MANUAL

DOCUMENTATION

Kreshnik Halili, MSc

HOAX GAMES
SOCIAL TRUTH FIGHTER

TABLE OF CONTENTS

HOAX GAMES
SOCIAL TRUTH FIGHTER

HOAX GAMES
SOCIAL TRUTH FIGHTER

QUICK START

INSTALLING ISTEP INTO A WORKING PROJECT

To get started with iStep for a working project, you simply import iStep through the package manager into your project. Please don't install iSteps demo content into a working project except you exactly know what you are doing. To get iStep running, add the FootIK script to your character (at the same level where your character has the Animator component attached to it) and make sure you have IK-pass enabled in your characters Animator Controller (for the Base Layer by default). For more details on custom character controller integration please continue with reading **CUSTOM CHARACTER CONTROLLER INTEGRATION** or by following the associated video tutorials. Please read the section **MOST IMPORTANT SETTINGS FOR CUSTOMIZATION** under **FURTHER INFORMATION** as well because it contains important information that will help you get the most out of iStep 😊.

INSTALLATION
URP DEMO

INTRODUCTION

To get started with the UPR demo scene you have first to import the corresponding package.

From the folder “Assets_HoaxGames\iStep\” import the package:

“iStep_Demo_URP.unitypackage”

REQUIRED PACKAGES

The demo scene requires the following Unity packages to be installed:

- *Cinemachine*
- *Input System*

Typically, the within the package included “Starter Assets” subset will install those dependencies automatically through the included Editor

scripts. Should there be any issues please try to resolve them by going to “Tools” → “Starter Assets” → “Reinstall Dependencies”.

Alternatively, you can install the required packages (dependencies) manually using the “Package Manager”. To do so go to “Window” → “Package Manager” and change the packages to show settings to “Packages: Unity Registry” in the top left corner of the “Package Manager” window (the setting is a drop-down menu).

Select “Cinemachine” from the list and press “install” (bottom right corner).

Select “Input System” from the list and press “install” (bottom right corner).

If you went for the manual installation, please make sure that under “Edit” → “Project Settings” → “Player” → “Other Settings” you have “STARTER_ASSETS_PACKAGES_CHECKED” added to the “Scripting Define Symbols”.

ANIMATOR SETTINGS

iStep requires an IK pass to be enabled in the Animator. To do so open the imported demo scene (“iStepShowcaseDemoURP.unity”) and select the gameobject “PlayerArmature” from the hierarchy. From the “Animator” component attached to “PlayerArmature” double-click the object assigned under the “Controller” variable (this is the corresponding “AnimatorController” object called “StarterAssetsThirdPerson”). This should open the players “AnimatorController” in the “Animator” window. Alternatively, you can search for “StarterAssetsThirdPerson” in the “Project” tab and double-click the found “AnimatorController”.

In the “Animator” window press the settings icon next to “Base Layer” under the “Layers” tab and activate the “IK pass” for the “Base Layer”.

URP SETTINGS

To make sure everything works flawlessly (like for example the footstep decals) one last step is required.

Go to “Edit” → “Player Settings” → “Quality” and set the “Render Pipeline Asset” to “UniversalRP-HighQuality” from “Assets_HoaxGames\iStep_Demo_URP\Settings\”.

INSTALLATION
HDRP DEMO

INTRODUCTION

To get started with the HDPR demo scene you have first to import the corresponding package.

From the folder “Assets_HoaxGames\iStep\” import the package:

“iStep_Demo_HDRP.unitypackage”

REQUIRED PACKAGES

The demo scene requires the following Unity packages to be installed:

- *Cinemachine*
- *Input System*

Typically, the within the package included “Starter Assets” subset will install those dependencies automatically through the included Editor

scripts. Should there be any issues please try to resolve them by going to “Tools” → “Starter Assets” → “Reinstall Dependencies”.

Alternatively, you can install the required packages (dependencies) manually using the “Package Manager”. To do so go to “Window” → “Package Manager” and change the packages to show settings to “Packages: Unity Registry” in the top left corner of the “Package Manager” window (the setting is a drop-down menu).

Select “Cinemachine” from the list and press “install” (bottom right corner).

Select “Input System” from the list and press “install” (bottom right corner).

If you went for the manual installation, please make sure that under “Edit” → “Project Settings” → “Player” → “Other Settings” you have “STARTER_ASSETS_PACKAGES_CHECKED” added to the “Scripting Define Symbols”.

ANIMATOR SETTINGS

iStep requires an IK pass to be enabled in the Animator. To do so open the imported demo scene (“iStepShowcaseDemoHDRP.unity”) and select the gameobject “PlayerArmature” from the hierarchy. From the “Animator” component attached to “PlayerArmature” double-click the object assigned under the “Controller” variable (this is the corresponding “AnimatorController” object called “StarterAssetsThirdPerson”). This should open the players “AnimatorController” in the “Animator” window. Alternatively, you can search for “StarterAssetsThirdPerson” in the “Project” tab and double-click the found “AnimatorController”.

In the “Animator” window press the settings icon next to “Base Layer” under the “Layers” tab and activate the “IK pass” for the “Base Layer”.

INSTALLATION
BUILD-IN DEMO

INTRODUCTION

To get started with the Build-In demo scene you have first to import the corresponding package.

From the folder “Assets_HoaxGames\iStep\” import the package:

“iStep_Demo_BuildIn.unitypackage”

REQUIRED PACKAGES

The demo scene requires the following Unity packages to be installed:

- *Cinemachine*
- *Input System*
- *Post Processing*

Typically, the within the package included “Starter Assets” subset will install those dependencies automatically through the included Editor

scripts. Should there be any issues please try to resolve them by going to “Tools” → “Starter Assets” → “Reinstall Dependencies”.

Alternatively, you can install the required packages (dependencies) manually using the “Package Manager”. To do so go to “Window” → “Package Manager” and change the packages to show settings to “Packages: Unity Registry” in the top left corner of the “Package Manager” window (the setting is a drop-down menu).

Select “Cinemachine” from the list and press “install” (bottom right corner).

Select “Input System” from the list and press “install” (bottom right corner).

Select “Post Processing” from the list and press “install” (bottom right corner).

If you went for the manual installation, please make sure that under “Edit” → “Project Settings” → “Player” → “Other Settings” you have “STARTER_ASSETS_PACKAGES_CHECKED” added to the “Scripting Define Symbols”.

ANIMATOR SETTINGS

iStep requires an IK pass to be enabled in the Animator. To do so open the imported demo scene (“iStepShowcaseDemoBuildIn.unity”) and select the gameobject “PlayerArmature” from the hierarchy. From the “Animator” component attached to “PlayerArmature” double-click the object assigned under the “Controller” variable (this is the corresponding “AnimatorController” object called “StarterAssetsThirdPerson”). This should open the players “AnimatorController” in the “Animator” window. Alternatively, you can search for “StarterAssetsThirdPerson” in the “Project” tab and double-click the found “AnimatorController”.

In the “Animator” window press the settings icon next to “Base Layer” under the “Layers” tab and activate the “IK pass” for the “Base Layer”.

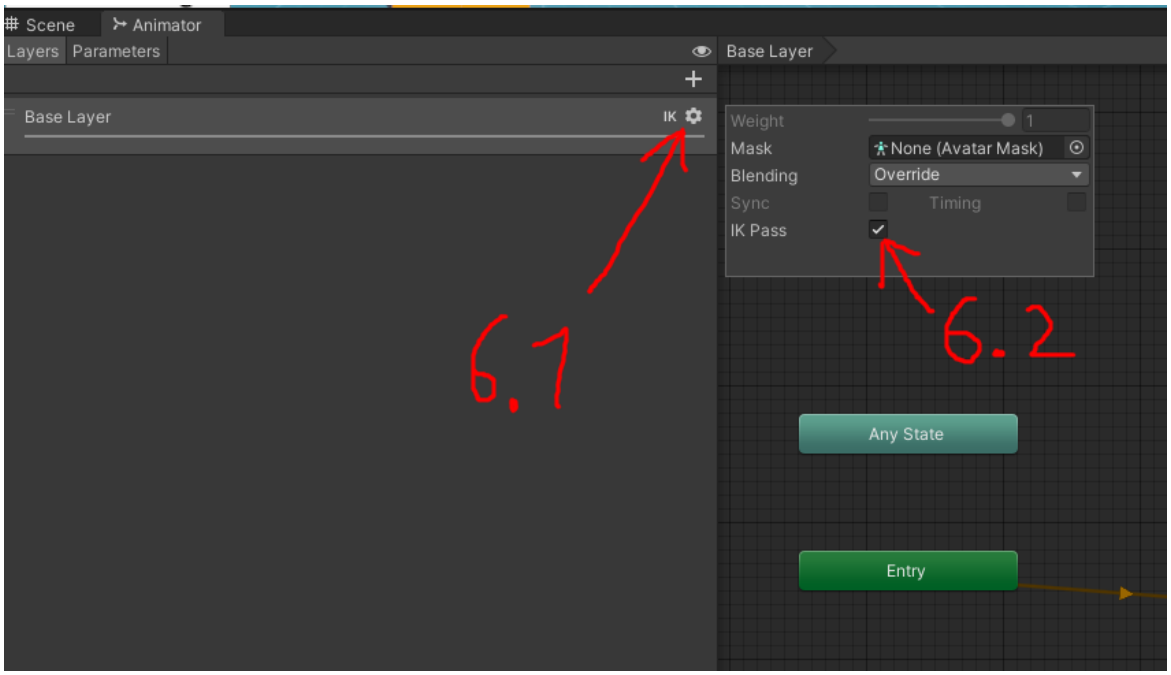
CUSTOM CHARACTER CONTROLLER INTEGRATION

STEP FOR STEP INSTRUCTIONS

Step 1: Open the corresponding Project.
Step 2: Select the gameobject representing your character controller (can be a player or an AI controlled character controller).
Step 3: Select the (child) gameobject of your character controller object-hierarchy having the Animator component attached to it.
Step 4: Add the component “FootIK” to it.
Step 5: Open the assign AnimatorController in the “Controller” variable of the Animator component by double-clicking it (the AnimatorController should open up in the Animator window).
Step 6: Make sure under the Base Layer Settings, IK pass is enabled.
Step 7: You are done. Press play to check if everything works as

expected.

Step 8: Configure your “FootIK” settings to your likings if necessary.



SOLVING FOR CROUCHING

In case your custom character controller supports crouching, you have two options to make iStep switch to a crouching specific behaviour back and forth.

The first and most simple option is to add your crouch-specific animation state name to the “Force Activate Crouching Behaviour” list in the inspector of the FootIK script.

The second option is to use the function “setActivateCrouchingBehaviour” from code to trigger the crouch-specific behaviour.

SOLVING FOR SPECIAL CHARACTER CONTROLLER ACTIONS

In case your custom character controller has any actions where iSteps foot placement solver would interfere, you can smoothly disable iStep with the following options (smoothly in this regard means that a smooth transition is applied):

Option 1: Add your action specific animation state name to the “Force Invalid On Animator State” list in the inspector of FootIK.

Option 2: Do it from code by calling “setIsValidAndShouldCheckForGrounded” function with the respective parameter.

SOLVING FOR ANIMATION RIGGING

In case your custom character controller makes use of Animation Rigging and therefore of IK solutions other than iStep, there are three use-case scenarios to consider:

1) *The constraints target(s) are inside of the Rig (Hierarchy) and don't require any additional adaption*
2) *The constraints target is outside of the Rig (Hierarchy) – like for example a global positioned look-at target*
3) *Custom, non-Animation Rigging solution*

Before you continue reading, you may should have a look at the corresponding video tutorial and sample project for Animation Rigging – explaining the content below in detail.

Animation Rigging Video Tutorial: https://youtu.be/KGn3csXMEms

Animation Rigging Sample Project: https://hoax-games-downloads.web.app/istep/iStepAnimationRiggingExample.unitypackage

1: If you use a TwoBoneIKConstraint (for example) and your IK target and IK hint target are children of the TwoBoneIKConstraint or Rig, they are considered relative to the Rig and will get automatically adapted for body position offsets that happen later. Therefore, you don't have to apply any additional offsets to the targets for this use-case and iStep should be compatible out-of-the-box.
If your targets however are not parented to the constraint or Rig and therefore not relative to the Rig (like for example you want to touch a world-element in the scene with an absolute position) or you happen

to use a constraint that don’t apply body position offsets to its targets, they fall under point 2).

For 2) you typically have to adapt the corresponding constraints target position by the FootIK scripts **fullBodyOffset** property. When using Animation Rigging you typically want to use the negated fullBodyOffset value applied to your global targets. The reason for that is because the Animation Rigging constraints are executed **before** iSteps FootIK script and therefore aligned to the global targets before iStep may adapt the characters body position. Therefore, to negate this effect, the negative fullBodyOffset from the FootIK script has to be applied to your constraint’s global targets. The easiest way to achieve this with Animation Rigging is by creating a child for your previous target, attach the **GlobalIKTargetCorrector** helper script to the child and set the child as the new target of your constraint (if you manipulated the previous targets position with custom code, you should keep manipulating the same (previous) target with your code and not the new child).
While for absolute positioned targets you have to use the negated fullBodyOffset, there are use-cases where you have objects parented to your Rig where Animation Rigging doesn’t apply the fullBodyOffset automatically. In this case you have to create a child element and attach the GlobalIKTargetCorrector to the child as well, but use the positive fullBodyOffset instead.

For 3) you should use the FootIKs script **fullBodyOffset** property as you see fit to achieve the behavior you require (or make correct use of the **GlobalIKTargetCorrector** helper script).

Reminder: Don’t forget to assign your characters FootIK component to your GlobalIKTargetCorrector when you use it!

TROUBLE-SHOOTING

COMMON ISSUES

1) ***Assertion failed on expression: 'CompareApproximately(det, 1.0F, .005f)' UnityEngine.Quaternion:FromToRotation (UnityEngine.Vector3,UnityEngine.Vector3)***

   In case that you run into this issue, please make sure the colliders (for example a BoxCollider) in your scene (or their parents) don't have scale or size 0 in the X or Y or Z axis. If one of the axes is 0, Unitys collision detection with raycasts/boxcasts/spherecasts can't successfully calculate surface-normals and therefore the above assertion will be thrown.

2) ***Assertion failed on expression: 'IsFinite(distanceAlongView)'***

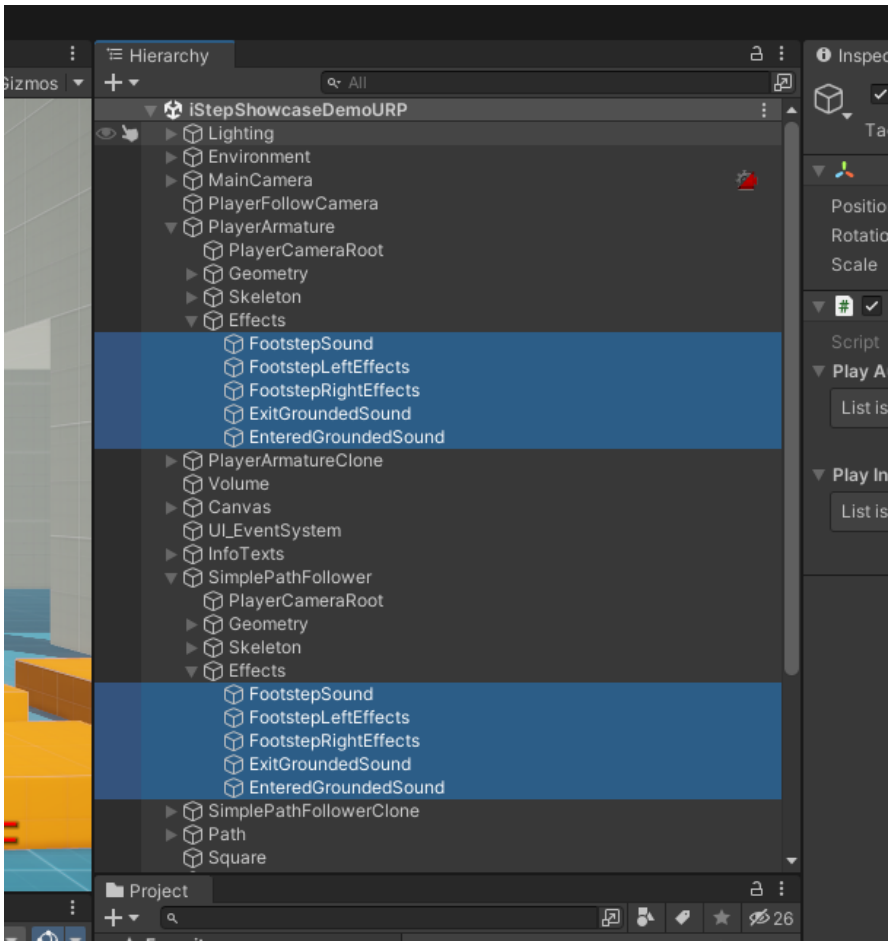
   Same as 1)

FURTHER INFORMATION

GENERAL INFORMATION

All serializable parameters in the FootIK script are documented with tooltips. Please read the tooltips for more information on the respective parameters. Also, when optimizing the FootIK parameters, it is strongly recommended that you start with the default settings first and optimize from there. The fastest way to optimize the most important settings is to go into Play-Mode and having “Minimum Movement Threshold” set to zero (temporarily – after finding the best settings, set this parameters value back to your preferred value). The FootIK script has an open interface to its footstep events. Please check out the included demo scene(s) to understand how the events were used in connection with the EffectsPlayer script to play audioclips as well as instantiate footstep effects. In the demo scene(s) all footstep effects are defined in the gameobjects selected in the above reference image.

Please note that the Effects Player script is just a way to use iSteps events but it is not limited by it.



MOST IMPORTANT SETTINGS FOR CUSTOMIZATION

- **Ik Max Correction**
  Sets how much you can step up visually.
- **Ik Foot Height**
  Sets the height of your foot.
- **Body Position Max Correction**
  Sets how much you can step down visually.
- **Check Grounded Distance**
  This parameter is used by iStep to determine if the IK placement algorithm should check for new foot IK positions/rotations or not. Only when grounded is true the foot IKs will adapt accordingly. This value should typically be set as high as "Body

Position Max Correction" or higher. Don't misunderstand this parameter with your character controllers internal grounded state.

- **Compute IK Hint Position Strategy**
  This parameter controls how IK hint position is being calculated. It is strongly recommended to set this to BONE_BASED in case you use Animation-Layer-Blending (with Avatar Masks).
- **Ik Upwards Pull Back Down Extrapolation**
  This parameter will help pulling the feet back down to the ground (when the feet is floating in the air) when running upwards on slopes.
- **Increase Body Position Max Correction With Forward Foot Distance**
  This parameter dynamically increases the max body position correction based on the distance the feet are apart from each other. Very useful to keep Body Position Max Correction as high as needed but as low as required while still adapting when necessary (for example on steep slopes).
- **IK Placement Stiffness**
  This parameter controls how smooth/stiff IK placement should be.
- **Body Stiffness**
  This parameter controls how smooth/stiff body placement should be.
- **Minimum Movement Threshold**
  Increase this parameter in case your Idle-Animations have large foot-micro-movements to avoid potential repeated up and down alignment changes on borders/edges.

VIDEO TUTORIALS

Getting Started with HDRP:
https://youtu.be/mEsm9SUTeV0

Getting Started with URP:
https://youtu.be/ycHZdAyiTYQ

Getting Started with Build-In:
https://youtu.be/diX8PBTkdRg

Custom Character Controller Integration:
https://hoax-games.web.app/istep/#customcharactercontroller

iStep Settings Explanation:

https://youtu.be/pS8lrgG9C44

Emerald AI Integration:

https://youtu.be/itCvnUkT4K4

https://youtu.be/GJKHdb9WCvg

Synty Studios Character Integration:

https://youtu.be/cGFuwfFz_O4

VRoid Studio Character Integration:

https://youtu.be/6AoNeqyKoBE

https://youtu.be/rCkcaDsH3gE

Animation Rigging:

https://youtu.be/KGn3csXMEms

CONTACT

E-mail: k.hoaxgames@gmail.com

Website: https://hoax-games.web.app