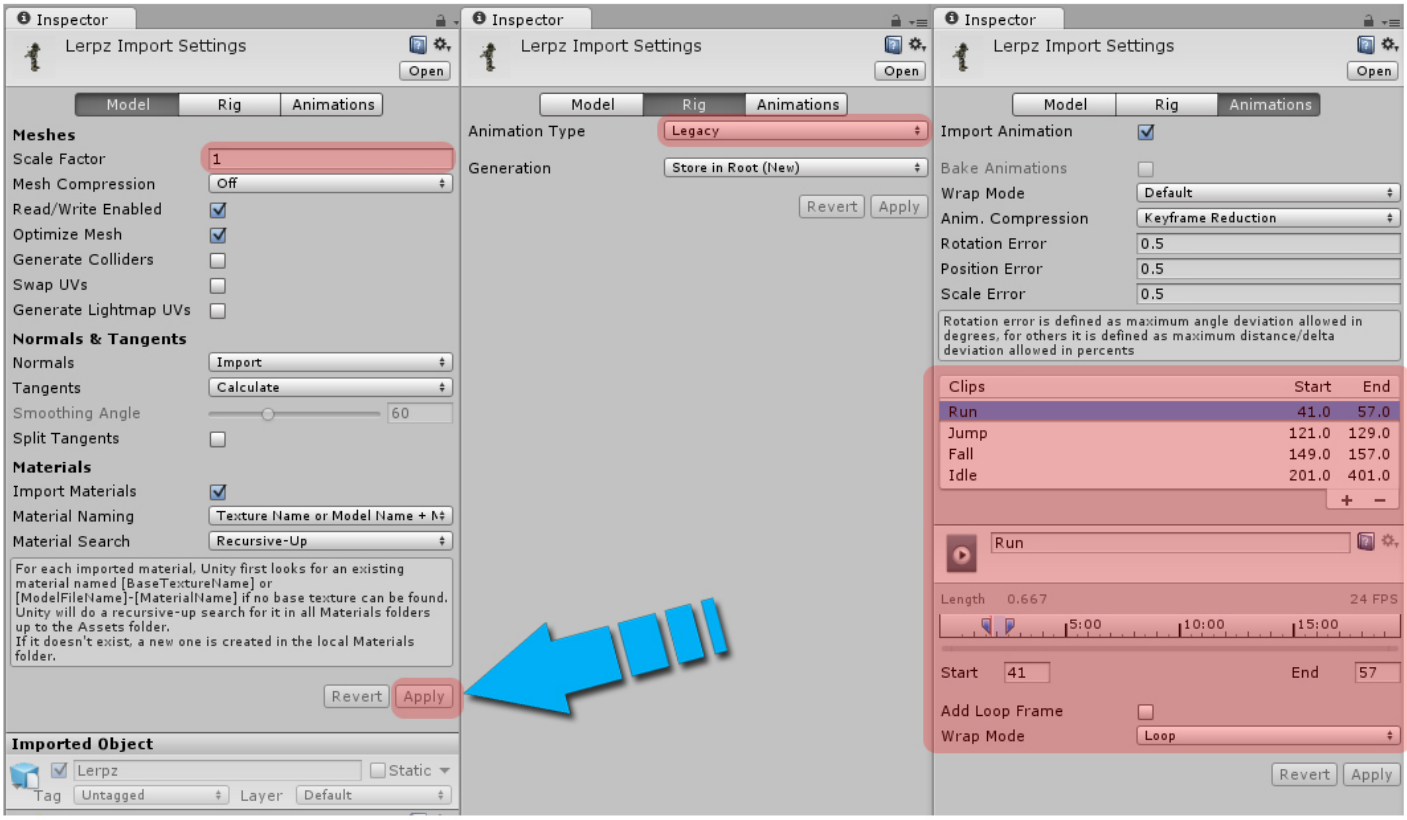
4 - Player Model Swapping

1. Begin by importing your character model with animations into Unity. For this example I am using the Lerpz model from the Unity 3D Platform Tutorial. Drag the model into the scene and place it next to the default "PlatformingController" as a reference. Don't worry about our character's orientation just yet. Select your model in the project tab under the folder you imported it to in order to see all of the model's settings in the Inspector panel.

2. If your model seems to be the wrong scale, you can adjust the size in comparison to the "PlatformingController" mesh you placed it beside in the scene view. You can access this setting under the "Model" tab in the import settings you just opened in the previous step. Adjust the scale factor and hit "Apply." Keep adjusting until the scale closely matches the default mesh.

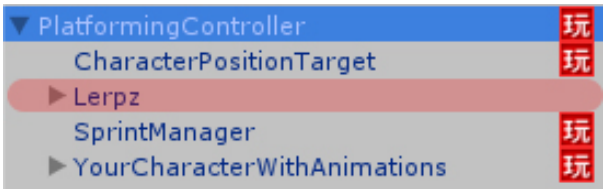
3. Under the "Rig" tab, set the Animation Type to "Legacy" and hit "Apply."

4. Under the "Animations" tab, take a look at the animation included in your model. To work out of the box with this kit, it will need to have the following clips available: ("Idle" / "Run" / "Jump" / "Fall") In this example, I have gone ahead and renamed the clips to match, and deleted the unused animations. (Remember that capitalization matters here, eg. "Run" not "run") Also, set the wrap mode of your "Run" and "Idle" animations to "Loop." Again, remember to hit the "Apply" button at the bottom when done.

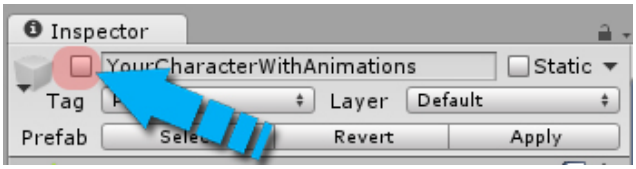


5. Match the new model's coordinates to the "PlatformingController."

6. Parent the new model to the "PlatformingController."

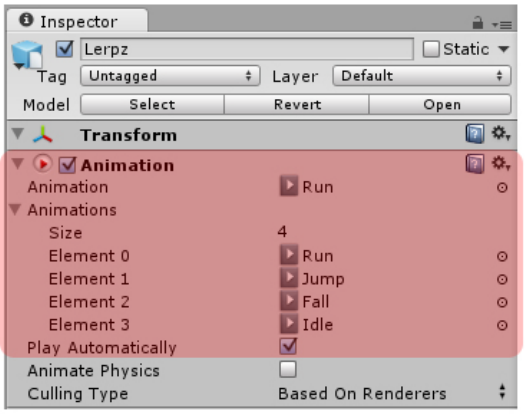


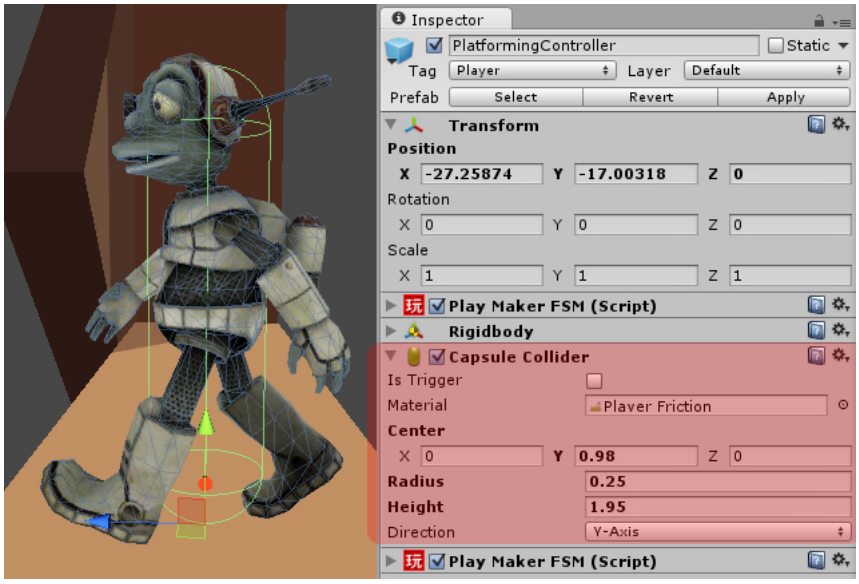
7. Disable the "YourCharacterWithAnimations" object in the inspector. This will make things easier to see.



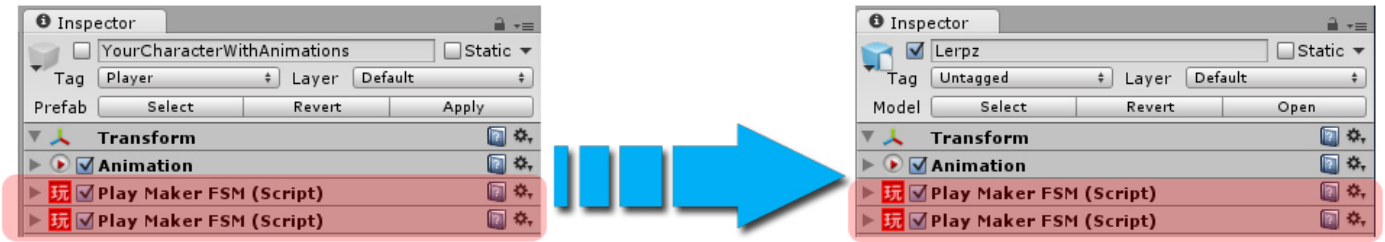
8. Adjust the Capsule Collider component of the"PlatformingController" to match the size of your new model.

9. Ensure that your character has an "Animation" component, and that all the animations are listed in it.

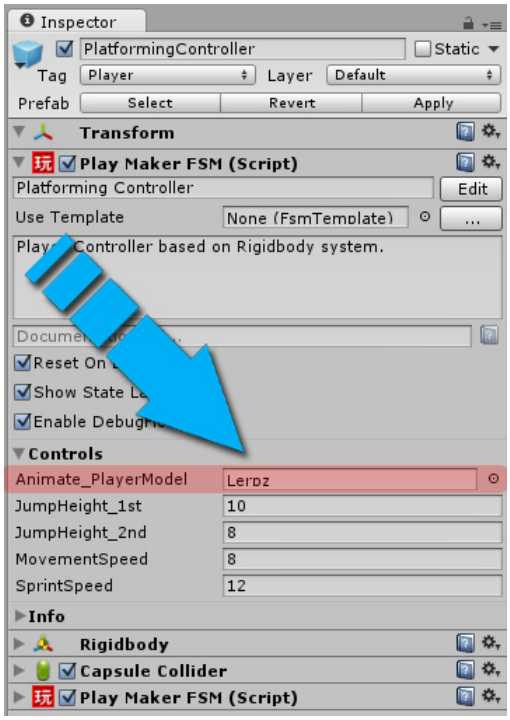




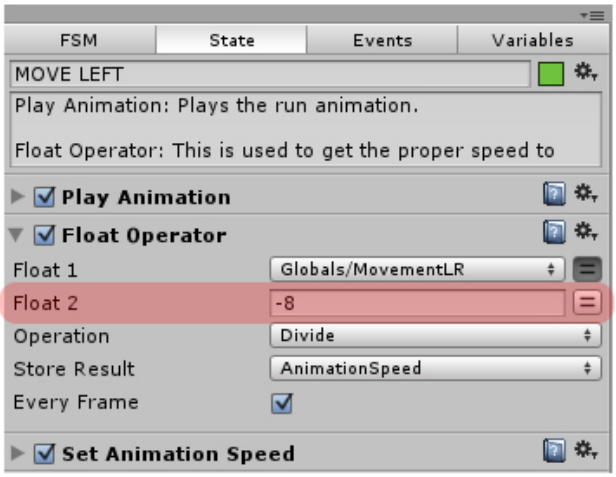
10. Copy both Play Maker FSM components from "YourCharacterWithAnimations" object, to the new player model.



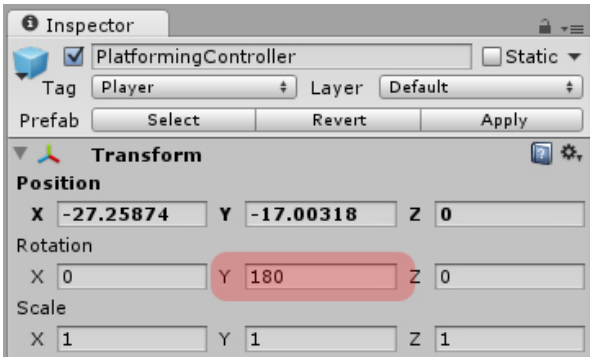
11. Drag the new model from the Hierarchy list into the exposed variable slot "Animate_PlayerModel" under the first FSM on the "PlatformingController" so that the new model is now listed.



12. You should now be able to press play to see your model running around. You may notice the speed of the animation is too fast or too slow. To adjust this, open the new models "Animation Controller" FSM. Under both the "MoveLeft" and "MoveRight" states, you will find the "FloatOperator" actions. Change the float 2 value on both of them and press play, repeat this step until the animation appears to be matching the speed of your character. (In the "MoveLeft" ensure to keep the value a negative so that the legs don't appear to be running in reverse.)



13. You may have to adjust the "PlatformingController's" Y Axis rotation if you have any orientation issues. I set mine to 180 degrees so that the smooth rotate would face the screen when changing directions.



14. If you don't like the way your clips are blending together, you may have to make a few tweaks to the clip durations and ranges under the animation tab that we explored in step 4. You can also Tweak the "Blend Time" of each Play Animation action in the animation controller. Keep in mind that each clip needs to be long enough in order for them to blend properly. In this example I deleted the "Falling" state entirely as it just looked better without it, it's up to you to decide what will best suit your character.

5 - Mobile Controls Setup

! You must have “Virtual Controls Suite” in order to use these virtual controls.
http://bitbybitstudios.com/portfolio/virtual-controls-suite/
Used with permission from Sean Sanders of BIT BY BIT STUDIOS

1. Begin by installing the Virtual Controls Suite Package from the unity asset store. When unpacking it you will be given choices as to which GUI configuration we’ll be using. Choose the “VCS_GUITextures” package from the Resources folder and import it.

2. In the “Playmaker Platforming Starter Kit” Folder you will find another unity package called “VCSMobileSupport.” Import this package as well.

3. At this point you can simply load the “DemoLevel(VCSMobile)” scene from the Scenes folder and take it for a spin!

Adding the player and controller to a new scene:

1. If you wish to load the Player and controls into a new scene from scratch, begin by dragging the “(VCSMobile)PlatformingController” prefab and the “Prefab_VCSMobileControls” into your new scene.

2. Make sure the “Prefab_VCSMobileControls” coordinates are set to 0,0,0 so that the virtual controls appear in the Game preview window.

3. In the “(VCSMobile)PlatformingController” you will find a state called “MobileJumpReset.” Drag the “buttonJump” child from the “Prefab_VCSMobileControls” into the blank GameObject slot within it’s Send Event action.

4. In the “buttonJump” child, the Fsm “State 2” and “State 4” each have a Send Event with an empty GameObject slot. Drag the “(VCSMobile)PlatformingController” into both of these slots.

5. In the “buttonSprint” child, The Fsm “State 1” and “State 2” also both have Send Event actions with blank GameObject slots. Drag the “SprintManager” child of “(VCSMobile)PlatformingController” into both of these slots.

6. Change your build settings to IOS or Android and take it for a spin!

Best of luck in your platforming adventures!

3dsauce.com