

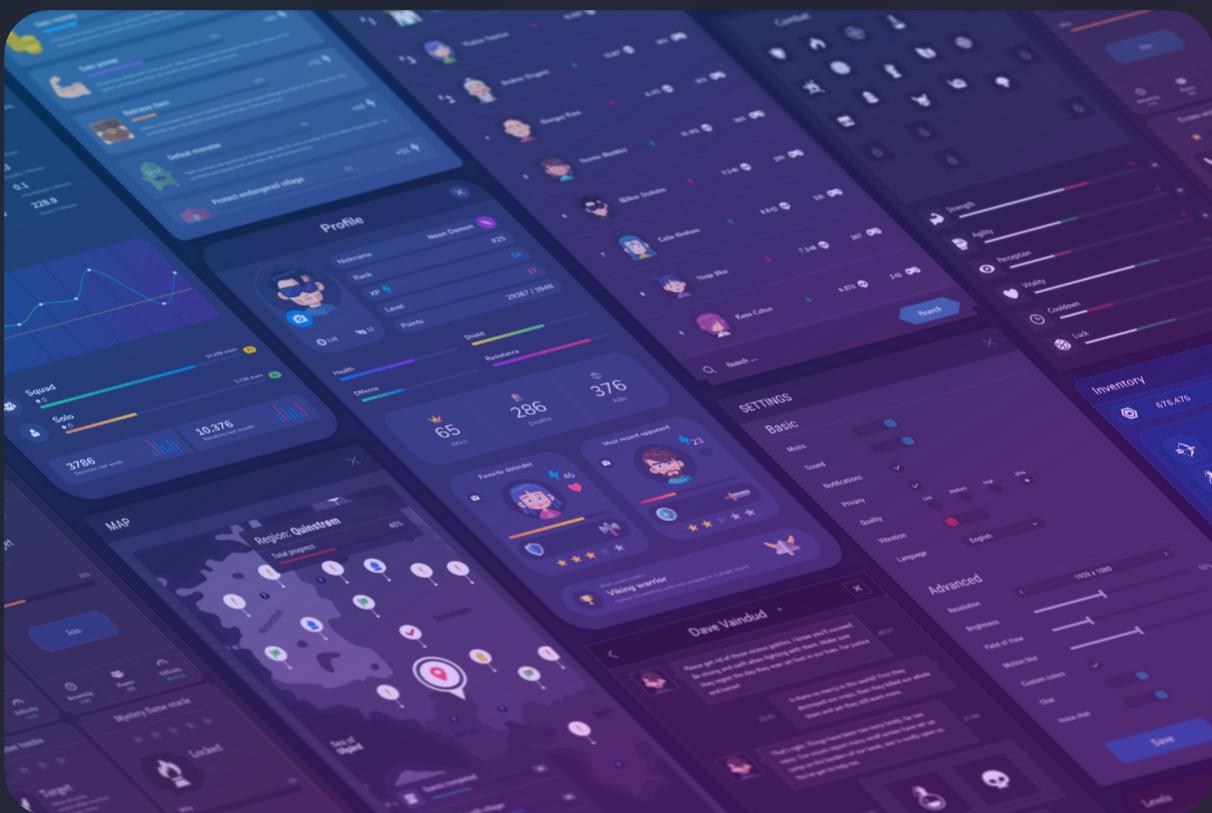
A big **thank you** for purchasing our



Ultimate Clean GUI Pack

Multipurpose Game UI

© gamevanilla



We hope you find this pack useful to create a great game!

If you have any support questions, please contact us [here](#).

Please make sure to include your invoice number.

The Ultimate Clean GUI Pack can only be used under the Unity Asset Store License Agreement which you can find [here](#).



Table of contents

1. What is Ultimate Clean GUI Pack?

1.1 Unity Version

2. What is included?

3. Asset structure and organization

4. Scripts

4.1. Demo

4.2. Reference resolution

4.3. Components

4.3.1. CleanButton

4.3.2. ButtonWithSound

4.3.3. SceneTransition

4.3.4. Popup / PopupOpener

4.3.4. ToggleWithLabel

4.3.6. Switch

4.3.7. SelectionSlider

4.3.8. ProgressBarAmountText / SliderAmountText

4.3.9. Notification / NotificationLauncher / Queue

4.3.10. Tooltip

4.3.11. URLOpener

5. Contact

6. Copyright & Terms of use

1. What is Ultimate Clean GUI Pack?

The **Clean & Minimalist GUI Pack** is a clean and modern multipurpose and customizable Game UI pack, containing a collection of UI elements and demos that can be used to create complete game user interfaces with a nice clean look.

1.1 Unity Version

While the sprites themselves do not depend on any specific Unity version, the accompanying demo project requires **Unity 2019.4.0 LTS** or higher.

2. What is included?

This game UI pack contains a complete demo project with **full C# source code** that you can use as a starting point for your own game.

Included are:

- Buttons
- Dropdown
- Scrollbars
- Input Fields
- Modal Windows
- Notification system
- Progress Bars
- Slider
- Selection slider
- Switches
- Toggles
- Tooltip system
- Pre-built demo scenes
- Pre-built popup prefabs
- White and colored icons
- C# scripts
- UI components
- Gradient component
- Sound effects
- Animations
- Color Palettes
- .PSD and .SVG Source icons
- .PSD Source Mockups

3. Asset structure and organization

After importing the asset package into your Unity project, you will see all the resources provided live in the [UltimateCleanGUIPack](#) folder.

This folder is further subdivided into the following folders:

Common

Contains all the assets used in the pack's demo. You can see this folder is further subdivided into folders according to the category of the corresponding asset (materials, prefabs, scenes, scripts, etc.).

- [Common/Animations](#)

Contains all Animations.

- [Common/Editor](#)

Contains two color palettes. The **Basic Color Palette** we used for the project and the **Complete Color Palette** is for choosing any advanced colors (very useful).

- [Common/Materials](#)

Contains the confetti (End game - Win popup prefab) and the Overlay material. The Overlay material is used 3d buttons of the Modern theme.

- [Common/Prefabs](#)

Contains the prefabs that are used across all themes.

- [Common/Scenes](#)

Contains scenes used for the demo project showcase. They act as a general launcher over the more specific scenes located in the Themes and UI Elements subfolders.

- [/Sprites/Demo](#)

Contains the **Main Demo Menu**. As a starting point, we recommend opening up the **Demo – Landscape** or the **Demo – Portrait** scene located in this folder.

- [/Sprites/Demo](#)

Contains the **UI Element** scenes.

- [Common/Scripts](#)

Contains the complete C# source code of this pack.

- **Common/Shaders**
Contains the Overlay shader (used in the Modern/Dark theme).
- **Common/Sounds**
Contains all sounds used in this pack.
- **Common/Sources**
Contains the original source files of the icons in .PNG, .PSD and .SVG formats.
- **Common/Sprites**
Contains all sprites used in this pack.
 - **/Sprites/Demo**
Contains all demo sprites.
 - **/Sprites/Icons**
Contains all icons, sorted by category.
 - **/Sprites/Icons Colored**
Contains all colored icons, sorted by category.
 - **/SpritesShapes**
Contains all shapes used for backgrounds, buttons etc., sorted by category.

Themes

Contains all the theme-specific items.

For each theme, you can find five subfolders:

- **Themes/Editor**
Contains the color palette for each theme.
- **Themes/Fonts**
Contains the OPL font for each theme. The pack uses:
 - **Classic theme:** Kreon (Light)
 - **Dark theme:** Vegur (Regular)
 - **Fantasy theme:** Kurale (Regular)
 - **Minimalist theme:** Roboto Condensed (Regular and Bold)
 - **Modern theme:** Roboto (Bold, Medium, Regular)
 - **Round themes:** Dosis (Bold, Medium, Semi Bold)
 - **SciFi:** Electrolize (Regular)

- **Themes/Prefabs**

Contains the prefabs for each theme.

- **/Prefabs/Popup Demos**

Contains the demo-specific popup prefabs.

- **/Prefabs/UI Elements**

Contains the fundamental, reusable UI element prefabs that you can use in your own projects and outside the context of the demos we provide.

- **Themes/Scenes**

Contains the scenes for each theme.

- **Themes/Sources**

Contains mockup source files as .PSD format for each theme.

As of update 2.0 and based on your feedback, we have tried to organize the asset in a way that is as convenient as possible for you to navigate through the vast amount of content included.

This means adding independent prefabs for the fundamental UI elements of the pack that you can use without having to extract them yourself from the demo scenes, and making their names and structure as intuitive and convenient as possible.

Please reach us via our official support channel if you have any questions or feedback about this and we will be happy to assist.

Documentation

Contains the documentation that you are reading right now.

4. Scripts

4.1. Demo

The pack contains a complete demo project with full C# source code that you can use as a starting point for your own game. While the source code is not intended to be a universal framework, it can be a very useful reference when it comes to learning how to approach the implementation of a game UI using Unity's built-in UI system. All the scenes in the demo project make use of Unity's Canvas to display their contents. The Canvas render mode is set to Screen Space – Camera and the canvas scaler is set to Scale With Screen Size UI scale mode.

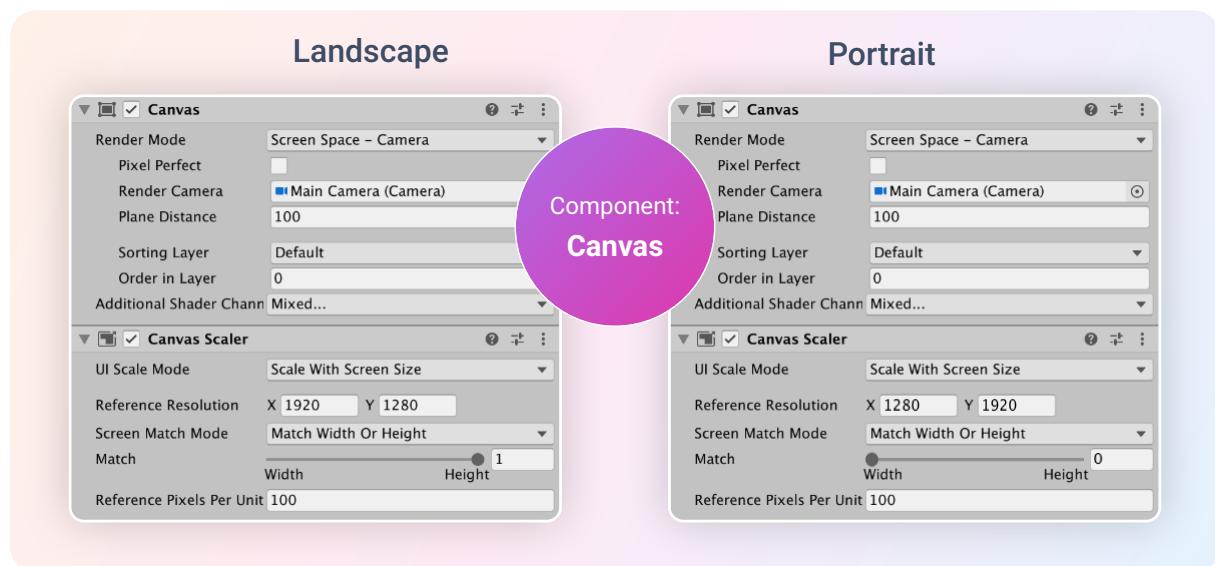
This, together with extensive use of anchors when positioning UI elements, makes it possible to automatically scale the UI across multiple resolutions.

You can find more details about how this works in the official Unity documentation [here](#).

4.2. Reference resolution

The demo is using a reference resolution of **1920x1280** (in landscape orientation) or **1280x1920** (in portrait orientation), which works well across a wide range of aspect ratios. This is particularly useful for mobile development, where screen sizes vary wildly between devices.

The Canvas render mode is set to **Screen Space – Camera** and the canvas scaler is set to **Scale With Screen Size** Canvas scale mode.

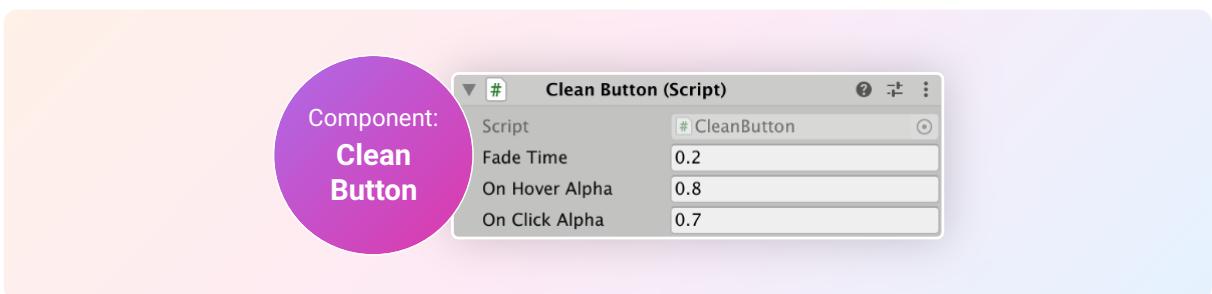


4.3. Components

The pack contains many useful scripts that extend Unity's UI system with additional features. In this section, we discuss in more detail what these components are and how they work.

4.3.1. CleanButton

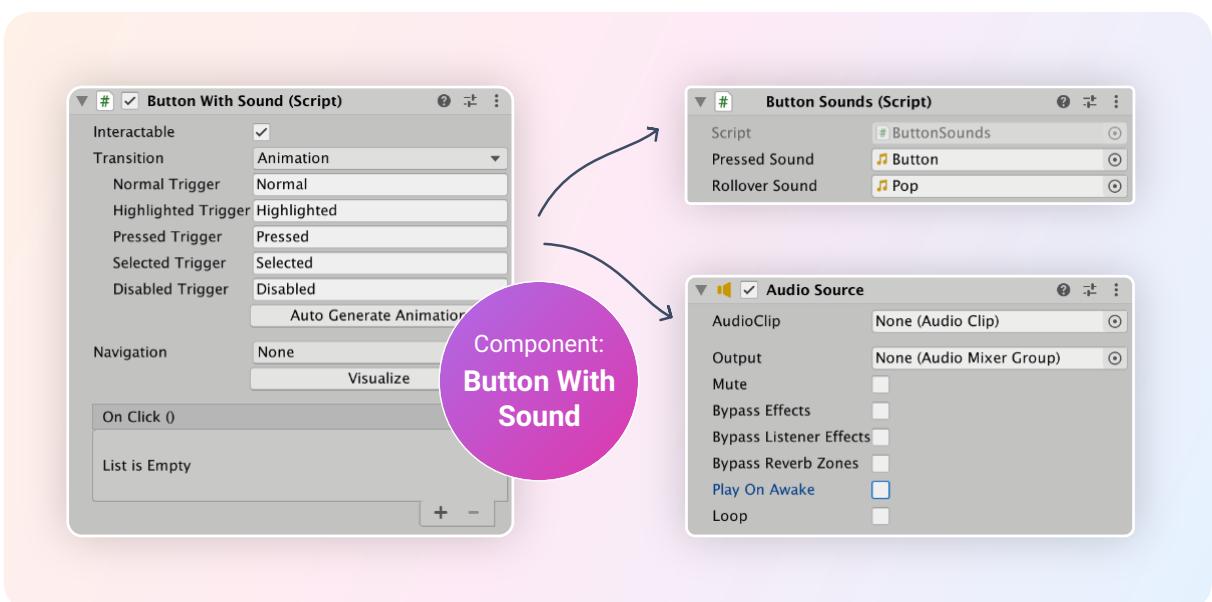
This component is used extensively in the pack and represents a button with the added ability to fade in/out as the player rolls over or presses it.



4.3.2. ButtonWithSound

This component is used extensively in the pack and represents a button with the added ability to play a sound as the player rolls over or presses it.

Additionally you need to add the [ButtonSounds component](#) to choose the sounds you like and the [Audio Source Component](#).

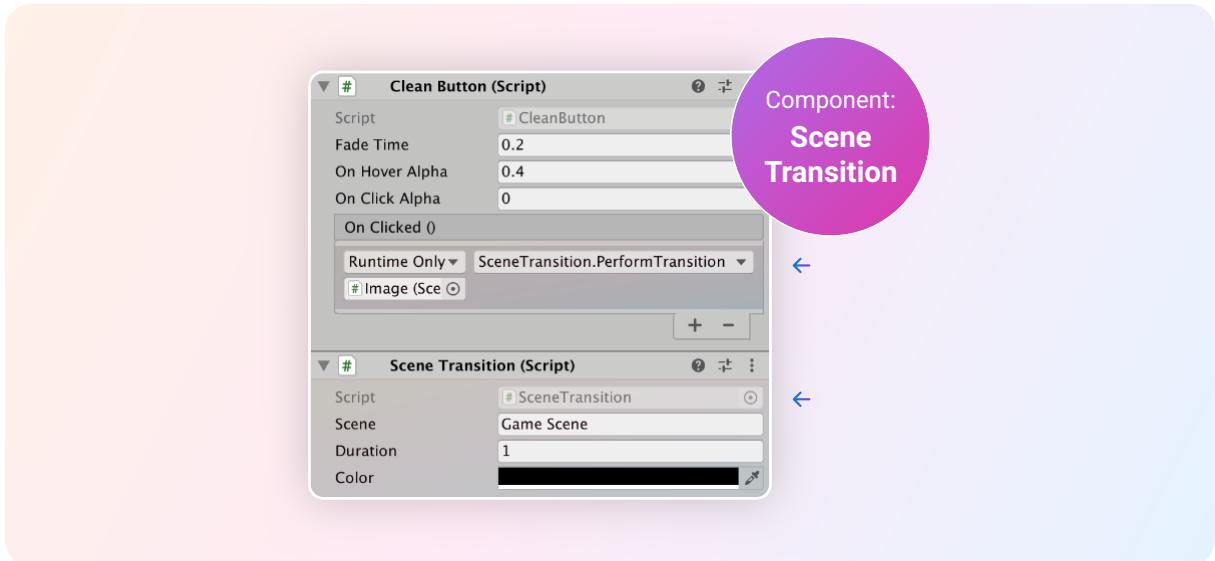


4.3.3. SceneTransition

This component allows you to transition from the current scene to a new one.

You can specify the following settings in this component:

- Scene: The name of the scene to transition to.
- Duration: The duration of the transition (in seconds).
- Color: The color of the fullscreen fade.

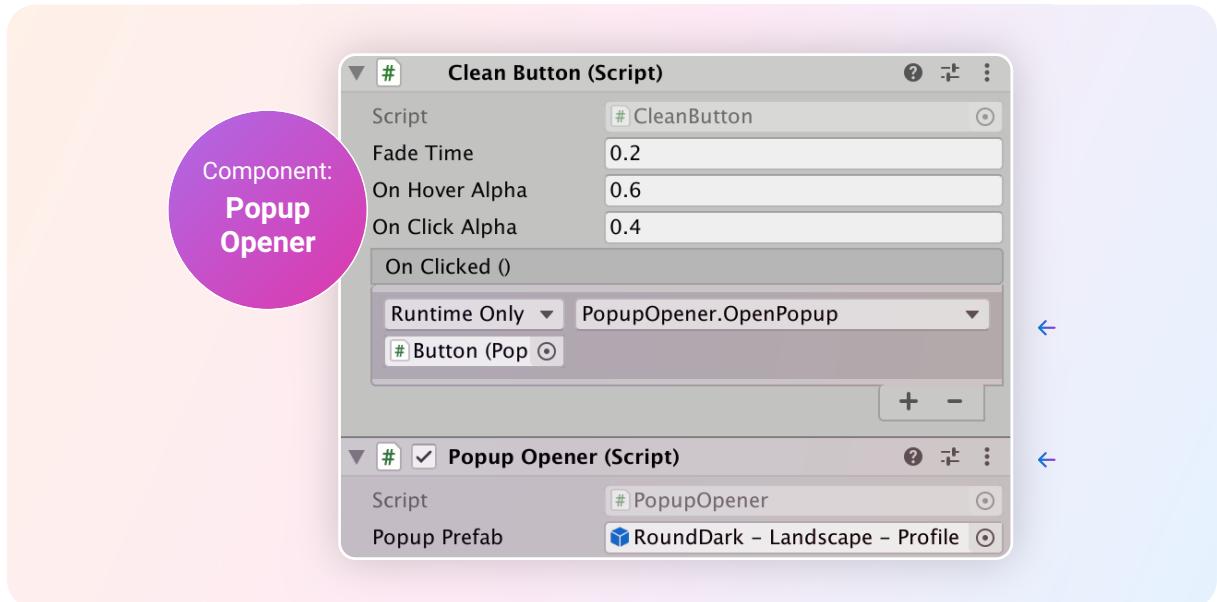


In order for the transition to be performed, you need to call the component's `PerformTransition` method.

We usually do this from a button's `OnClicked` action, as you can see in the example screenshot.

4.3.4. Popup/PopupOpener

This component allows you to open a popup prefab. This prefab is required to have an associated Popup component.



You specify the popup prefab in the Popup Prefab field. In order for the popup to be opened, you need to call the component's `OpenPopup` method.

We usually do this from a button's OnClick action, as you can see in the example screenshot.

4.3.5. ToggleWithLabel

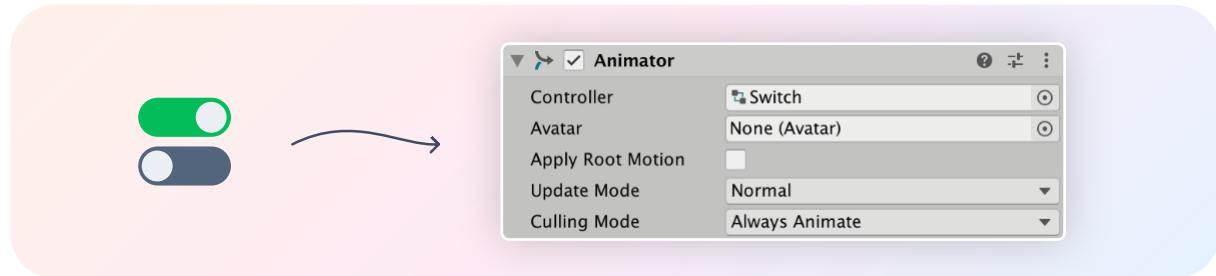
This component extends Unity's built-in toggle with an extra label that automatically changes as the value of the toggle changes.



You can find them inside each Checkbox folder.

4.3.6. Switch

This component works just like Unity's built-in slider, with the difference that it provides a smooth animation when changing between its enabled/disabled states.

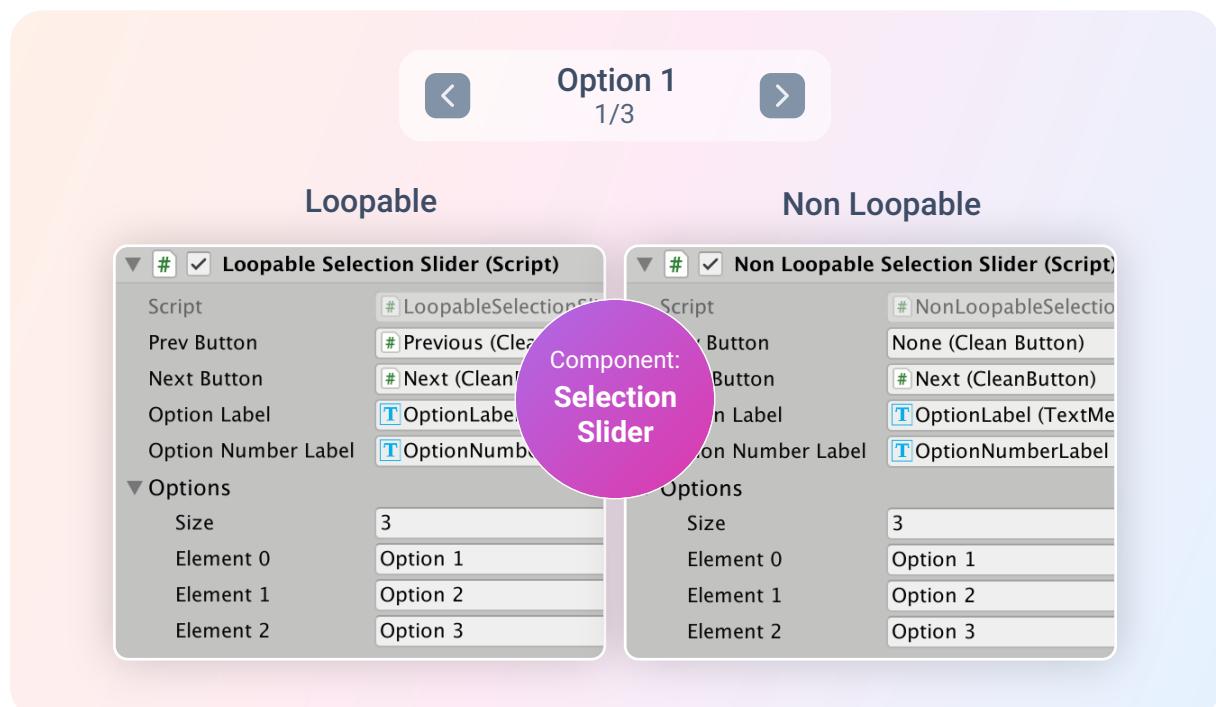


We also provide a Unity's built-in slider for each theme.

4.3.7. SelectionSlider

This component provides a slider that allows you to select between a set of fixed, predefined options.

Both a **loopable** and a **non-loopable** version are provided.

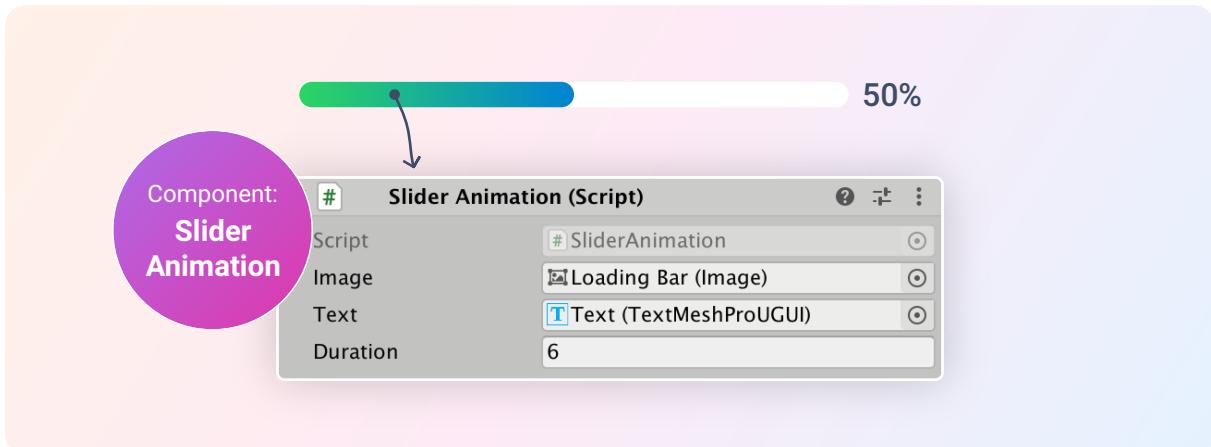


In the Options array, you can define the fixed options that you will be able to go through at runtime. You can use as many options as you like.

4.3.8. ProgressBarAmountText / SliderAmountText

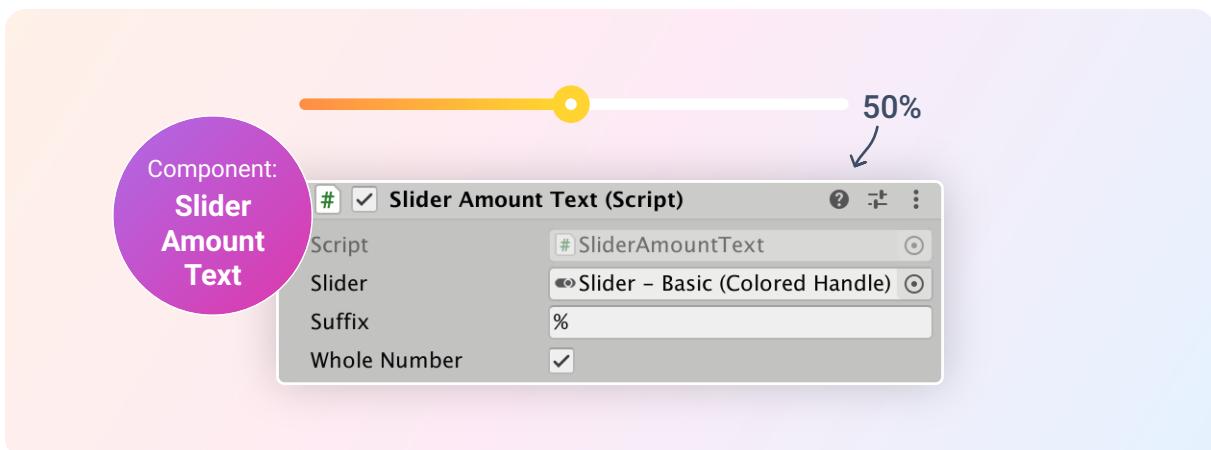
These components are used to link the value of a text label with the current amount of a progress bar/slider.

Progress Bar



For the progress bars you have to add the Slider Animation Component to the Loading Bar. You can choose the Duration.

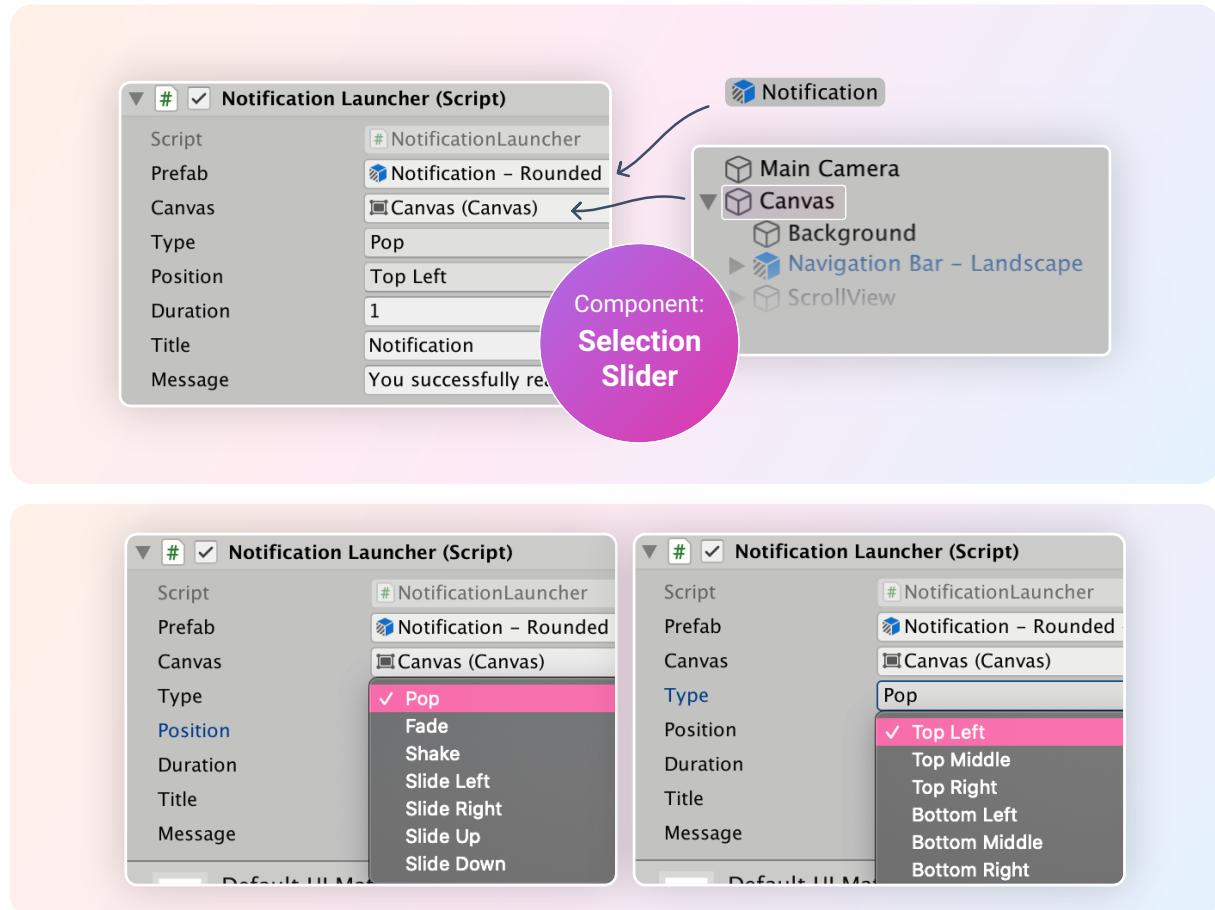
Slider



For the Slider you have to add the Slider Amount Component to the Text. You can choose it to any Suffix.

4.3.9. Notification / NotificationLauncher / NotificationQueue

This component allows you to open a notification prefab. This prefab is required to have an associated Notification component.



You can specify the following settings in this component:

- **Prefab:** The notification prefab to launch.
- **Canvas:** The canvas in which to instantiate the notification.
- **Type:** The type of animation with which to open the notification.
- **Position:** The screen position in which the notification should appear.
- **Duration:** The time (in seconds) the notification should be present.
- **Title:** The title of the notification.
- **Message:** The message of the notification.

In order for the transition to be performed, you need to call the component's `LaunchNotification` method. We usually do this from a button's `OnClick` action.

The `NotificationQueue` component is used to show notifications that are launched at the same (or similar) time to be displayed nicely one after another and prevent any kind of overlapping between them. This object is `DontDestroyOnLoad`, so we only add it to the initial scenes in the demo.

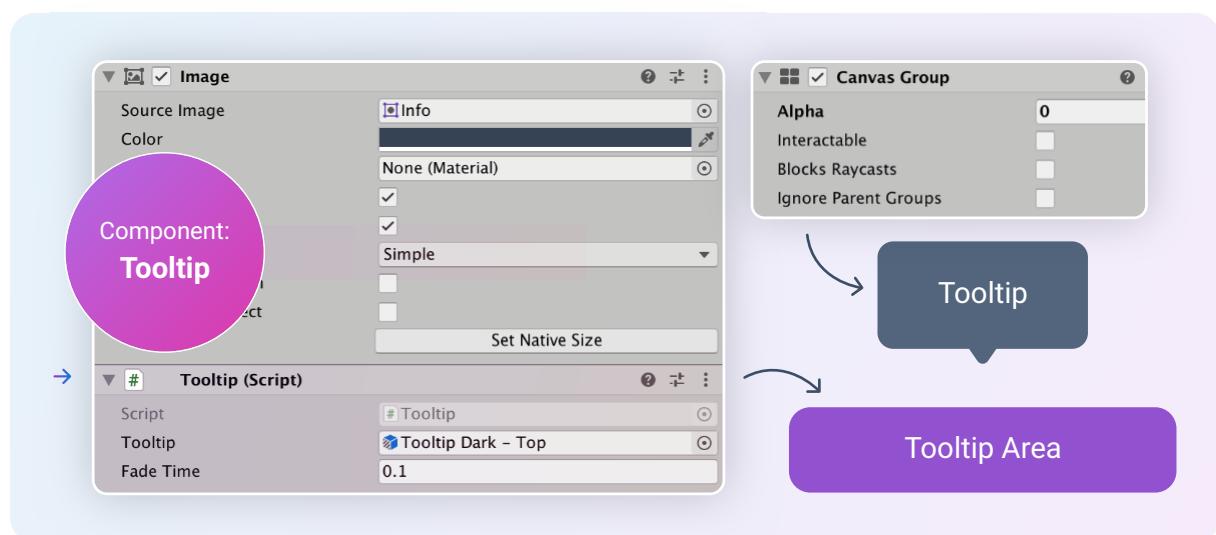
4.3.10. Tooltip

This component allows you to open a tooltip prefab.

You can specify the following settings in this component:

- **Tooltip:** The tooltip prefab or game object to show.
- **Fade Time:** The fade in/out time (in seconds).

The tooltip will appear/disappear automatically as the player enters/exits the associated UI image.



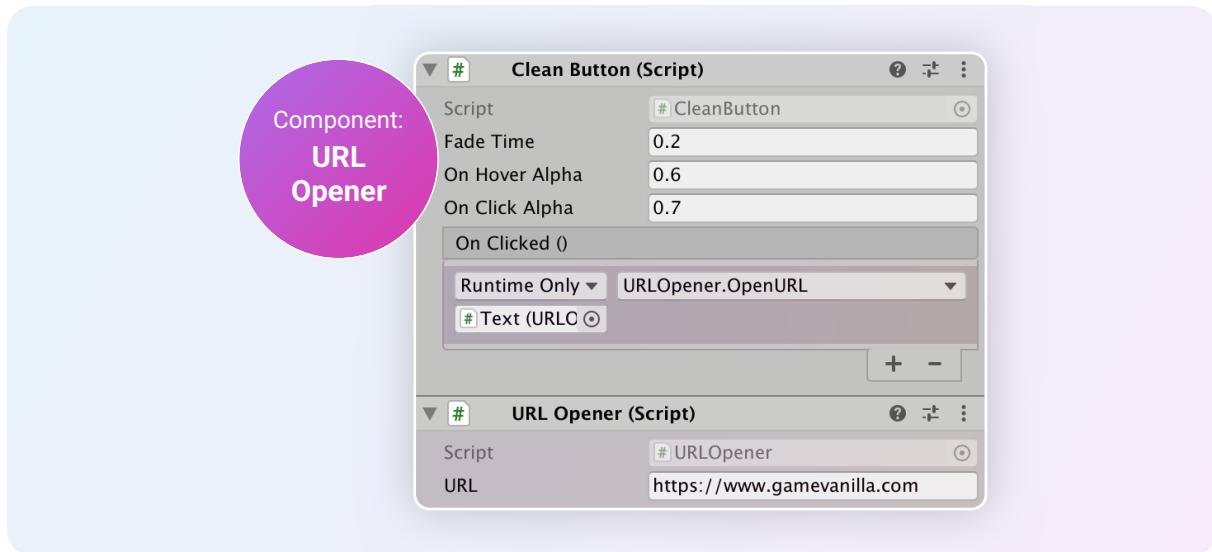
You can find our pre-made tooltip prefabs with a size fitter component (the size of the window fits automatically to its content) inside the folder [prefabs/UI Elements/Tooltip](#) for every theme.

Besides, you can literally choose any game object, text or single sprites as a tooltip. Just drag the tooltip-item into the Tooltip section.

4.3.11. URLOpener

The URLOpener component provides functionality to open a website from an UI element. You specify the URL to open in the URL field. In order for the transition to be performed, you need to call the component's OpenURL method.

We usually do this from a button's OnClicked action, as you can see in the example screenshot.



5. Contact

If you have any questions or suggestions, please feel free to contact us via our support website on gamevanilla [here](#).

We are always happy to help. 😊

6. Copyright & Terms of use

The **Ultimate Clean GUI Pack** can only be used under the [Unity Asset Store License Agreement](#) which you can find [here](#).

The copyright of the Ultimate GUI Pack and all of its contents belongs to ©gamevanilla.

After purchasing the Ultimate Clean GUI Pack, you have the right to use it only for the purposes of developing and publishing a game.

You are NOT allowed to redistribute or resale the Clean & Minimalist GUI Pack or any of its contents for any purpose. To distribute or resale this product is NOT permitted under any circumstances and is strictly prohibited.

Thank you for respecting our work.



Ultimate Clean GUI Pack

Multipurpose Game UI

© gamevanilla

Copyright © gamevanilla - All rights reserved.

© gamevanilla
gamevanilla.com



@ricimiart



@gamevanilla