Universal Device Preview

v1.11.0

for Unity 2018.4 to 2022.2

January 20, 2023

Contents

Thank you for purchasing *Universal Device Preview*! I wish this package will meet your needs and expectations. Please do not hesitate to contact me if you have a feature request, or any question, issue or suggestion.

Arnaud,
support@wildmagegames.com

1 OVERVIEW

1.1 Introduction

Universal Device Preview is an advanced and easy-to-use tool that gives you an accurate preview of your game for a huge variety of devices, and help you adapt your games to them. Instantly test how your content looks and feels on each of your target devices within the editor to prevent long and tedious iterations, and incredibly fasten your game development process!

Figure 1: Advanced in-editor preview.

1.2 Requirements

Preview and gallery:

- The solution works with Unity 2017.2 and later free or pro edition.

Safe Area requirements:

- For iOS: Unity 2017.2.1 or higher.
- For Android: Unity 2018.4.1 or higher, and the device must be running on API level 28 or higher (Android 9).

1.3 Known Issues

- On Unity 2018.1 and later, UI element with a stretch anchor within canvas using a constant physical size canvas scaler may not be positioned correctly in the preview.

2 QUICK START GUIDE

1. Open the gallery window in the menu *Window/AlmostEngine/Universal Device Preview/-Gallery*, or the preview window in the menu *Window/AlmostEngine/Universal Device Preview/Device Preview*.

2. Open the settings window in the menu *Window/AlmostEngine/Universal Device Preview/Settings*, or using the *Settings* button in the gallery or preview window.

3. Set your screen PPI (Pixel Per Inch) in the Gallery section. A wrong PPI would result in a bad device scaling in ppi mode.

4. Select the resolutions to be used for the preview, create your owns, or use the device preset selector to add one of the preset resolutions.

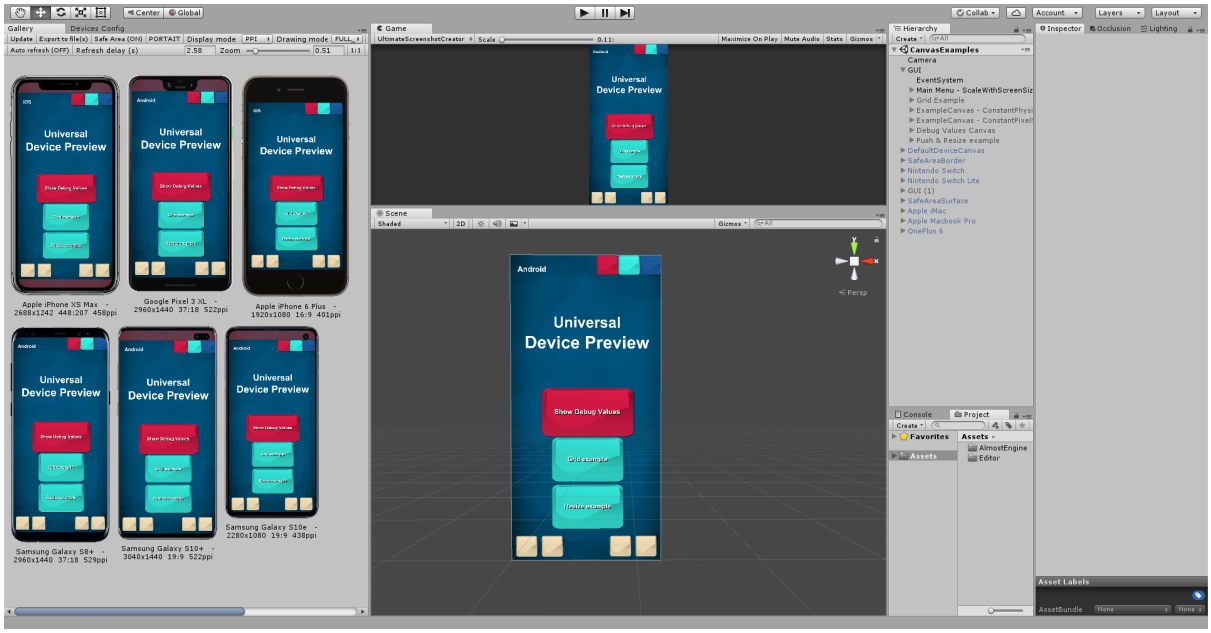
5. Set the destination folder and the filename for exporting the screenshots.

6. Select the display mode: $RATIO$, PPI, or $PIXELS$.

7. Select the drawing mode: $TEXTURE\_ONLY$, $SCREEN\_MASK$, or $FULL\_DEVICE$.

8. Click on Update in the window to create a preview of all selected resolutions. Note that you can also use and customize the menu item hotkeys (Section 5.6).

Important Android and iOS configuration. *Universal Device Preview* comes with all *Ultimate Screenshot Creator* features, including the ability to save to iOS and Android galleries. You need to configure it properly (see *Ultimate Screenshot Creator* documentation) or to exclude those features (see Section 5.7).



3 GALLERY WINDOW

The Gallery Window allows you to preview multiple devices or resolutions at a glance. The displayed devices are managed in the Settings Windows (see Section 5). Open the gallery window in *Window/AlmostEngine/Universal Device Preview/Gallery*.

Figure 2: The gallery window

Update. Press this button to update all active resolutions. Note that you can use the hotkey (F5) to update the resolution previews when focused on the gallery window or on the sceneview.

Export. Will export the preview images to files, using the export settings of the *PreviewManager*. Note that it will also export the device image is you are in *FULL_DEVICE* drawing mode. To export only the screen content, switch to the *TEXTURE_ONLY* mode.

Safe Area. Show or hide the safe area. Note that only device with valid safe area data will have the safe area rendered.

Landscape/Portrait. Switch between landscape or portrait for all active devices.

Display Mode. There are three display modes:

- *RATIO*: the preview width is proportional to the window width, and its height is computed based on the resolution ratio. A zoom of 1 means the preview width is equals to the window width.
- *PIXEL*: the preview dimensions are proportional to the resolution dimensions. A zoom of 1 means one pixel on screen is equal to one pixel on the device.
- *PPI*: the preview dimensions are proportional to the resolution physical dimensions. A zoom of 1 means the preview physical size on screen is equal to the device screen physical size in real life.

Drawing Mode. Select the drawing mode. See Section 5 for more details.

Settings. Open the settings window.

Auto Refresh. Auto refresh can be set to play mode and/or edit mode in the settings window.

Refresh Delay. You can set the time waited between each refresh pass. Set to 0 for a real-time preview, set to a higher value to reduce performance costs.

Zoom. Use the zoom slide to change the preview dimensions.
Note that you can also use Ctrl + Mouse scrollWheel.

1:1. Use that button to quickly reset the zoom value to 1.

Device Selection. You can select which of the active devices you want to preview in the window. To edit the active devices, open the settings window and use the preset selector to add or remove devices.

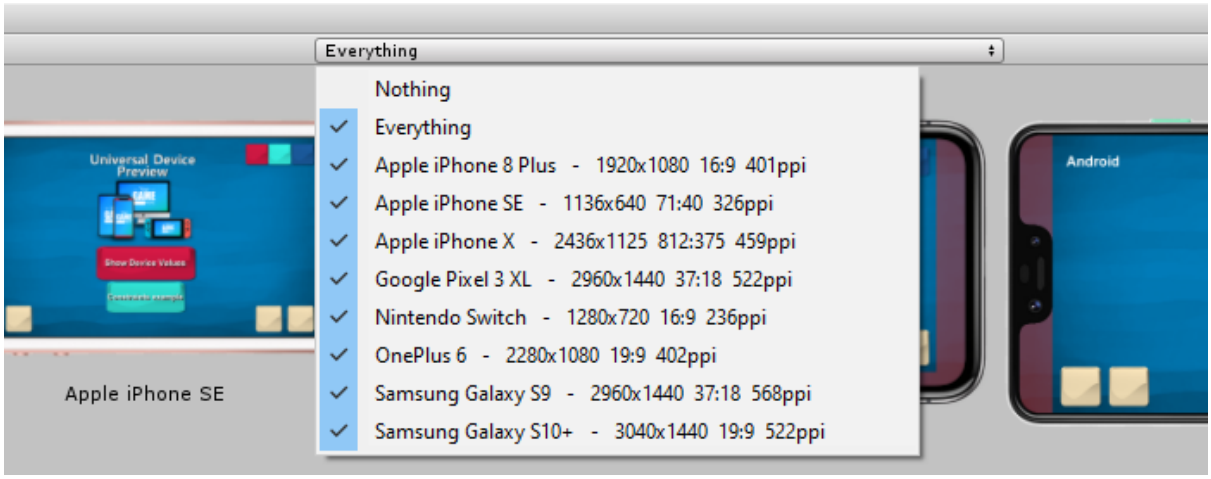


Figure 3: Devices selection.

4 DEVICE PREVIEW WINDOW

The Device Preview Window allows to preview a specific device, and do a live preview in play mode. Open the preview window in *Window/AlmostEngine/Universal Device Preview/Device Preview*.

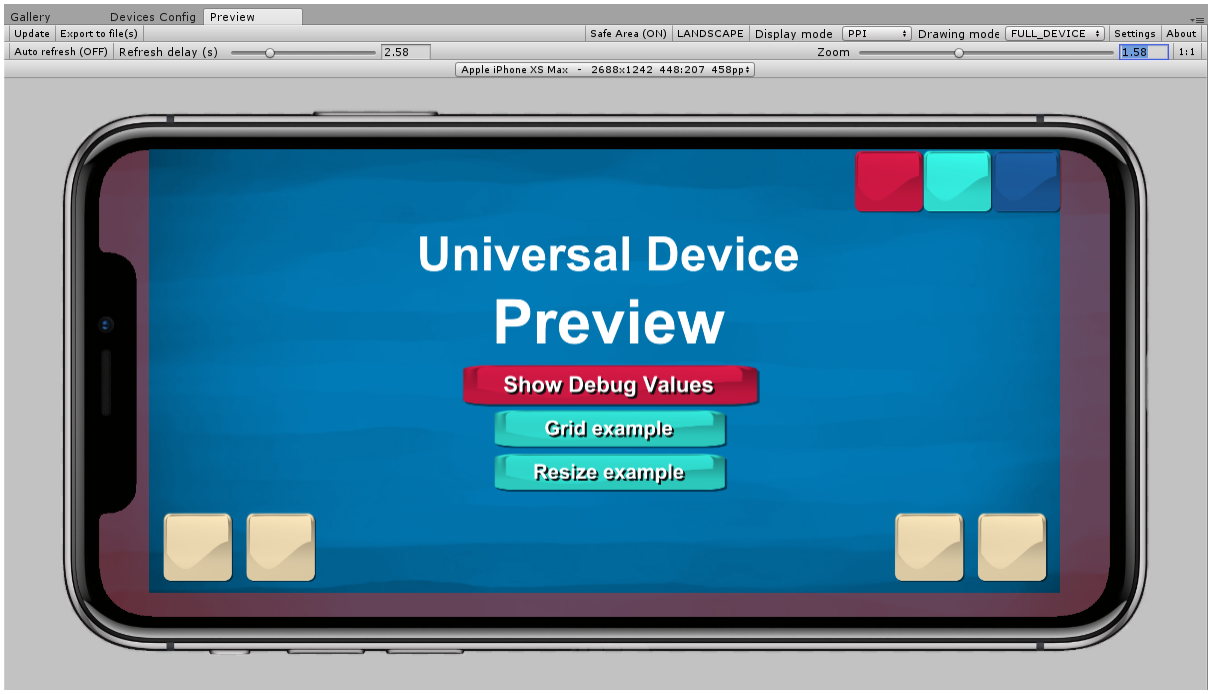


Figure 4: The preview window

Device Selection. Select which of the active devices you want to preview in the window.

Figure 5: Device selection.

Live Preview with Auto refresh. If you want a live preview of your game, enable auto refresh and start playing. You can stop/start the auto refresh at any time while playing.

Note that it is recommended to close the Gallery Window and only use the Preview Window with the desired resolution.It is also recommended to use the $TEXTURE\_ONLY$ mode. Indeed, with only one active resolution, the gameview and UI will be resized only one time, so the game will still be playable with the mouse. With several active resolutions, the gameview will be constantly resized, making it hard to play the game while previewing the devices. In that case, it is recommended to set a high delay value, or to manually refresh the preview using the hotkeys.

5 SETTINGS WINDOW

The settings window allows to configure the gallery and device preview window. Open the settings window in *Window/AlmostEngine/Universal Device Preview/Settings*. The Settings Window settings are saved in the file *MultiDevicePreviewConfig.asset*.

5.1 Preview Settings

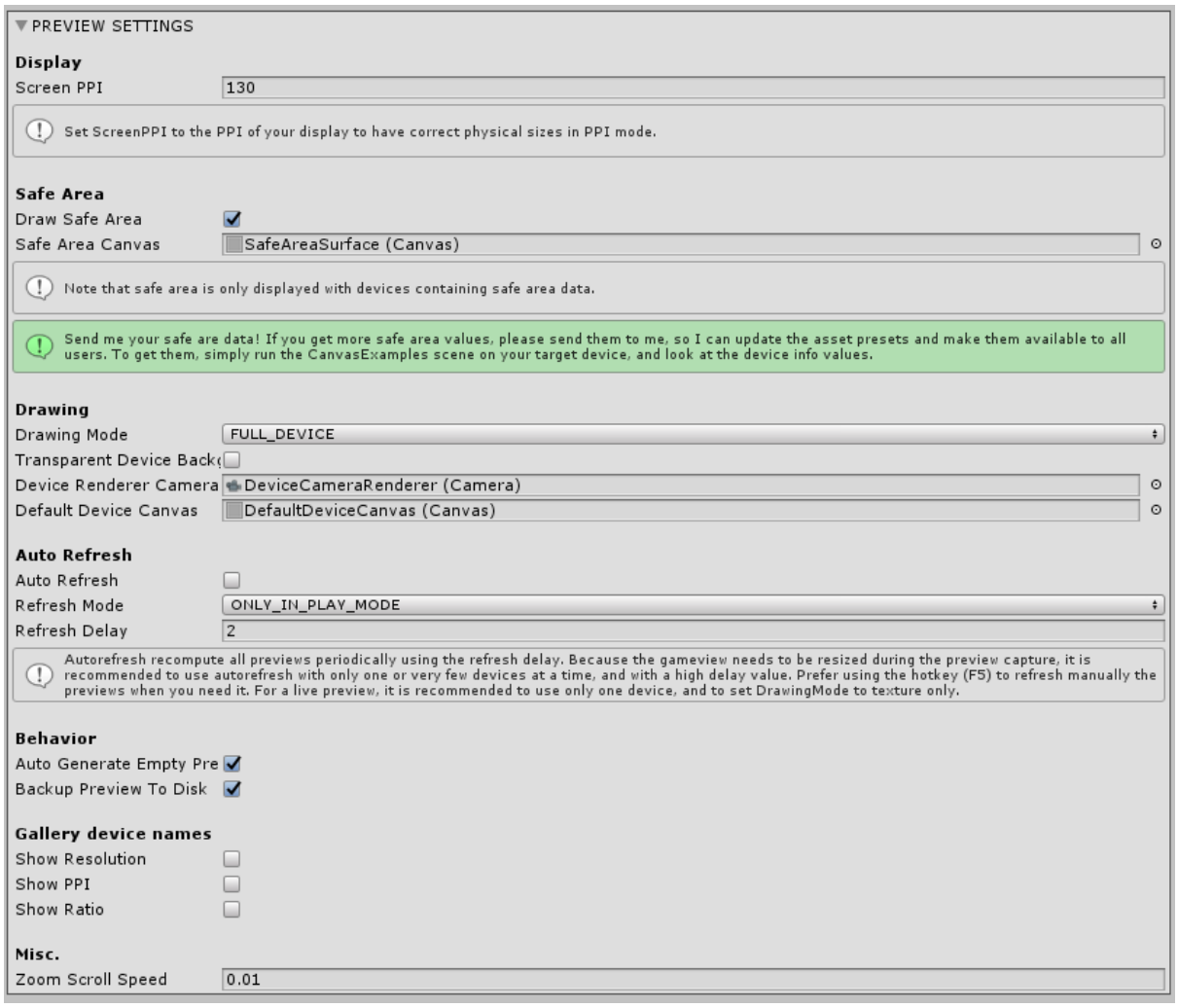


Figure 6: The settings window

Screen PPI. To use the gallery in PPI mode, you must correctly set your display screen ppi. In ppi mode, with a zoom of 1, the physical preview dimensions on your screen will physically match the device dimensions.

Auto Refresh. Can be enabled in play mode, in editor mode, or in both mode. It is recommended to enable auto refresh only with one active resolution and in $TEXTURE\_ONLY$ mode to prevent the gameview to be constantly resized, but it works with any number of resolutions.

Safe Area. Show or hide the safe area. You can customize the safe area renderer by setting a custom canvas.

Drawing Mode. You can chose between three device drawing mode.

- *TEXTURE_ONLY* draws the screen at the correct scale and resolution.
- *SCREEN_MASK* draws the device screen shape, for instance phone with notches will have the notches part cut.
- *FULL_DEVICE* draws the device using a device canvas, perfect to really see your game on the target device.

Note that devices with no custom device canvas will be rendered using a generic device contour.

Device Renderer Camera. The camera used to render the devices. Use the default DeviceCameraRender prefab.

Default Device Canvas. When a device do not have a custom canvas, the specified default canvas is used. Note that it can be customized, for instance to change the device color.

Transparent Device Background. By default, the background color of the full device preview matches the editor color. If you want to export the full device preview images to files, enable the transparent device background.

5.1.1 Behavior

Auto generate empty preview. Automatically generate a preview for all devices that have no preview yet. Disable if you want to call the preview yourself.

Backup preview to disk. Backup the generated preview images on disk not to lose them when changing play mode or exiting the editor. Note that creating a backup of the preview of a large quantity of devices can take some time, depending on the hardware used. Disable that feature if you experience performance issues when changing game modes.

5.2 Resolutions

Universal Device Preview comes with hundred of device and resolution presets. Use the preset selector to search and add a device to the active resolution list. Presets are sorted by categories and collections.

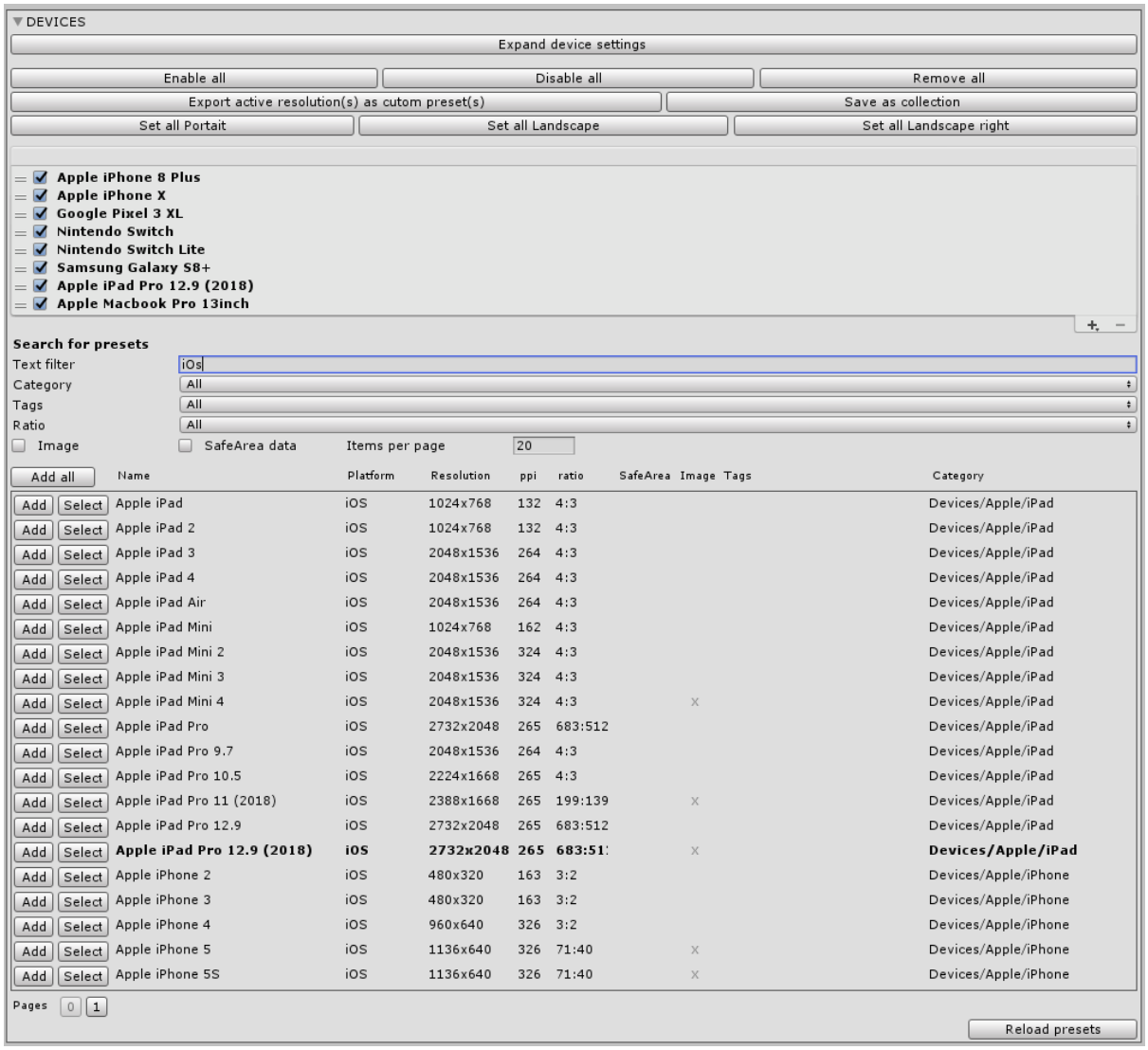


Figure 7: Top: The active resolutions list. Bottom: The device preset selector.

Expanded settings. By expanding the active resolution settings, you can see and customize the active resolution settings. Note that they are instances of resolution presets, modifying the resolution does not affect the preset, and preset changes are not applied automatically.

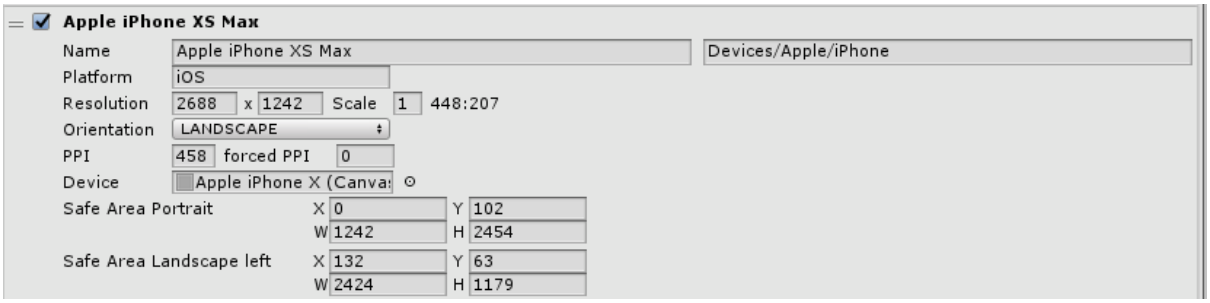


Figure 8: Expanded resolution settings.

Save as collection. You can create custom collections by clicking on "save as collection". A new Collection containing the active presets will be created. Note that if your list contains customized presets, i.e. if you changed one of the preset settings (resolution, scale, name, etc.), a new custom preset will also be created. The original preset will not be modified.

Tags. You can add tags to presets by clicking on "select" and editing the preset tags in the inspector. Note that tags are not saved on the preset files but in the DeviceTagDatabase.asset file.

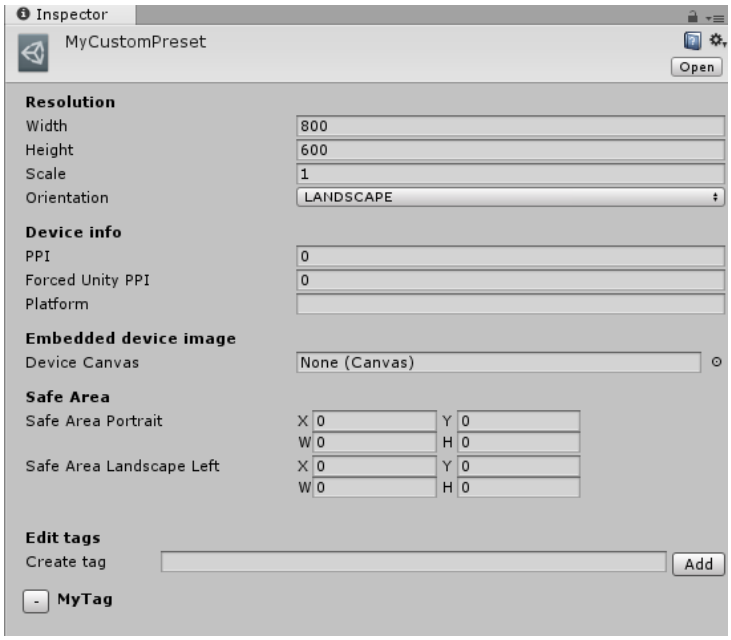


Figure 9: Edit tags in the preset inspector.

PPI. All the devices presets come with a *ppi* (pixel per inch) information, to be used to display the devices into the gallery window at their physical size.

Do not forget to set your screen *ppi* (in the Gallery section of the PreviewManager inspector) to the value corresponding to your display screen.

PPI "Forced" value. Sometimes, Canvas with Constant Physical Size scale will not be scaled in the Unity editor as in the device. This can occur because on the device Unity uses the Screen.dpi value, which may be wrong. Set the forced ppi value to the Screen.dpi returned by your device to get the same scaling for Constant Physical Size canvas scaler in the preview than in your device.

Note that there is a know issue on Unity 2018.1 and later: the Canvas with Constant Physical Size scaler may not be scaled properly.

Device. The device renderer Canvas to be used to perform the integration of the picture into a real device picture.

Note that with an empty device, the default device canvas will be used.

5.3 Managing Resolution Presets

Resolution presets are stored as asset files in the project folder. It is possible to easily create your own presets.

Creating Custom Presets. Click on "Save active resolution(s) as custom preset(s)" to automatically create new preset files using the resolutions in your custom resolution list.

Also, you can right click on your project folder and select *Asset/Almost Engine/Ultimate Screenshot Creator/Custom preset*.

Creating Custom Collections. Collections are set of presets that can be grouped and loaded simultaneously.

To create a new collection, you can export your active resolution list to a new collection by clicking on "Save as collection".

Also, you can right click on your project folder and select *Asset/Almost Engine/Ultimate Screenshot Creator/Custom collection*. Then, add all desired resolution preset to the list.

Creating Custom Popularity Collections. Collections are set of presets that can be grouped and loaded simultaneously.

To create a new popularity collection, right click on your project folder and select *Asset/Almost Engine/Ultimate Screenshot Creator/Popularity preset*. Then, add all desired resolution preset to the list, and specify their associated popularity values.

Copying Presets and Settings to an New Project. If you have some custom presets you want to copy to a new project, simply copy and paste your preset .asset and .meta files.

To copy your Screenshot Manager and Device settings, copy ScreenshotWindowConfig.asset and MultiDevicePreviewConfig.asset to your new project.

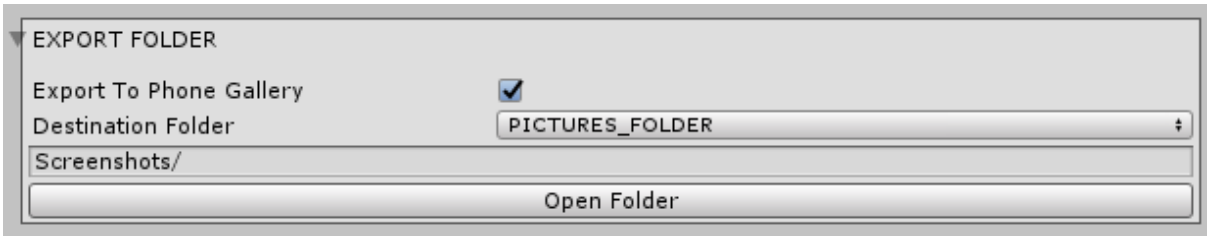


Figure 10: Destination settings.

5.4 Destination

There are three export modes:

- *CUSTOM_FOLDER* allows you to select the export folder.
- *DATA_PATH* exports the screenshots relatively to the project data path.
- *PERSISTENT_DATA_PATH* exports the screenshots relatively to the persistent data path.
- *PICTURE_FOLDER* exports the screenshots relatively to the platform picture folder. It is the recommended mode for taking screenshot in-game on all platforms.

Export to Phone Gallery. Set if the screenshot should be automatically added to the phone gallery on iOS and Android.

5.5 File Name

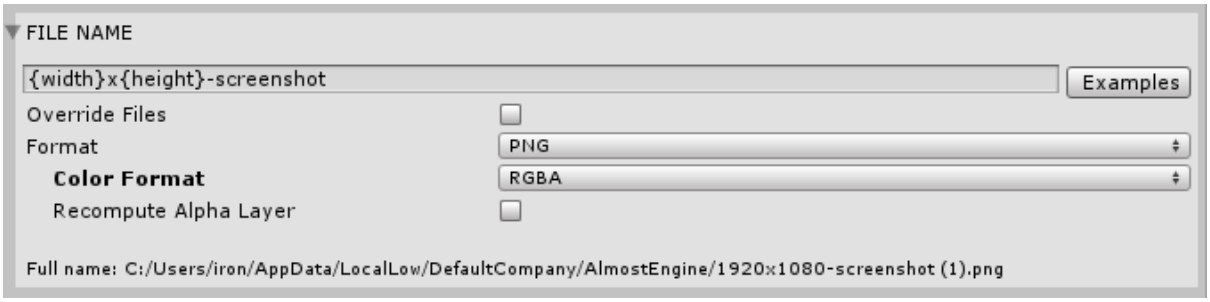


Figure 11: File name settings.

FileName. Defines the file name to be used for the screenshots. Use the *Examples* button to select one of the name presets.

You can use and combine the following symbols to better match your needs:

- {year}, {month}, {day}, {hour}, {minute}, {second} for the current time information,
- {width}, {height}, {scale}, {ratio}, {orientation}, {ppi}, {percent}, {name}, {category} for the resolution information.
- {layer} for the camera name in separated layer mode.
- {composer} for the current composer name in composition mode.
- {batch} for the current batch process name in batches mode.

Note that you can use the file name to group screenshots by resolution folders, etc. For instance {*width*}-{*height*}/*screenshot.png* will create one folder by resolution and create one *screenshot.png* in each of them.

Overwrite Existing Files. By default, if you try to create a screenshot file that already exists, its name will be incremented. For instance: screenshot.png, screenshot(1).png, ... Set *overwrite* to *true* if you want to overwrite the existing files.

Format. You can export to *PNG* or to *JPG* with a custom quality.

Color Format. In *PNG*, you can export to *RGB* or to *RGBA*. Use *RGBA* to create screenshots with an alpha layer, enabling transparent backgrounds.

Recompute Alpha Layer. Force alpha layer to be recomputed. This is a costly process. Use only if you have alpha problems in *RGBA* mode.

Full Name Preview. You can look at the full name preview to check if everything is correct. Note that the resolution information used for the full name preview is the one of the first resolution in the list.

5.6 Hotkeys

To update or export screenshots, you can use the menu items hotkeys *Tools/Device Preview/Update Preview(s)* and *Tools/Device Preview/Export Preview(s)*.

The hotkeys can be edited by editing the scripts *UpdateDevicePreviewMenuItem.cs* and *ExportDevicePreviewMenuItem.cs*. More details can be found on the script files. If you change the hotkeys, remember not to overwrite the menu item scripts in the next asset updates.

5.7 Permissions

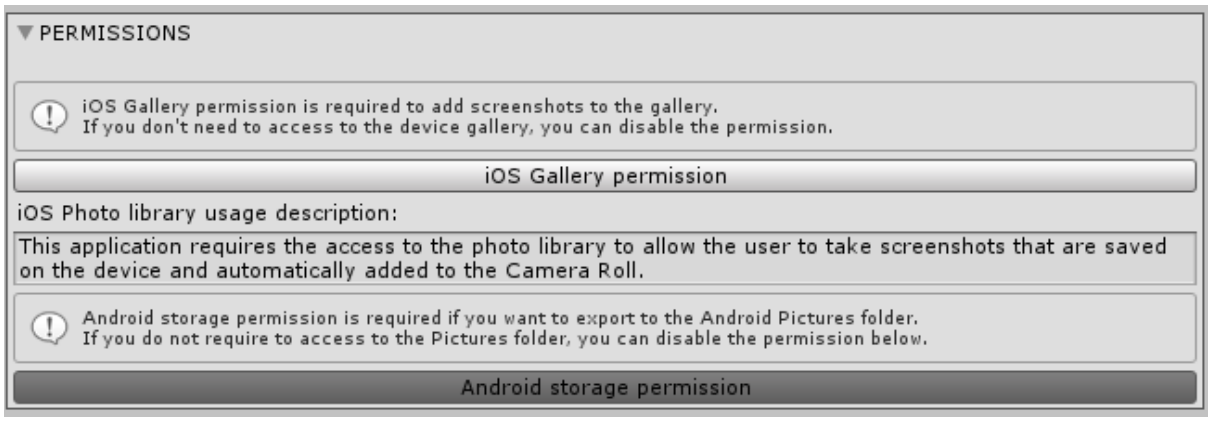


Figure 12: Permission settings.

iOS Gallery Permission To display your screenshot on the iOS Gallery, you need to request the Gallery permission. Toggle the iOS Gallery permission to do it automatically. You will be able to customize the usage description that is adde to your iOS app.

Android Storage Permission To export to the Pictures folder, you need to request the storage permission. Toggle the Android Storage Permission button to automatically do it.

5.8 Exclude Features from Build

You can choose to exclude some features from the build depending on your needs. If you want only an in-editor use, you can exclude everything and nothing will be added to your project. You can also exclude the Gallery and Share features if you do not use them to prevent the need for any iOS permission request.

Figure 13: Exclude form build settings.

6 SAFE AREA

6.1 Safe area preview

Enable the safe area preview to better manage phones with notches. Safe area is only displayed with devices having safe area simulation values.

Requirements.

- For iOS: Unity 2017.2.1 or higher.
- For Android: Unity 2018.4.1 or higher, with device running on API level 28 or higher (Android 9).

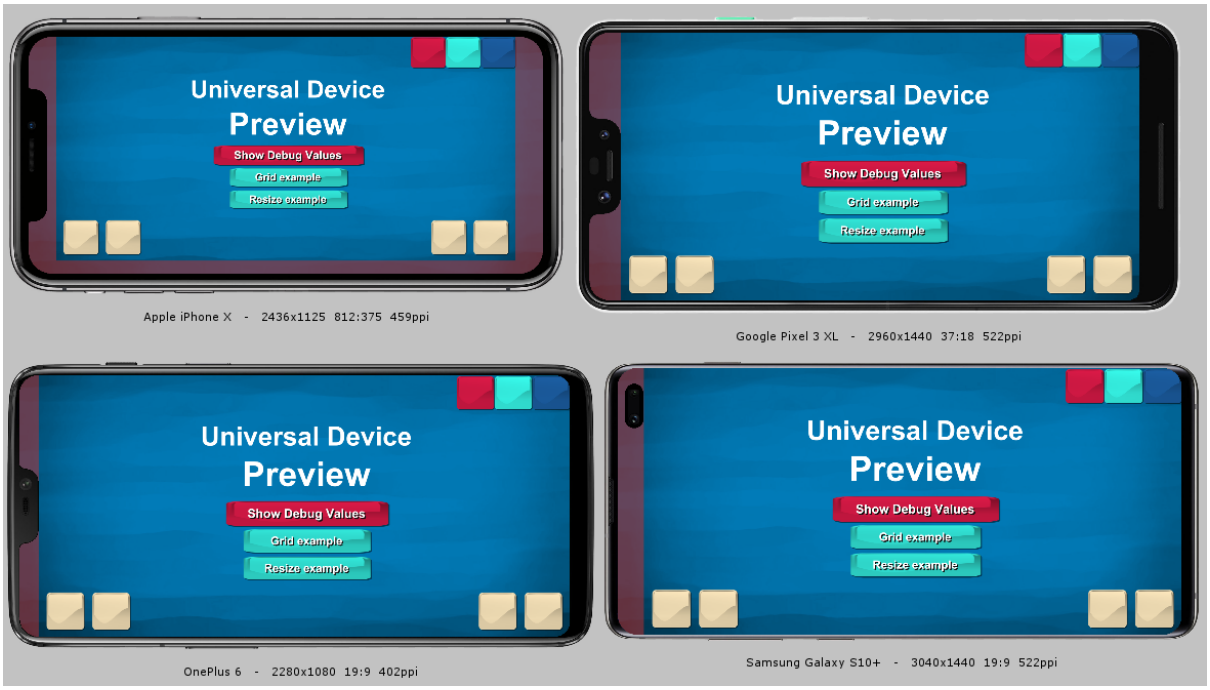


Figure 14: Safe area preview

6.2 Getting new safe area values

Getting safe area values for the simulation is not an easy process, since the data must be acquired manually for each device. The asset comes with some safe area values, but not all devices with notches have safe area values.

To get new safe area values, build the project and run the CanvasExample scene on your target devices. Use the "show debug values" to get the safe area values.

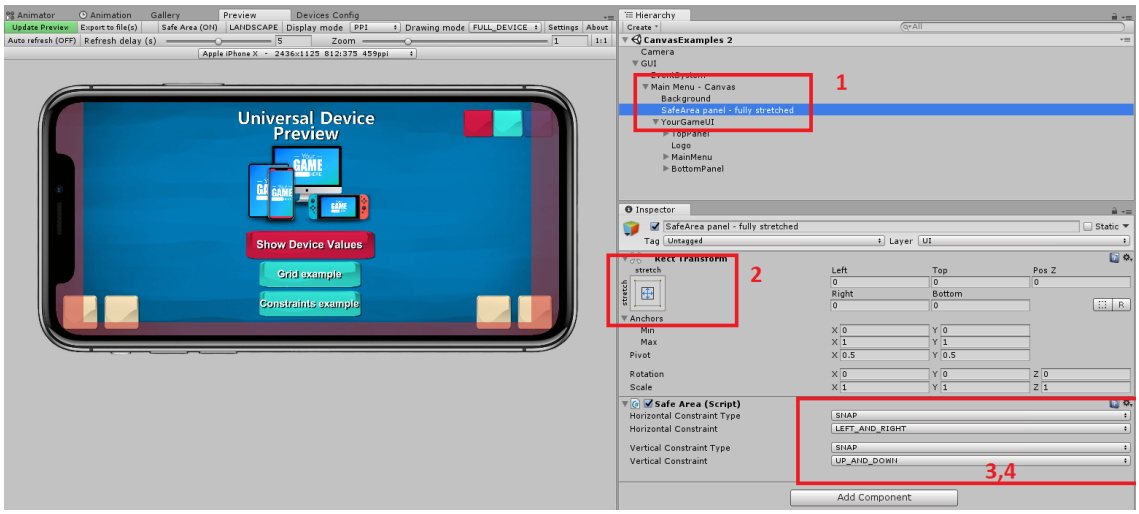
If you get more safe area values, please send them to me, so I can update the asset presets and make them available to all users.

6.3 Automatic layout adaptation

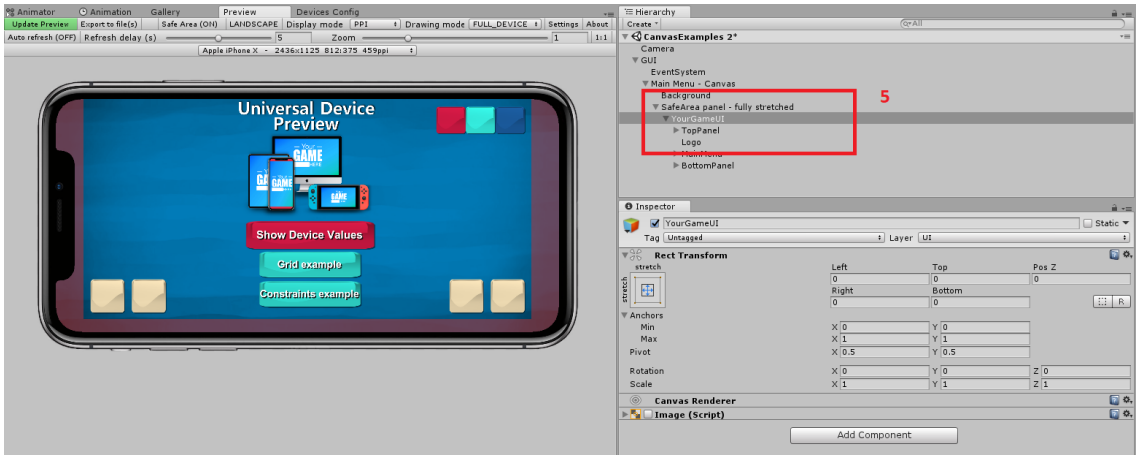
By adding a SafeArea component to a UI layout, you can automatically adapt your UI to the safe area. The most common use of the SafeArea component is to have one unique safe area layout and put all your game UI inside.

To work properly, **the safe area layout must be a child of a Canvas**, or a child of a fully stretched panel in the orientations of the constraints (horizontal and/or vertical).

1. Create an empty panel, as a direct child of your Canvas. It is recommended that the Canvas has a CanvasScaler component "Scale with screen size".
2. Check that the safe area panel anchor is fully stretched.
3. Add a SafeArea component to that panel.
4. Select Horizontal and Vertical SNAP and LEFT AND RIGHT and TOP AND BOTTOM.



5. Move all your game UI inside that panel. Your game UI is now adapted to the safe area constraints. Note that it is best to keep your background out of the safe area constraints so it still covers all the screen.



6.4 Automatic layout adaptation - Advanced constraints

For more specific adaptation cases, the asset comes with more customizable safe area constraints.

You can independently set horizontal and/or vertical constraints, and set the constraint to snap, push or expand. Play with the different constraints, explore the example scene to understand how to adapt your content.

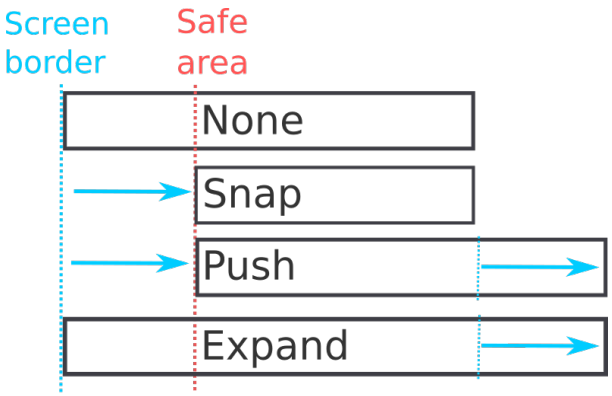


Figure 15: SafeArea constraint types.

6.4.1 Snap.

The component borders are snapped to the safe area border. The content size is consequently reduced.

The most common use of the SafeArea component is to add a full snap constraints (horizontal and vertically to top and bottom, right and left) to a fully stretched layout, and put all the game UI inside. This way, you can ensure that everything that is put inside will always be out of the safe area.

6.4.2 Push.

The component is pushed away from the safe area, keeping its size.

Note that pushing from left AND right, or top AND bottom, will often do nothing, as the constraints are cancelling each others.

6.4.3 Expand.

The component is expanded to compensate the safe area border. Its content is not adapted, it is up to you to make sure it is. For instance, the child of the expanded panel should have an anchor to the expanded side so it moves with the panel border (see example scene).

Note that expanding from left AND right, or top AND bottom, will often do nothing, as the constraints are cancelling each others.

Figure 16: Example of different constraint types.

6.4.4 Check your anchors

Note that the **anchor** of the panel that you are trying to adapt is of **primary importance**. For instance, to push from the left, it is recommended that the panel has an anchor point to the top-left, left, or bottom-left. You should first ensure that your layouts are working properly when deforming the gameview before trying to add a SafeArea component.

Depending on the desired constraints, the following anchor are recommended:

- LEFT: the anchor should be to the top-left, left, or bottom-left.
- RIGHT: the anchor should be to the top-right, right, or bottom-right.
- LEFT AND RIGHT: the anchor should be fully stretched horizontally.
- TOP: the anchor should be to the top-right, top, or top-left.
- BOTTOM: the anchor should be to the bottom-right, bottom, or bottom-left.
- TOP AND BOTTOM: the anchor should be fully stretched vertically.

6.5 Custom device borders

You will quickly notice that, depending on the manufactured, the safe area have different shapes. For instance, on iPhone, the safe area borders are very large. On the contrary, on most Android phones, the safe area are minimal, often only corresponding to the camera.

This is because they decided to handle the SafeArea differently. During your UI creation, you may be wanting to change the SafeArea constraints to better match your needs, for instance to have similar safe area on all devices.

The DeviceCustomBorder component has been created to add arbitrary border constraints to a specific list of devices. It can be used in combination or independently of the SafeArea component.

Note that the SafeArea visualization is NOT affected by the DeviceCustomBorder component.

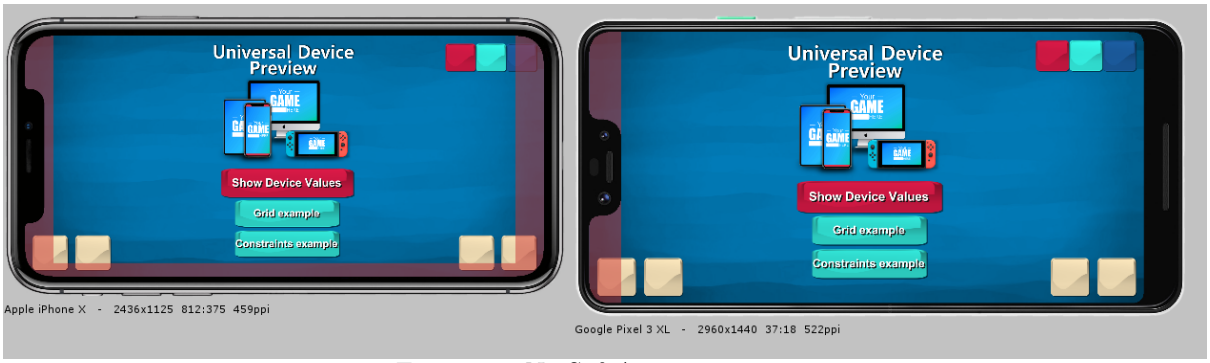


Figure 17: No SafeArea constraints.

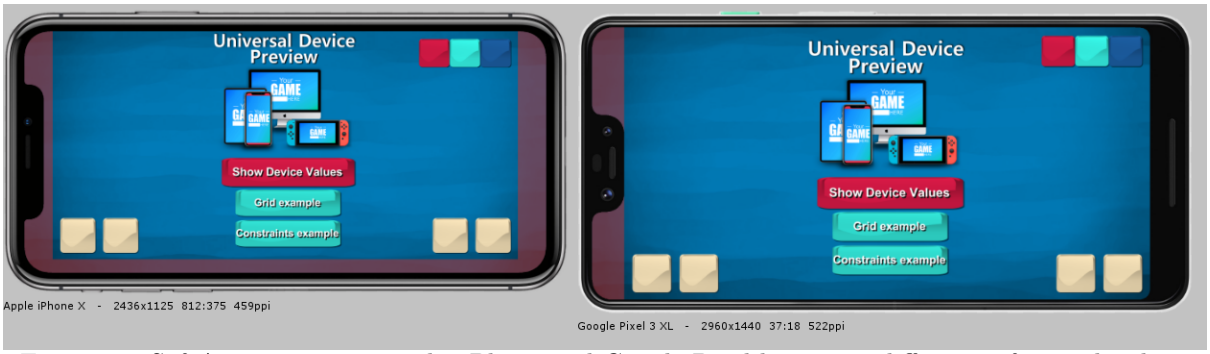


Figure 18: SafeArea constraints. The iPhone and Google Pixel have very different safe area borders.

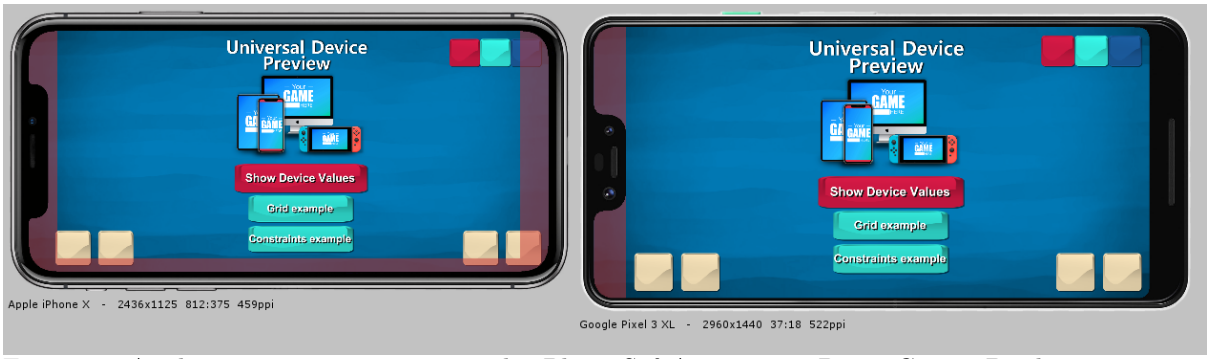


Figure 19: Applying custom constraints to the iPhone SafeArea using a DeviceCustomBorder component. Now the iPhone behaves almost like the Google Pixel. Note that the SafeArea visualization is NOT affected by the DeviceCustomBorder component.

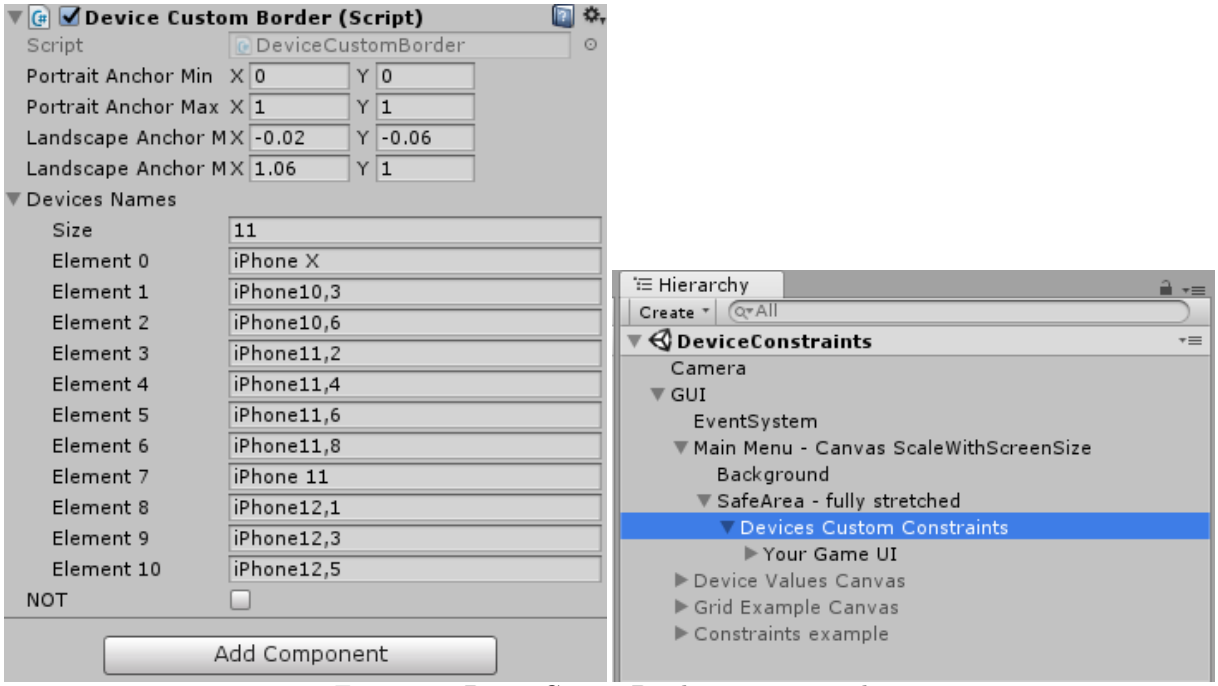


Figure 20: DeviceCustomBorder settings used.

7 API

7.1 DeviceInfo

In editor, the DeviceInfo class provides information on the simulated device, like the name, resolution, ppi, platform. On build, the DeviceInfo will return the information of the device.

7.2 Events

7.2.1 Preview events

If you want to catch simulation events, you can register to the ScreenshotTaker delegates:

```
public static void onResolutionUpdateStartDelegate (ScreenshotResolution res)
```

Is called just before capturing the given resolution.

```
public static void onResolutionScreenResizedDelegate (ScreenshotResolution res)
```

Is called just after resizing the gameview in *GAMEVIEW_RESIZING* mode.

```
public static void onResolutionUpdateEndDelegate (ScreenshotResolution res)
```

Is called just after capturing the given resolution.

7.2.2 Device events

If you want to catch the change in orientation event, add a DetectOrientationChange component to your scene and listen to:

```
public static UnityAction<bool> onOrientationChangedDelegate = (bool isPortrait)
```

8 FAQ

8.1 Common issues

8.1.1 I have a compilation error on iOS

The **Photos** framework dependency, required by the *Ultimate Screenshot Creator* plugin, may not have been added automatically. If you do not want the feature to take screenshots in build with the *Ultimate Screenshot Creator*, please exclude that feature from build in the settings windows. If you want to use it, please refer to *Ultimate Screenshot Creator* documentation for more details.

8.1.2 The PPI mode preview is different from what I see on my device

1. Check that your $Screen\_PPI$ value is correct.
2. Check that the device PPI value is correct. If you find a device preset with an incorrect value, please contact me.
3. Sometimes, Unity does not scale the device UI with the good PPI value. When your app is running on the target device, check that the *Screen.dpi* value returned by Unity matches the real screen PPI value. You can use the *ExampleCanvas* demo on your device to display the *Screen.dpi* value used by Unity to scale the UI. If it is different from the real device PPI, set the *Forced* PPI value to match the value returned by Unity.
4. Still not working? It's time to contact me.

8.1.3 My GameView is blinking when I update the device or gallery previews.

This is expected. To render the previews, the plugin needs to render the game by deforming or resizing the GameView for a few milliseconds. More details on Section 8.1.4.

8.1.4 My GameView and layout are doing strange things when I update the device or gallery previews.

Having the GameView deformed or blinking is expected when you use the $GAMEVIEW\_RESIZING$ capture mode.

For Unity 4.6 to 5.3, the algorithm rescales the GameView window to match the screenshot dimensions, and this undocks the GameView window. To prevent it, the editor layout is saved before the capture and restored after it, creating a sort of "flashing" effect. If this annoys you, you can set *Force Layout Preservation* to false.

With Unity 5.4 and later, the GameView has an inner "scale" which allows the modification of its dimensions without modifying the editor layout. During the capture, its resolution is changed several times before being restored, creating a sort of "blinking".

8.1.5 Nothing Happens when I try to Update the Preview(s)

Sometimes, the renderer may be locked in Editor and refuse to update the preview. This happens rarely, but if you do not have any response from the *ScreenshotManager*, stop and play the game.

Do not hesitate to contact me if you can reproduce this bug, so I can correct it.

9 SUPPORT

Please do not hesitate to contact me if you have a feature request, or any question, issue or suggestion : support@wildmagegames.com.