Table of Contents:

Introduction:

Thank you for purchasing this package!
You will find the following inside of the package:

- A folder containing a scene that demonstrates the controller in its default configuration.
- A folder containing the files that make up the controller, consisting of the following:
    - A Prefab of the controller with the default configuration.
    - Two prefabs that mark the dashing target.
    - Materials for the dashing targets.
    - The scripts that drive the controller.
    - A physics material for the controller that gets modified by its scripts.

More information on how to use this package can be found on the following pages!

Setup and simple Customization

This package is intended to be used without extensive setup.
To start using the controller, simply drop the “PlayerPrefab” into your scene.

To change the behaviour of the controller, you can adjust the values of the scripts attached to the “PlayerPrefab” GameObject (more on this in the next chapter).

Speed-Lines:

You can customize the effect that is displayed at the edges of the screen when running by modifying or removing the “SpeedLines” particle-system, which is a child object to that attached camera.

Dashing Target:

The spheres that by default mark the ending location of the dash can be customized by either modifying the included target prefabs (“DashTargetAir”, ”DashTargetGround”) and the “DashTarget” script in the scripts folder, or by replacing the “Dash Target Marker” references in the attached “PlayerController.cs” script.

The Scripts and deeper Customization

The controller is driven by five important components:

1. The detectors
2. "PlayerInput.cs"
3. "PlayerController.cs"
4. "PlayerMovement.cs"
5. "CameraController.cs"

The detectors:
A set of scripts attached to multiple objects of the controller. They provide data about the environment around the controller.

In their associated scripts you can modify how and what they detect.

PlayerInput.cs:
This script includes methods that looks for player input and interprets the movement keys.

Here you can modify the keybinds.

PlayerController.cs:

The heart of the system. This script contains a state machine that takes information from the detectors and from "PlayerInput.cs" and uses that information to decide what state it should be in, e.g. "Running", "Wallrunning", "Climbing", etc. It then uses methods located in "PlayerMovement.cs" to apply physics forces to the controller.

Here you can modify the controller's (movement-)behaviour.

The state that this script determines can also be used to attach other assets.
For example they can be utilized as triggers for specific animations.

PlayerMovement.cs:

This script contains a number of methods corresponding to specific actions, that apply physics forces to the controller.

There is no customization here, since this function gets all required parameters from "PlayerController.cs"

CameraController.cs:

This script does multiple things that affect the camera attached to the controller.

- Has control over mouselook. All modifications to sensitivity are done here.
- Uses the state from "PlayerController.cs" to determine its behaviour.
- Contains multiple coroutines to move and rotate the camera smoothly during movement transitions.
- Creates the "headbob" motion.
- Controls the "SpeedLines" particle system that is used to create the screen effect.