Parkour System v.1.0.5 Documentation

Thanks for purchasing Parkour System!

The Parkour System is an add-on for UFPS, meant to extend the already impressive UFPS core system with more mobility options for players. With this system, you can create immersive first person parkour game, to more action-frantic double jumping and wallrunning game!

This system is actively being developed for Ascend, and priorities will be focused on features and bugs for our game first, though we plan to extend this to include more first person parkour abilities.

Once again, we sincerely thank you for your support in this system, and I wish you best of luck with your game!

Cheers
Gamers Frontier

Table of Contents

Introducton

How do acParkour system work?

AcParkour relies heavily on Raycast. It does by checking the colliders in front of the character controller, and and from collission. From there, it cast a ray and check for clearance on the top, left or right and then trigger the appropriate States.

Parkour System works by extending the vp_FPPlayerEventHandler and adds this additional States

Dash
DoubleJump
GroundSlide
LedgeGrab

WallJump
WallRun
WallHang

Getting Started

IMPORTANT!
Make sure that you are using ***UFPS 1.5.2*** and above, as the parkour system relies on several new methods and variables that exposed. Once you have imported UFPS, then you can import acParkour into the project and it should work.

VRTraining

The **VRTraining** is a short level that demonstrate the mobility of the ParkourPlayer, by introducing the player to the few parkour abilities in the game.

Playing the Demo

Demo Controls

WASD	Move
C	Crouch/ Groundslide
Space	Jump/ DoubleJump/ WallJump
Left Shift	Sprint *[Hold]* / Dash *[Tap]*

Creating a Parkour Player from scratch

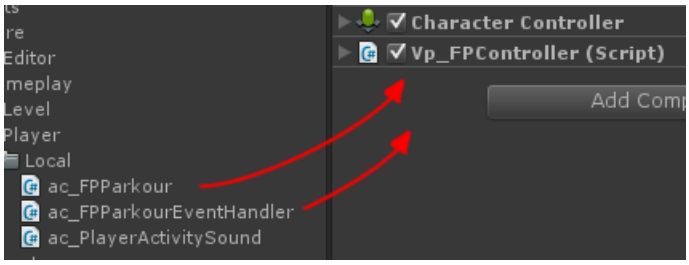
This is the steps to create an entirely new Parkour Player from scratch.
First we'll setup a UFPS basic Player, following UFPS manual.

Basic Setup (From UFPS Manual)

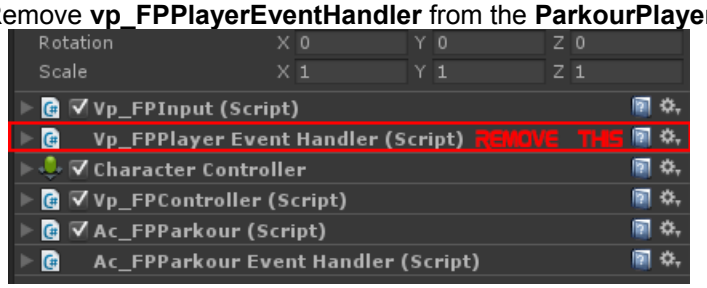
1. Create an empty gameobject and name it **ParkourPlayer**.
2. Place **ParkourPlayer** above any kind of floor with collision.
3. Create another empty gameobject, name it **Camera** and drag it onto the first one so it becomes a child of the ParkourPlayer object.
4. Set the position of the the new child gameobject to (0, 0, 0).
5. In the project view search box, type **vp_fp**.
6. Drag the **vp_FPController** script onto the **ParkourPlayer** gameobject.

7. Drag the **vp_FPCamera** script onto the **Camera** gameobject.

Parkour Setup



1. In the project view search box, type **ac_fp**.
2. Drag the **ac_FPParkour** and **ac_FPParkourEventHandler** script onto the ParkourPlayer gameobject.
3. Right click and Remove **vp_FPPlayerEventHandler** from the **ParkourPlayer.**



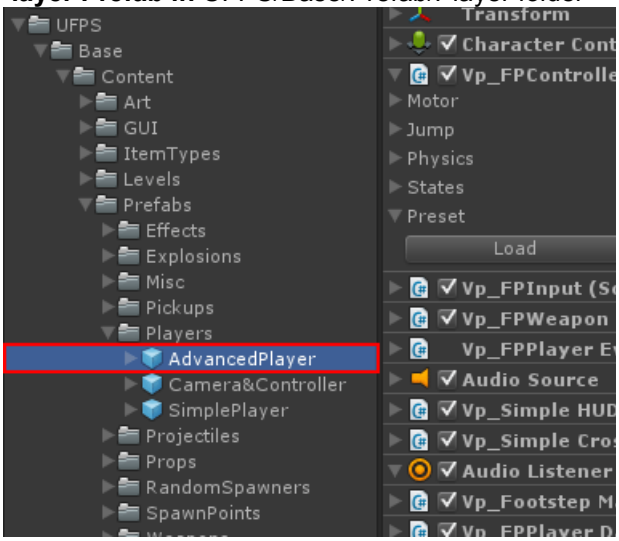
That's about it! You should now be able to run and jump like a ninja!! Try tweaking and experimenting with the editor to tailor suit to your game.

Augmenting Advanced Player Prefab

You could choose to use UFPS AdvancedPlayer prefab to make full use of UFPS inventory system, footstep system, etc.

Advanced Setup

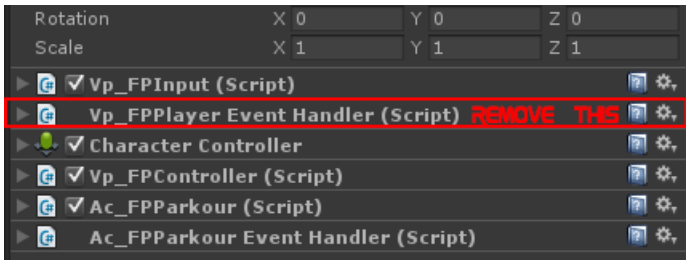
1. Find the **AdvancedPlayer Prefab in** UFPS/Base/Prefab/Player folder



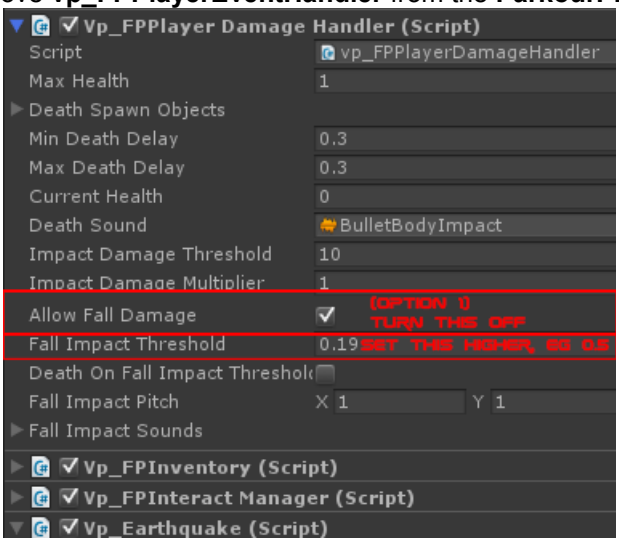
2. Duplicate this and rename it **AdvancedParkourPlayer**

3. Place **AdvancedParkourPlayer** in a new scee with a floor with collision.

4. In the project view search box, type **ac_fp**.
5. Drag the **ac_FPParkour** and **ac_FPParkourEventHandler** script onto the ParkourPlayer gameobject.



6. Right click and Remove **vp_FPPlayerEventHandler** from the **ParkourPlayer.**

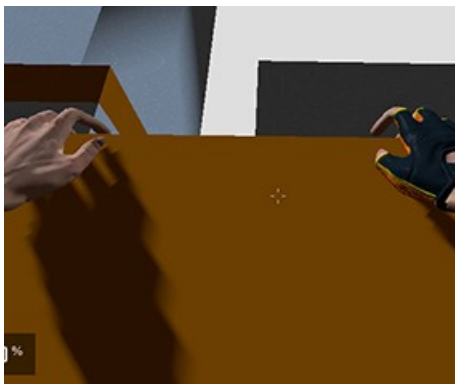


7. Highly recommend to turn off Fall Damage in vp_FPPlayerDamageHandler
8. Alternatively, set the fall impact threshold to 0.5 or so. Tweak this value to best suit your game

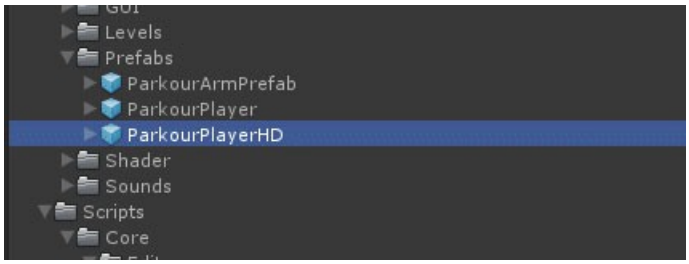
Full Body Awareness

New to acParkour 1.0.5 is the integration of UFPS 1.4.8's Full Body Awareness.
On top of UFPS's vp_FPBodyAnimator, we have mainly added certain keys animation that acParkour will requires for the first person as well third person animation. Included with acParkour is a fully setup and animated character, ParkourGirl that demonstrate how acParkour is combined with UFPS and also full body awareness.

ParkourGirl, from first person view and third person view.



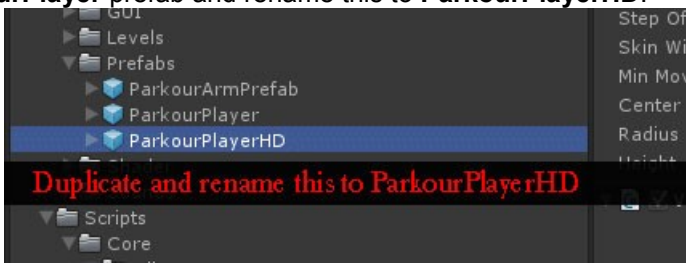
We have setup this as a prefab for ease of use, just drag and drop the ParkourPlayerHD to the scene it should work.



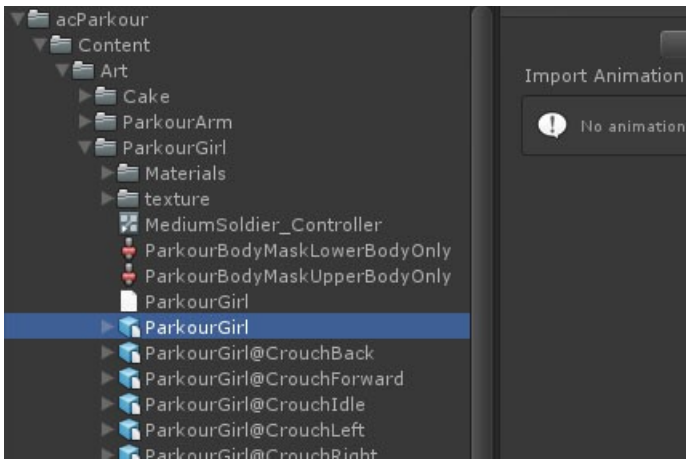
Augmenting with Full Body Awareness

If you wish to add this to an existing playerPrefab, you can just follow the few steps. I will be using the ParkourPlayer prefab that comes with acParkour as an example.

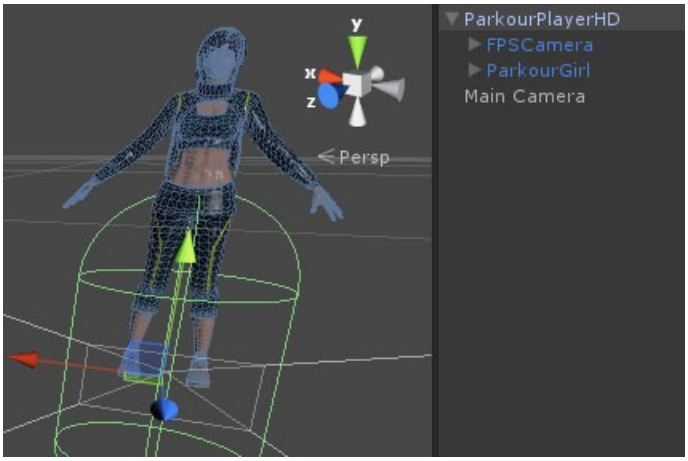
1. Find the **ParkourPlayer Prefab in** acParkour/Content/Prefabs folder
2. Duplicate **ParkourPlayer** prefab and rename this to **ParkourPlayerHD**.



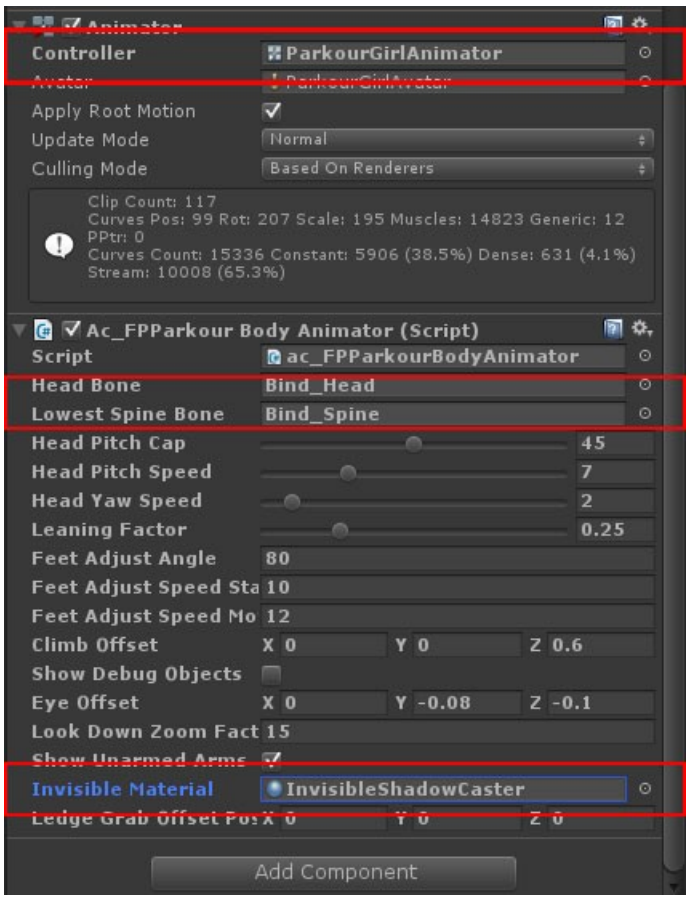
3. Drag this to an empty scene, so we could add **ParkourGirl** player model.
4. Look for the ParkourGirl player model in acParkour/Content/Art/ParkourGirl



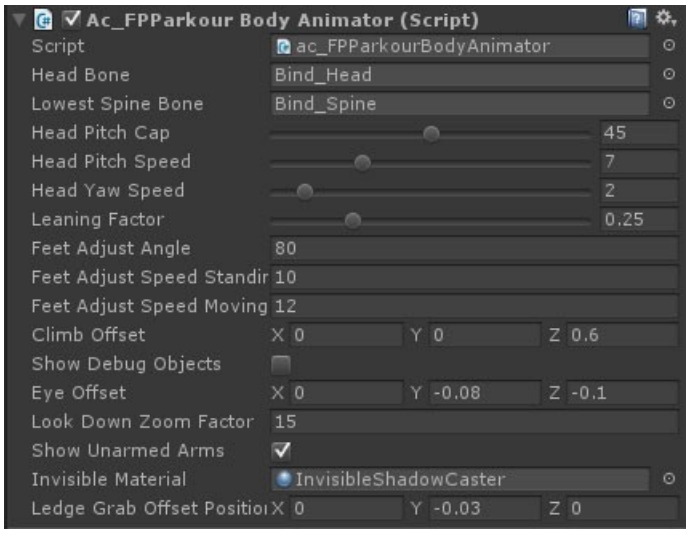
5. Drag **ParkourGirl** into the **ParkourPlayer** in the scene.



6. Next, look for **ParkourGirlAnimator** in acParkour/Content/Art/ParkourGirl
7. Drag this to the **Animator** component of **ParkourGirl** in the scene.
8. In the project view search box, type **ac_fpparkourboyanimator**.
9. Drag the **ac_FPParkourBodyAnimator** script onto the ParkourGirl .
10. Expand the ParkourGirl player model and look for Bind_Spine and Bind_Head
11. Drag this to the respective ac_FPParkourBodyAnimator slot
12. Finally, don't forget to add **InvisibleShadowCaster** Material in the Invisible Material slot.
13. Save the prefab, and that's it, the Full Body Awareness is fully setup.



ac_FPParkourBodyAnimator



This will be similar to UFPS vp_FPBodyAnimator in all except for

Ledge Grab Offset Position

Use this to finetune the position of the arm when ledge hanging.

Flickering Issues with Full Body Awareness

As the full body awareness in acParkour is not using UFPS's second “Weapon Camera' and thus will actually interact with the environment, one of the graphical issues that you will face is 'flickering'.

Shadow

The shadow will be one of major flickering that you will need to wrestle with when working with Full Body Awareness. In general , if you plan to use real time shadows, you will need to tweak the Shadow Bias for a good result.

A lower shadow bias looks great for contact shadows, but it will introduce flickering, where else a higher shadow bias will reduce flickering at the expense of contact shadow accuracy.

Shadow Bias 0.05 Shadow Bias 0.1 Shadow Bias 0.2

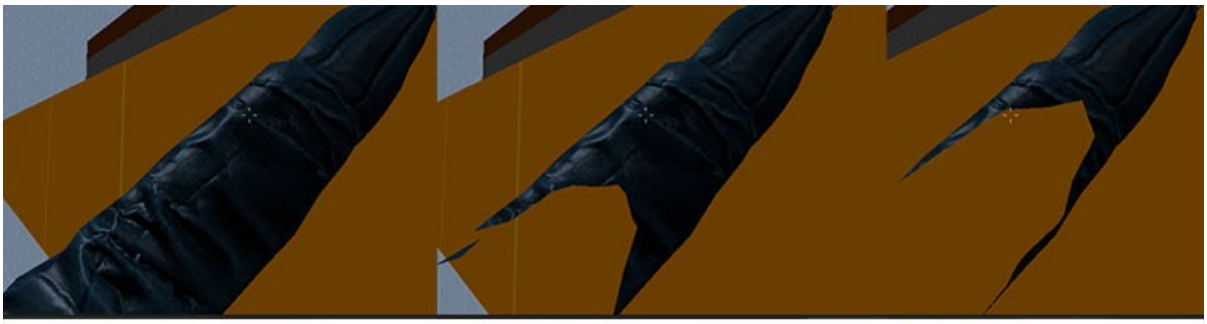
Lower Shadow Bias creates more accurate ‘contact’ shadow but will introduce flickering.

Higher Shadow Bias minimizes flickering but loses contact shadow precision.

Camera Far Clip and Near Clip

Another issue that will contribute to the flickering issues is the camera's near clip and far clip plane. This will depend on the scene size and per level but

A lower shadow bias looks great for contact shadows, but it will introduce flickering, where else a higher shadow bias will reduce flickering at the expense of contact shadow accuracy.



NearClip 0.1 Near Clip 0.18 NearClip 0.2

Lower NearClip won’t ‘clip’ the model but will affect flickering.

Higher NearClip minimizes flickering but will ‘clip’ the model from view

Working in the Editor

The editor for Parkour System aims to be similar to UFPS's editor, so you could utilize sliders for changing the values and save it as text presets.

Dash

Count

How many times you can dash before you runs out. Setting it to 0 will turn it off

Force

The amount of force applied to the character controller in a single frame when player dash.
The dash direction is influenced by InputMoveVector, so it will push the player towards the direction they are moving.

RecoverSpeed

Determines how fast your regain the dash count, up to the initial max dashcount.

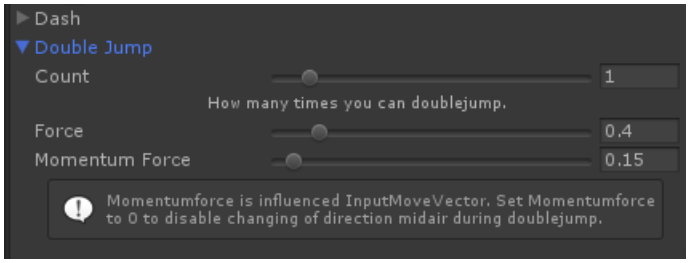
Sensivity

Controls how sensitive the tapping of inputdash is to activate Dash. A smaller number requires a quicker tap, while higher number has a bigger frame of window.

Cooldown

How soon you can dash again. This is to prevent dash from executing too soon, regardless of dash count available.

Double Jump



Count

How many times you can double jump before running out of double jump counters. Set this to 0 to turn off double jump.

Force

The amount of up force applied to the character controller in a single frame when player double jump

Momentum Force

The double jump direction influenced by InputMoveVector. Turn it to 0 to disable changing of direction during double jump.

GroundSlide

Force

The amount of forward force applied to the character controller over 10 frames when player executes ground slide.

Minimum Run Duration

How long to run before you can start ground slide.

Minimum Run Speed

The minimum speed needed continously to be able to start ground slide

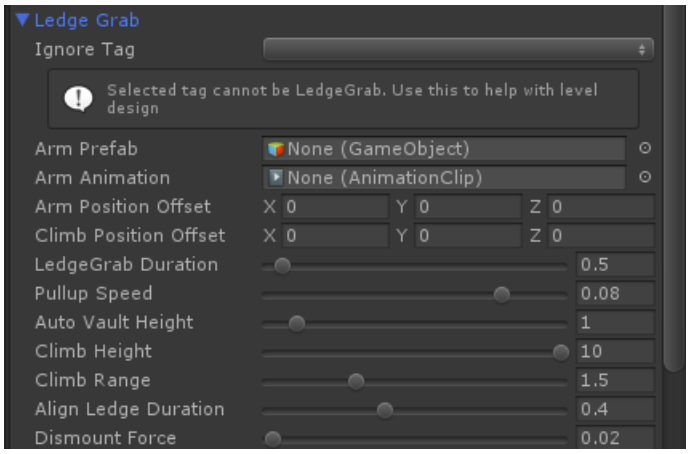
Cooldown

How soon you can groundslide again, to prevent groundsliding too soon

Minimum Speed

This checks for player's velocity when ground sliding, and stops groundsliding when the player's velocity drops below this minimum speed check. This will prevent players from sliding up a ramp at the same speed as sliding down the ramp.

LedgeGrab



Ignore Tag

Any objects with this tagged will not allow ledge grabbing. Use this for more control in level design.

Arm Prefab

Prefab arm to spawn for ledge grab animation. If there's nothing here it will be skipped.

Arm Animation

The corresponding animation for Prefab arm to play during ledge grab.

Arm Position Offset

Offset the arm animation's position based on camera's orientation. Use to fine tune the spawn position of the arm.

Head Offset

Offset the raycast based on character controller's height. Setting this value will determine how high the raycast should start for checking ledges. A small values works great for interior

Climb Position Offset

Offset's the target climb position by this amount. A little bit on the Y axis should help clear any glitch during ledge grab.

Hang Position Offset

Offset's the target hang position by this amount. A little bit on the Y axis should help clear any glitch during ledge hanging.

1P Hang Pos Offset

Offset's the player model's hang position by this amount. This is on top of the hang position offset but is meant for visual purpose.

3P Hang Pos Offset

Offset's the player model's hang position by this amount. This is on top of the hang position offset but is meant for visual purpose.

Ledgegrab Duration

How fast to climb up the ledge in seconds.

Pullup Speed

This handles how fast to pull up in ledge hanging mode.

Auto Vault Height

The minimum height to automatically vault over this obstacles. This is to simulate automatically stepping over any small obstacles. This will trigger if it's over this range but not within the Climb Range. **Note this will skip ledge hanging!**

Climb Height

The minimum distance to be able to ledge climb. This is calculated from around approximately “chest level, then if the range from this to the top of the of the collider is within this range, will then proceed to ledge grab.

Climb Range

This is the range of the raycast forward for detecting if the obstacles can be ledge climbed. Set this to 0 to disable ledgegrab.

Align Ledge Duration

How quickly to align the character controller to face the ledge at the start of ledgegrab, in seconds.

Dismount Force

The amount of force applied to the character to help push the player away from the obstacles. A small amount helps to prevent glitching.

TIPS
Make full use of the Ignore Tag. By default, acParkour will try to ledgegrab everything, so by strategically applying Ignore Tags, you can control certain areas that will prevent players from climbing that area, for example some sort of barbed electric fence.

Ledge Shimmy Distance

How far the ledge shimmy can 'travel'. Keep this to a low value for a more instant transition.

Ledge Shimmy Duration

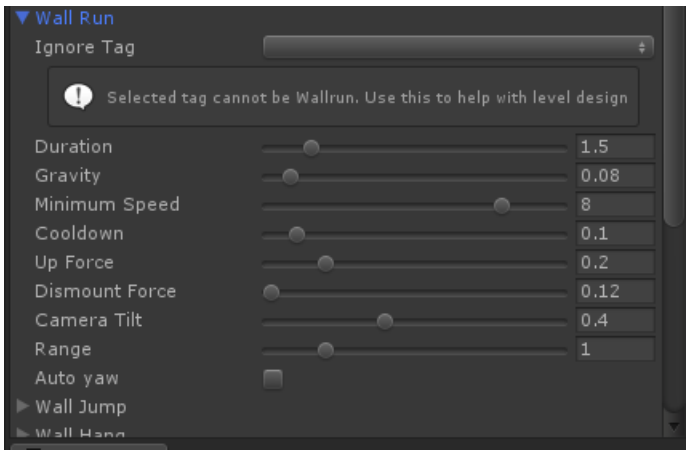
How long to shimmy duration will happen. Keep this to a low value for a more instant transition.

Ledge Transition Distance

How long to transition edges. This is used when doing 90 degree turns, which will override the above.

TIPS
Make full use of the Ignore Tag. By default, acParkour will try to ledgegrab everything, so by strategically applying Ignore Tags, you can control certain areas that will prevent players from climbing that area, for example some sort of barbed electric fence.

WallRun



Ignore Tag

Any objects with this tagged will not allow wallrunning. Use this for more control in level design.

Infinite Wallrun

Turn this on if you want to cling to the wall forever. For best result, turn Gravity and Up Force to 0.

Duration

How long does the wallrun lasts in seconds. Set to 0 to turn off WallRun.

Cooldown

How long before you can wallrun again after a wallrun ends.

Up Force

Force to push the character upwards a little bit during the wallrun.

Dismount Force

Pushes the player away from the wall at the end of the wallrun.

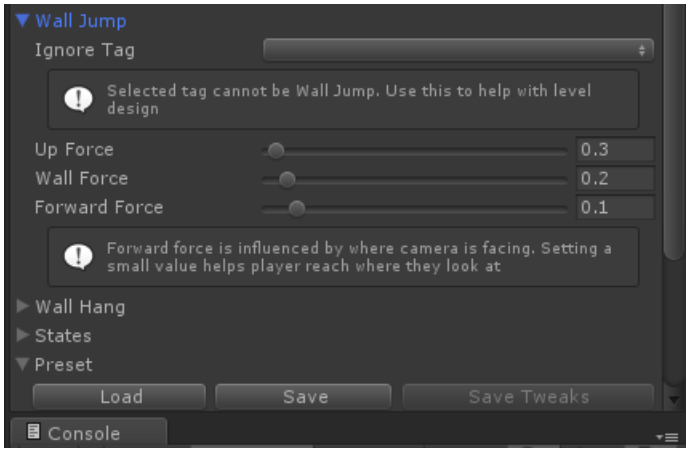
Camera Tilt

Tilts the camera while wallrunning. Higher value tilts the camera more. Turn this to 0 to disable it.

Range

This is the range of the raycast forward for detecting if the wall can be wallrun.
Setting this to 0 to will disable wallrun.

Wall Jump



Ignore Tag

Any objects with this tagged will not allow walljump. Use this for more control in level design.

Up Force

The up force that is applied in one frame to the player.

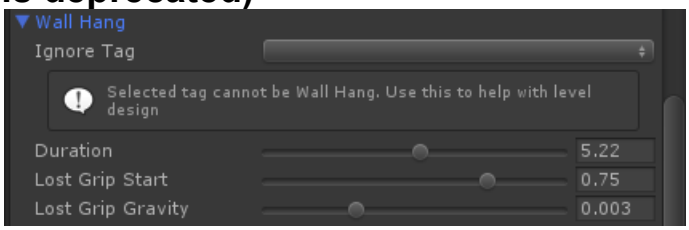
Force

How strong the jump force is perpendicular to wall normal.

Forward Force

The force that pushes towards camera forward.

Wall Hang (This is deprecated)



Ignore Tag

Any objects with this tagged will not allow wall hang.

Duration

How long the wall hang last in seconds.Set to 0 to disable Wallhang.

Lost Grip Start

At what percentage should lost grip start happening. 0.5 would be halfway for example.
Set this to 1 to disable this feature.

Lost Grip Gravity

The gravity force applied when losinggrip. Use this to simulate losing grip and slipping.

Known Issues

1. ~~Ledgegrab currently does not work well within a very small and confined space, this will be addressed in the next few releases.~~ Addressed in version 1.0.5, with using HeadOffset for raycasts
2. acParkour by default will work with all colliders, but it is highly recommended to utilize the “Ignore Tag” in your levels, or players will be able to climb everywhere and anything.
3. Wallrun and vp_Grab objects are currently very fidgetable but it can work. Try to set the Z distance offset to be a little bit more further then usual, and set the tag on the vp_Grab object to be ignored by WallRun, LedgeGrab and WallHang.
4. WallRun, LedgeGrab and WallHang currently does not work with Moving platforms. Will be address in subsequent releases.

Help and Contact

Drop us an email if you have any issues. We will get back to you within a week.

Please provide us your **invoice number** for email support.

Please provide a brief description of your problem.

support@gamersfrontier.my