For any questions contact me here: pantheradigitalonline@gmail.com

2D Character Controller aims to make it easier and faster for those wanting to make a 2D platformer style game. It's quick and easy to set up. Example objects are also given in the Prefabs folder. This includes two playable characters, a moving platform, and a simple object that follows a target transform when it gets too far away. These just serve as examples for how to use the character controller and its other features in code.

Important to notes:
The character controller uses raycasting for ground checks and other things so if your character will exist in a layer shared with the things you want them to walk on properly then got to Edit->ProjectSettings->Physics2D and down the list look for “Queries Start in colliders” and uncheck the box.

When setting the variables in the editor to adjust how the character moves keep in mind that the character controller will adjust values like gravity and the material on the rigid body during gameplay so set these before running the game in editor as the values may not save properly after.

To make a playable character:
You need an object with at least a Capsule Collider 2d, Circle Collider 2d, or Box Collider 2d attached and a sprite to see it.

- Attach the CharacterMovementScript script to a game object (this script handles the movement of the playable character. More info bellow.)
  The script can be found under MonoBehaviours. Be sure to set the GroundLayers variable to Default or your specified layer for ground objects.

- Next the movement script needs a controller to get input from the user. Under MonoBehaviours >> Controllers find and attach the PlayerControllerScript to the same game object.

- To get things moving the controller needs a scriptable object called ScriptableInputMap that holds information about key presses and their desired function. One can be found in the Scriptables folder or one can be made by right clicking then going to
  Create >> ScriptableObjects >> ScriptableInputMap

(Optional)
If you want to use Unity’s input systems or another input system you can simply modify the CharacterMovementScript to listen for different input. Open the script and locate the

HandleInput() function. There just replace instances of m_controller with the desired input method.

After filling the ScriptableInputMap with your desired controls your character should now move around. With further adjustment of the CharacterMovementScript your playable character should behave how you want them to.

Animation

How you animate is up to you. The CharacterMovementScript uses a struct of variables that communicate the sate of the character that can be used by other classes or in the animator. A simple animator that is set up is provided as well. This can be found under Anim >> Animations >> PlayerObject. A simple script is also used to communicate between the animator and CharacterMovementScript called AnimUpdator. It can be found under MonoBehaviours. Simply attach it to your playable character and fill in the two variables in the inspector.

Additional info

CharacterMovementScript:

-Requires a Rigidbody2D and a Collider2D (supported colliders: CircleCollider2d, CapsuleCollider2D, BoxCollider2D)

-Implements a struct called MovementConstraints that can be used in other scripts to block specific movement types

SiimpleController and SimpleMove scripts are included as examples of how the Controller base class and MovementComponent can be used together in other objects to more easily make moving objects like enemies. (SimpleController found in MonoBehaviours >> Controllers)(SimpleMove found in MonoBehaviours)

Two simple moving platform scripts can be found in the MonoBehaviours folder. MovePlatform moves between two points. MovePlatform_Multiple moves through multiple points. Both scripts require a collider marked as a trigger positioned at the top to allow objects to stay on top of them while they move.

The list of input types can easily be expanded for use in the ScriptablInputMap object. This list can be found in the InputData file (Scripts >> Input) in the struct named InputType.

There are many useful things in place for debugging. The character controller will show extra info about it when the Inspector is set to debug mode. There are also useful gizmos set up as well as others that are still in the code but commented out. Just uncomment them in code to activate them. They can be found in the CharacterMovementScript file, PlayerControllerScript file, and the CollisionCheckComponent file.

Useful tips can be found about the variables in the editor by hovering your mouse over them in the inspector.

For coders who want to expand on or integrate these scripts in their code comments about the code can be found in the files