Character Animation Pack
README

1. What you can do with Mixamo Character Animation Packs

Animate **ANY** characters from **ANY** of the Mixamo Character Packs of the same gender – mix 'n match to your heart's content*! (*several Mixamo Character Packs are available in the Unity Asset Store, see step #3 for details*)

Use the pack assets in a variety of ways:

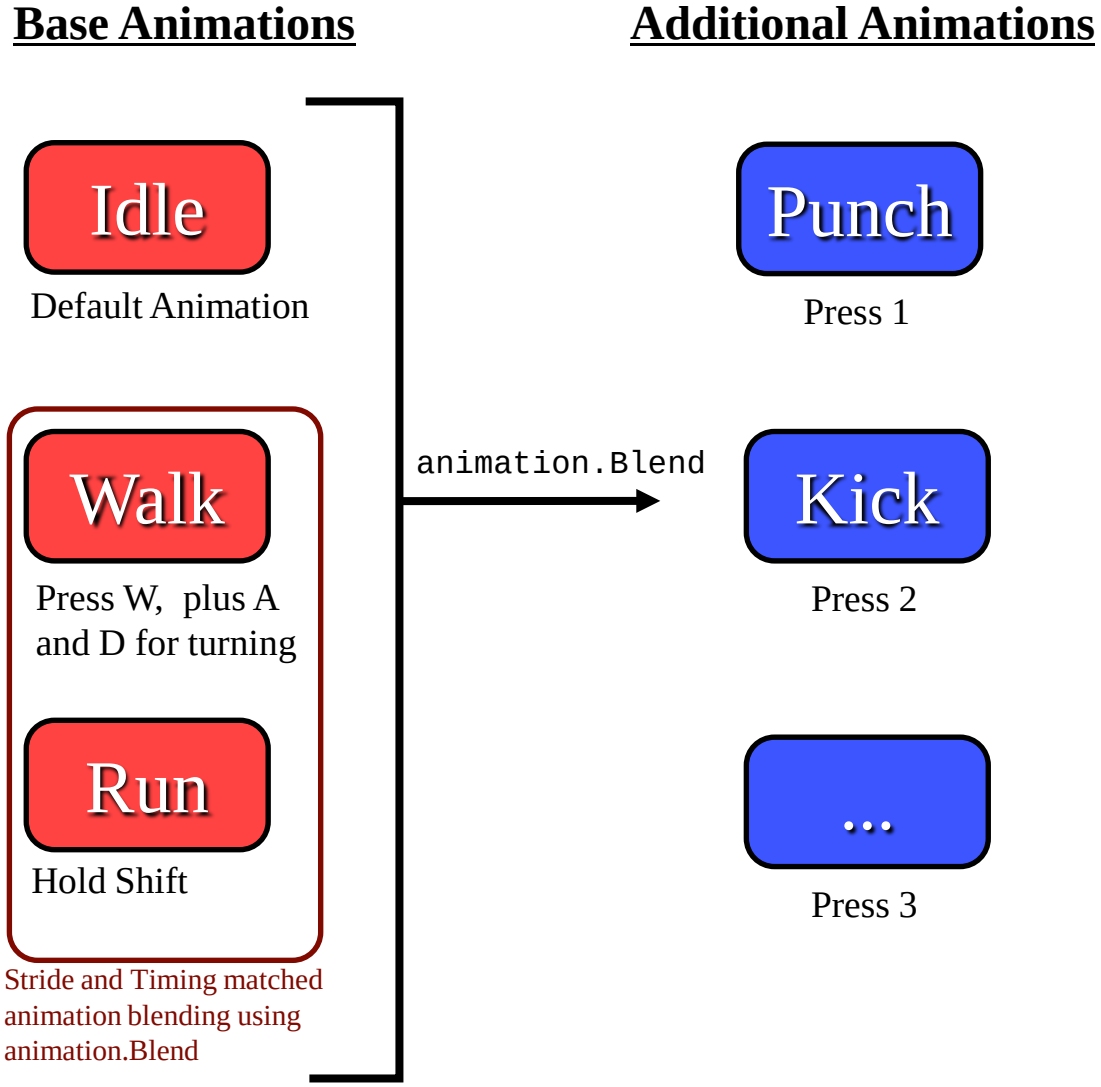
- Use the included **Demo Scene** to see all of the animations using preconfigured controls that are already hooked up for you (*see next page for details*).
- Use the **Prefabs** in your own scenes, including assets and associated code.
- Or just use the **original assets** (characters, animations) individually in your own scenes, by applying the Root Motion Controller or your own script. The Root Motion Controller is an easy way to animate your character. (*learn more at www.mixamo.com/c/unity*)

NOTE: All Mixamo Character Animation Packs REQUIRE the "Root Motion Controller", which can be downloaded for FREE from the Unity Asset Store.

** Biped characters & animations only. Mixamo quadruped packs cannot be mix 'n matched.*

2. How the Demo Scene Works

The Mixamo Character Animation Pack comes with a complete Unity project and animation scripts. The Demo Scene in this project allows you to view all of the animations using preconfigured keys that are described to the right:



3. Setting up ANY Mixamo Character with the Root Motion Controller

1. Drag the character into the Scene
2. Attach a Character Controller (Set the height, radius and center)
3. Attach the RootMotionComputer (found in the Root Motion Controller package on the Asset Store)
4. Attach the RootMotionCharaterController (found in the Root Motion Controller package on the Asset Store)
5. Attach the Required Character Animations to the animation component.
6. Play Scene.

NOTE

The Root Motion Computer and Root Motion Character Controller will automatically map the required information, you should only need to map this information if there is a problem with the automatic mapping.

NOTE2

You may need to customize the RootMotionCharacterController in order to work with all the animations you've applied to the character. If such is the case please see the next page for instructions on how to edit the RootMotionCharacterController Script to work with any and all your animations.

4. Setting up all of your Animations on the Root Motion Character Controller

In the RootMotionCharacterController Script edit the following lines to include your animations and the proper WrapMode.

```
// set up properties for the animations
animation["idle"].layer = 0; animation["idle"].wrapMode = WrapMode.Loop;
animation["walk01"].layer = 1; animation["walk01"].wrapMode = WrapMode.Loop;
animation["run"].layer = 1; animation["run"].wrapMode = WrapMode.Loop;
animation["attack"].layer = 3; animation["attack"].wrapMode = WrapMode.Once;
animation["headbutt"].layer = 3; animation["headbutt"].wrapMode = WrapMode.Once;
animation["scratchidle"].layer = 3; animation["scratchidle"].wrapMode = WrapMode.Once;
animation["walk02"].layer = 3; animation["walk02"].wrapMode = WrapMode.Once;
animation["standup"].layer = 3; animation["standup"].wrapMode = WrapMode.Once;
```

Then edit the following lines to map your actions to different keys. The Default are number 0-9 This example shows the Zombie Animations mapped to 1-5.

```
// all the other animations, such as punch, kick, attach, reaction, etc. go here
if (Input.GetKeyDown(KeyCode.Alpha1)) animation.CrossFade("attack", 0.2f);
if (Input.GetKeyDown(KeyCode.Alpha2)) animation.CrossFade("headbutt", 0.2f);
if (Input.GetKeyDown(KeyCode.Alpha3)) animation.CrossFade("scratchidle", 0.2f);
if (Input.GetKeyDown(KeyCode.Alpha4)) animation.CrossFade("walk02", 0.2f);
if (Input.GetKeyDown(KeyCode.Alpha5)) animation.CrossFade("standup", 0.2f);
```

Please note: Idle, Walk and Run are the default names for the three basic animations. However, depending on the character pack, these animation names are sometimes different, and might need to be renamed in the code. In order to get additional animations to work properly with the scripts, you need to declare them using the “WrapMode.Once” call, similar to the code displayed above.

5. Setting up all of your Animations on the Root Motion Character Controller

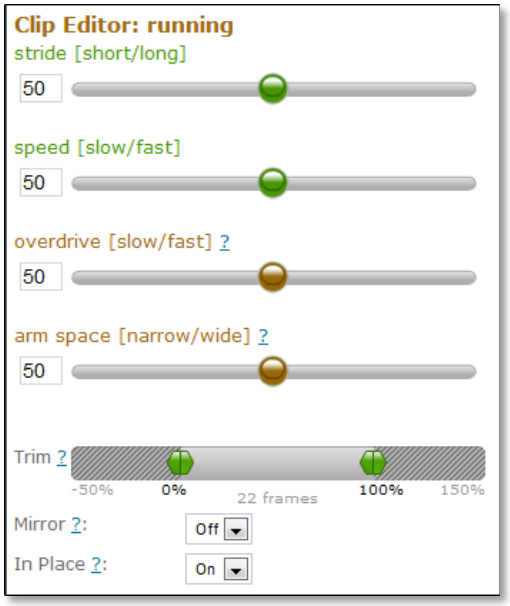
```
// forward movement keys
// ensure that the locomotion animations always blend from idle to moving at the beginning of their cycles
if (Input.GetKeyDown(KeyCode.W) &&
(animation["walk"].weight == 0f || animation["run"].weight == 0f))
        {
                animation["walk"].normalizedTime = 0f;
                animation["run"].normalizedTime = 0f;
        }
        if (Input.GetKey(KeyCode.W))
        {
                targetMovementWeight = 1f;
        }
        if (Input.GetKey(KeyCode.LeftShift) || Input.GetKey(KeyCode.RightShift)) throttle = 1f;

        // blend in the movement

        animation.Blend("run", targetMovementWeight*throttle, 0.5f);
        animation.Blend("walk", targetMovementWeight*(1f-throttle), 0.5f);
```

Please note: The indicated areas above highlight the sections of code which must be changed in order to apply alternate walk or run animations

WHAT'S NEXT?

Find more **characters** & **animations**, and even customize them in real-time: all possible at www.mixamo.com!

Create or Upload a Character → Find and Customize Animations for that Character → Preview in 3D and Download Directly in FBX (Unity naming ready)



Need more Help?

Visit www.mixamo.com/c/unity

Or email us at
support@mixamo.com