
SEAFOOD
SALON
24
OPEN
Coffee
Kronnect
BUILT-IN/URP/HDRP
SPLIT SCREEN PRO

USER GUIDE

Contents

Introduction

Thank you for purchasing!

Split Screen Pro is a camera composing system that enables screen sharing between two players or targets. This asset can automatically detect when the two targets are far from each other and show a dynamic split line which separates the view in two halves, one per each target or player.

The asset includes many options that allows you to configure both the behaviour of the split and the appearance producing an effect like those shown in AAA games.

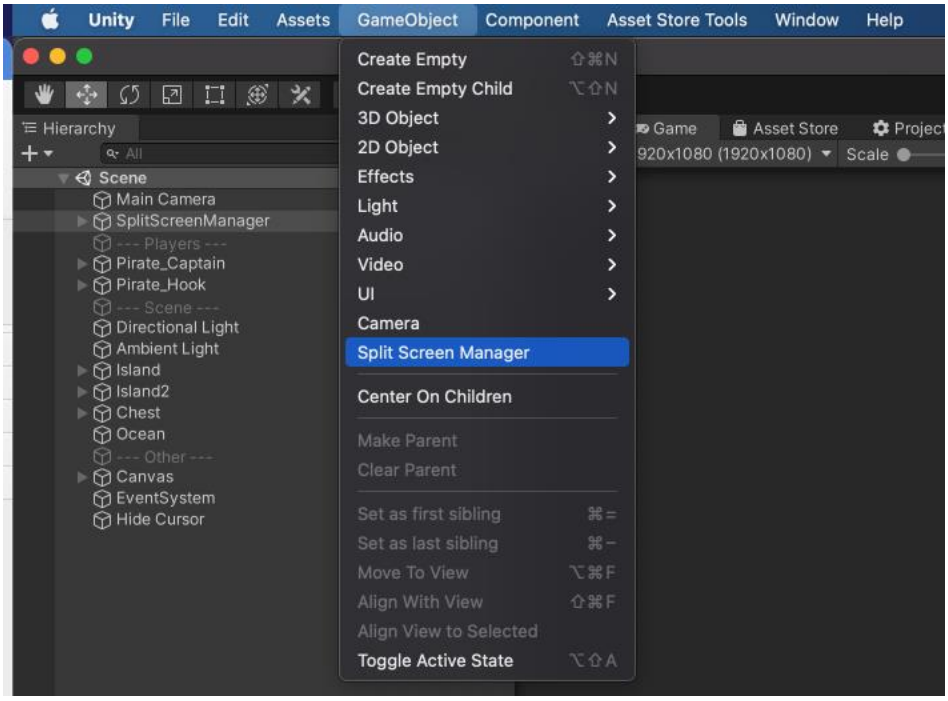
We hope you find the package easy and fun to use. Feel free to contact us for any enquiry.
Visit our Support Forum on https://kronnect.com for suggestions or support requests.

Kronnect Technologies
Email: contact@kronnect.com
Support Forum: https://www.kronnect.com/support

Quick Start and Demo Scene

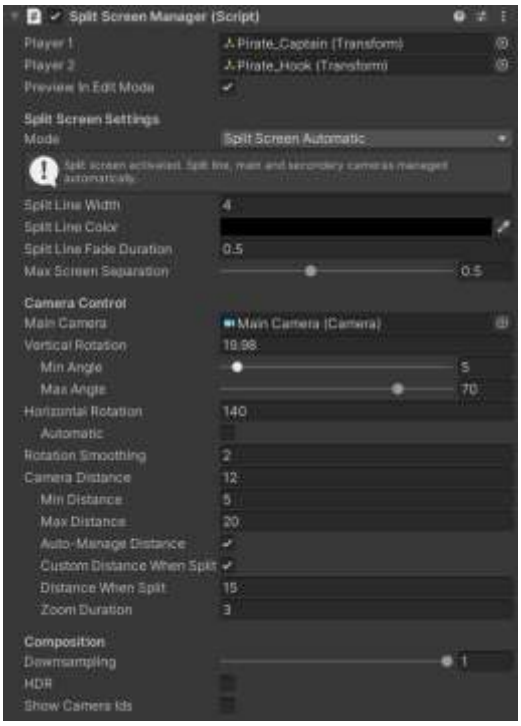
Inside the *demo* folder, you will find a sample scene of a small island two pirates. Move around each pirate to experience how the split system operates. Select the *SplitScreenManager* gameobject to explore the different options shown in the inspector and learn about its effects. Continue reading this document for more details.

To start using the asset in your scene, just select GameObject -> Split Screen Manager from the Unity top menu. After this, you should see a SplitScreenManager gameobject in your scene, ready to be configured.



Setting up Screen Split Manager

The Split Screen Manager requires only 3 mandatory fields to be assigned before it can run properly:



- **Player1** & **Player2**: assign the two players or targets that you want to follow (*required*). You can assign these fields using scripting as well at any time.

- **Main Camera** (under Camera Control): the asset will use the camera tagged as *MainCamera* by default, but you can assign any other camera here. This camera should follow player 1.

More options:

- **Preview In Edit Mode**: when enabled, the split screen system will run also in edit mode.

- **Split Screen Mode**: choose between "Off" (disables the system completely), "Follow Player 1" (main camera will follow player 1, no split), "Follow Player 2" (main camera will follow player 2, no split), "Split Screen Fixed" (split system is enabled but cameras can move and look anywhere, no player follow operates), "Split Screen Automatic" (split system is enabled when players separate and cameras are managed in automatic mode) and "Split Screen Independent" (split screen is enabled and cameras rotation and distances can be controlled).

- **Split Line Width**, **Color** and **Fade Duration**: lets you customize the appearance of the split line.

- **Max Screen Separation**: a value of 0.5 (default) will keep the targets or players in the middle of each correspondent split screen. A value of 0.9 will allow them to approach to the edge of the screen before the split line appears.

- **Vertical Rotation, Min Angle, Max Angle**: determines the default vertical rotation for the cameras as well as a clamping range.

- **Horizontal Rotation**: let you choose the default horizontal angle. The "**Automatic**" field will make the cameras look perpendicular to the axis formed by the two targets producing a vertical split line on the screen.

- **Rotation Smoothing**: makes camera rotation soft. A value of 0 means no smoothing. The higher the value, the fastest the rotation.

- **Camera Distance**: default camera distance. This value can be clamped by the **Min Distance** and **Max Distance** parameters. When "**Auto-Manage Distance**" is enabled, the camera will zoom out when players start separating until the "**Max Distance**". At this point, the screen is split.

- **Check Camera Occluders**: when this option is enabled, the camera will be moved towards the player when some object (like a wall) blocks the view. This occlusion test relies on colliders so make

sure blocking objects have a mesh collider attached if you want to use this option. Note: when the camera is occluded, it needs to be pushed toward the player character – this will always trigger the split screen mode.

- **Custom Distance When Split** and **Distance When Split**. If the first parameter is enabled, when the split screen is activated, the camera zoom will transition to the **Distance When Split**, instead of the regular **Camera Distance**. This allows you to specify a different camera distance when players are on the same screen or when players are in split screen mode. The "**Zoom Duration**" option let you specify the duration of the zoom transition.

- **Split Audio**: if enabled, the system will ensure audio is emitted from the nearest player. Read the chapter "Splitting Audio" for more details.

- **Downsampling**: when the split screen is activated, two cameras render the scene from each player/target perspective. You can downsample the rendering to improve performance. This option can be useful on less powerful devices.

- **HDR**: enabling this option will compose the two images from each camera on an HDR-enabled render buffer. Enable only if you wish to apply some kind of post-processing on the final image.

- **Show Cameras Ids**: this option will display a label for each camera on each split side.

Default Camera Controller

The SplitScreenManager prefab includes a simple default camera controller. This script reads mouse movement and rotates camera accordingly. Feel free to use, edit or replace this script with your own.

Scripting Support

You can change any SplitScreenManager property using scripting through the SplitScreenManager instance. Example:

```
SplitScreenManager.instance.cameraVerticalRotation = 15;
SplitScreenManager.instance.cameraDistance = 40;
SplitScreenManager.instance.player1 = <my player1 transform>
SplitScreenManager.instance.mainCamera = <my camera>
SplitScreenManager.instance.splitLineColor = Color.blue;
…
```

(any property on the SplitScreenManager inspector can be set this way)

Technical Details

The system uses a secondary camera (child of the SplitScreenManager gameobject) to render the scene from the second target or player. A custom material/shader combines both images, from the main camera and the second camera, and displays the result in a canvas that's rendered by the SplitScreenManager camera.

The second camera will copy all parameters from the main camera. If you have some post-processing affecting the main camera, you can add those post-processing script to the second camera as well (in built-in pipeline).

You can also add custom post-processing effects to the SplitScreenManager camera. For example, if you use Beautify, you could add Beautify to the SplitScreenManager camera, instead of adding it to the main camera.

Splitting Audio

Since version 2.0, Split Screen Pro provides support for splitting audio. Since Unity only allows one single Audio Listener in the scene, there're two options regarding audio splitting:

- Use a completely different sound system for your game, like FMOD, which supports multiple listeners. In this case you would replace your AudioListener and AudioSources with the components provided by the new system.
- … or enable the "Split Audio" option provided by the Split Screen Manager.

Playing a sound at any position with Split Screen Pro

Use **SplitScreenManager.PlayClipAtPosition(clip, position, volume)** method.
The system will ensure the sound is played from the nearest player position.

Setting up Audio Sources to work with Split Audio option

Add the component "**SplitAudioSource**" to your AudioSources. A children gameobject with a mirrored audio source will be created. Split Screen Pro will mute the original audio source and manage the position and state of the mirrored audio sources, so they play the sound from the nearest player.

If you're using an object pooling system for certain objects that are created frequently during the game, edit your prefab containing the AudioSource and add SplitAudioSource to your gameobject. This will create the children/mirrored AudioSource as part of the prefab, so it benefits from your object pool.

Rendering Pipelines Considerations

The system has been designed and tested to work on the three main pipelines: built-in, URP and HDRP.

However, in HDRP it's necessary to set the VolumeMask property of the SplitScreenManager's camera to Nothing. This is important so no default post-processing is applied redundantly to the compositing camera (the SplitScreenManager camera).

If you have any question or suggestion, please don't hesitate to contact us.