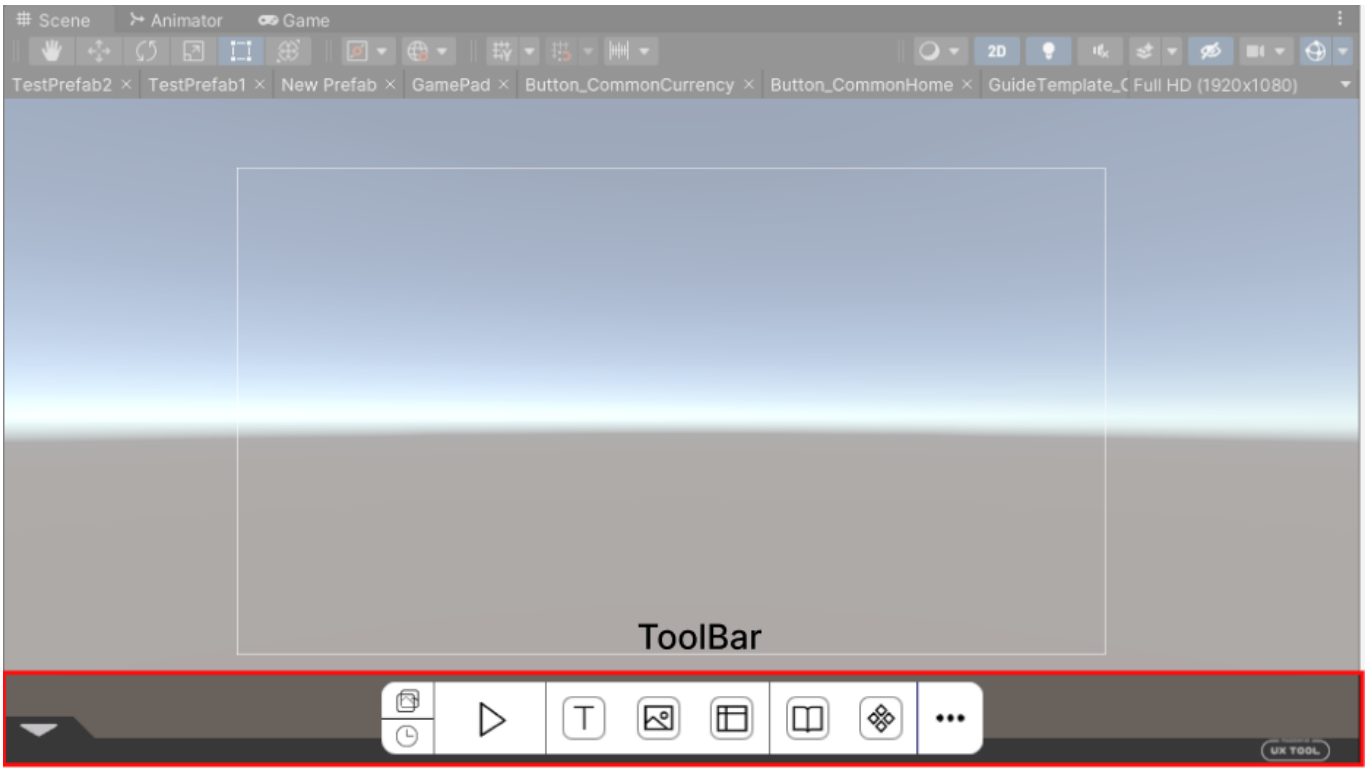
Using Tutorials

Introduction to the Tools Panel

ToolBar

The ToolBar is a supplementary toolbar located at the bottom of the Scene panel, designed to facilitate rapid interface stitching. It can be enabled or disabled through the Unity top bar [ThunderFireUXTool - ToolBar].



Panel Functions

- Quick Background: Create a background for reference in the scene or Prefab. Then create the interfaces based on the quick background.
- Recently Opened: Open the Recently Opened panel to see the most recently opened Prefab.
- Preview: Preview the currently opened Prefab in the Game panel.
- Text: Create a UXText object in the Scene or Prefab, and draw a checkbox of UXText by dragging and dropping.
- Image: Create a UXImage object in the scene or Prefab, and draw a checkbox of UXImage by dragging and dropping.
- Reference Lines: Create a set of reference auxiliary lines in the Scene or Prefab.

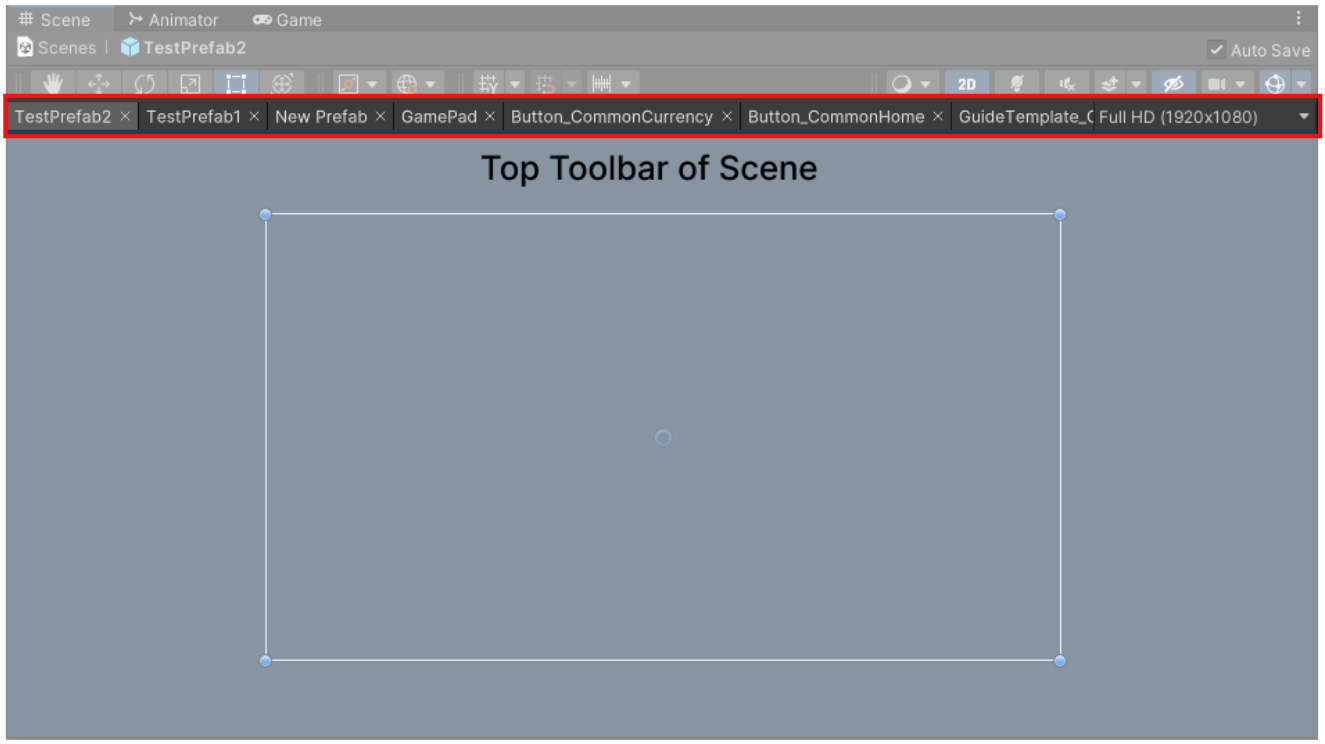
- Widget Library: Open the Widget Library panel to view all components.
- New Widget: Stow the currently selected node or Prefab as a component in the Widget Library.
- More: Opens the Tool Information panel or the Settings panel.

Hide Panel

The open ToolBar can be hidden by clicking the small triangle button located on the left side of the ToolBar. To open it, click the same button again.

Top Toolbar of Scene

In the Scene panel, below the toolbar in Unity itself, a new UX toolbar is added to display the Prefab tab and adjust the resolution.



Prefab Tab

Each new Prefab opened by the user in the project is displayed as a tab in the toolbar. Clicking on a tab opens the corresponding Prefab directly.

When the number of tabs exceeds the display space in the toolbar, you can slide to display them by scrolling the mouse wheel.

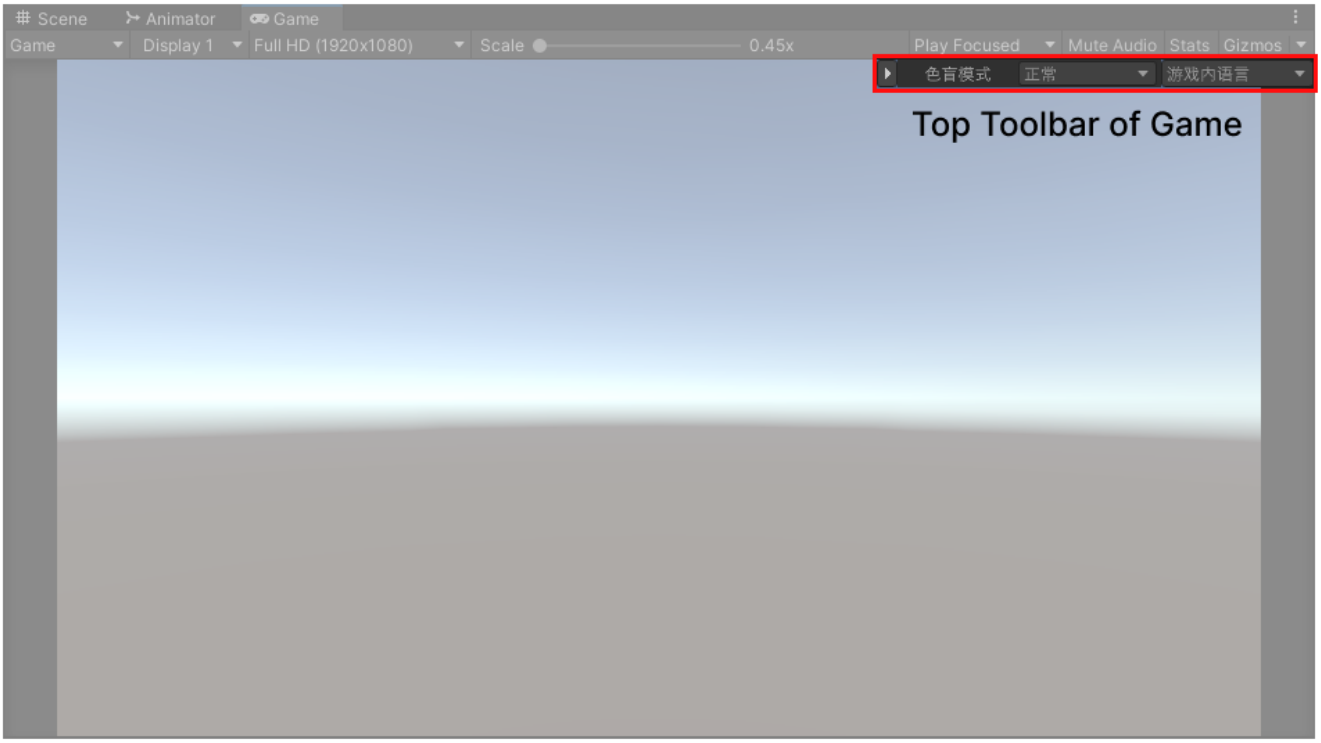
Click the × sign of a tab to close the corresponding Prefab in the tab.

Adjust the Resolution

The rightmost part of the toolbar is the canvas resolution, which is always the same as in the Game panel. If one of them is modified, the other one will change as well.

Top Toolbar of Game

In the Game panel, a new UX toolbar has been added to the bottom right of the toolbar in Unity itself to toggle the content while the project is running. This toolbar can be collapsed by clicking the small triangle button on the left, and expanded when clicked again.

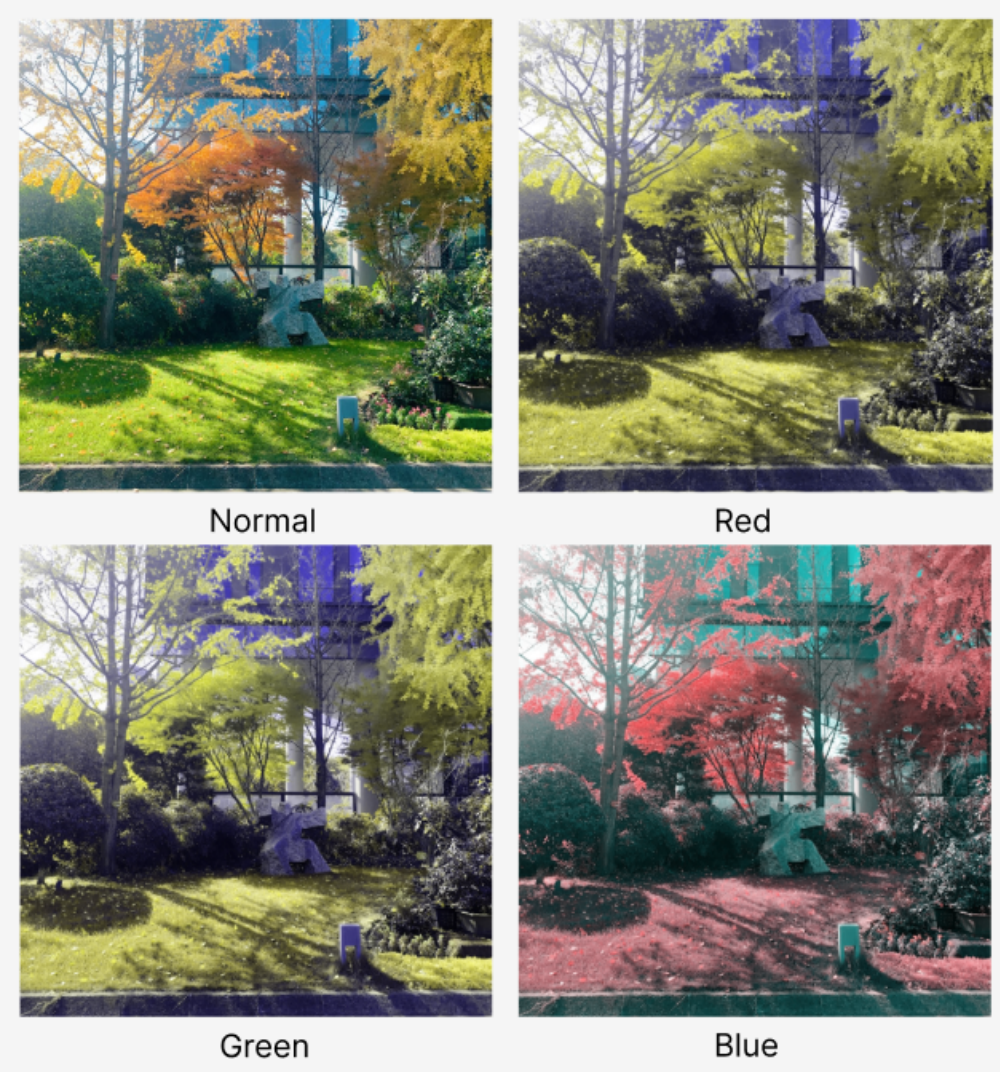


Color Blind Mode

Adjust the color mode of the project in Unity to preview how the interface will appear for individuals with color vision deficiencies.

The default setting is normal, which reflects the appearance of the interface in standard states.

You can switch between red, green, and blue colorblind modes using the drop-down menu (refer to documentation on color blind mode for more information).



Switch Language

Preview how the project interface will look like in different languages. Check the language used in the project in Localization-Settings-Language to switch the corresponding language at runtime and see how the localized text and images appear in the interface.

The default language is the in-game language, i.e. the language initially used in the title. and the in-game language is prioritized to display at this time.

In addition, there are Text-Free Mode and Key Mode in the drop-down list:

- Text-Free Mode: All the text in the interface is displayed in the form of placeholders, which is used to test the usability of the interface.
- Key Mode: All the text in the interface is displayed as its corresponding KEY for quick location in localization assets.

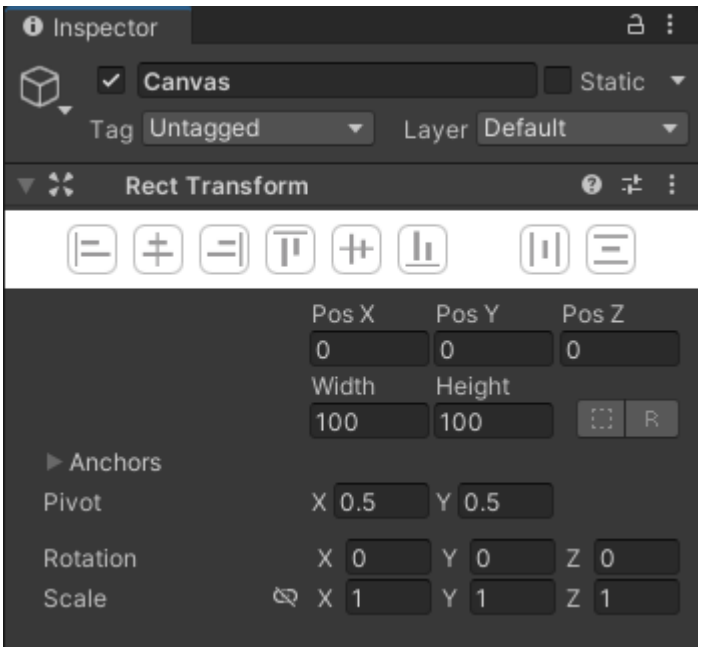
(Refer to the localization function document for more details)

Layout Tools

ThunderFire UX Tool provides basic UI layout tools to help designers to make quick and efficient interface stitching in Scene. The layout tool can be turned on or off in [ThunderFireUXTool - Settings (Setting) - Function Switch].

Alignment and Distribution

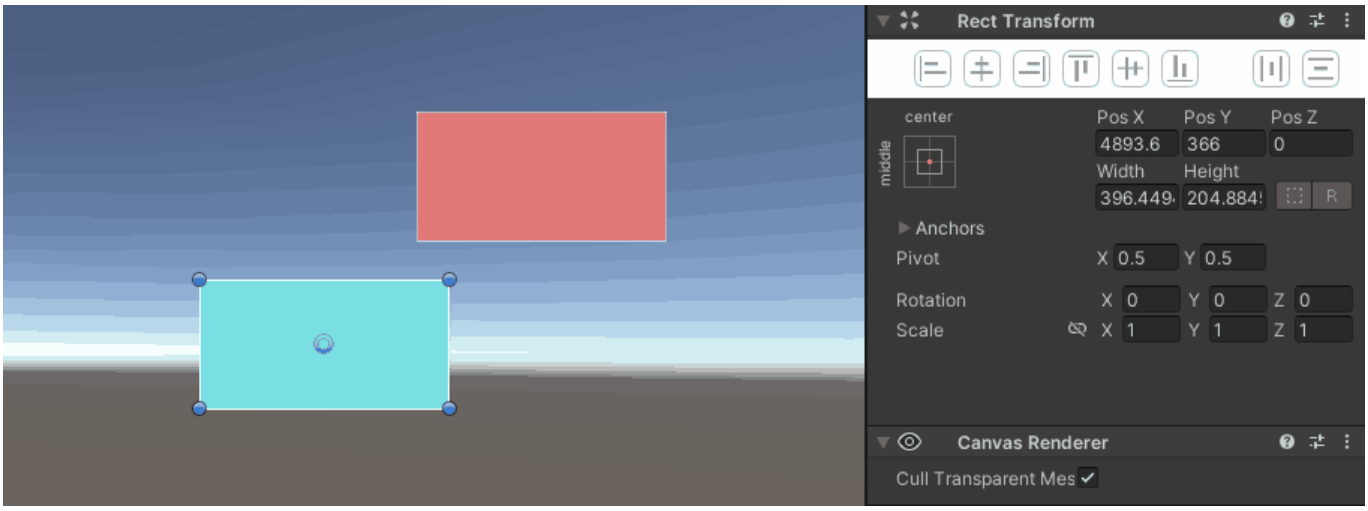
The Alignment and Distribution tool is located in the Inspector panel, under the Rect Transform component, and provides 6 types of alignment: left alignment, horizontal center alignment, right alignment, top alignment, vertical center alignment, bottom alignment, and 2 types of distribution: vertical distribution and horizontal distribution.



Align Aligns 2 or more objects along the selected direction.

- **Left Alignment**：Align the positions of 2 or more objects to the left edge of the leftmost object.
- **Horizontal center alignment**: Align the positions of 2 or more objects to the vertical centerline of the object as a whole.
- **Right Alignment**: Align the positions of 2 or more objects to the right edge of the rightmost object.
- **Top alignment**: Align the position of 2 or more objects to the top edge of the topmost object as the reference alignment.
- **Vertical center alignment**: Align the position of 2 or more objects to the horizontal center line of the object as a whole as a reference alignment.
- **Bottom alignment**：Align the position of 2 or more objects to the bottom edge of the bottommost object as a reference.

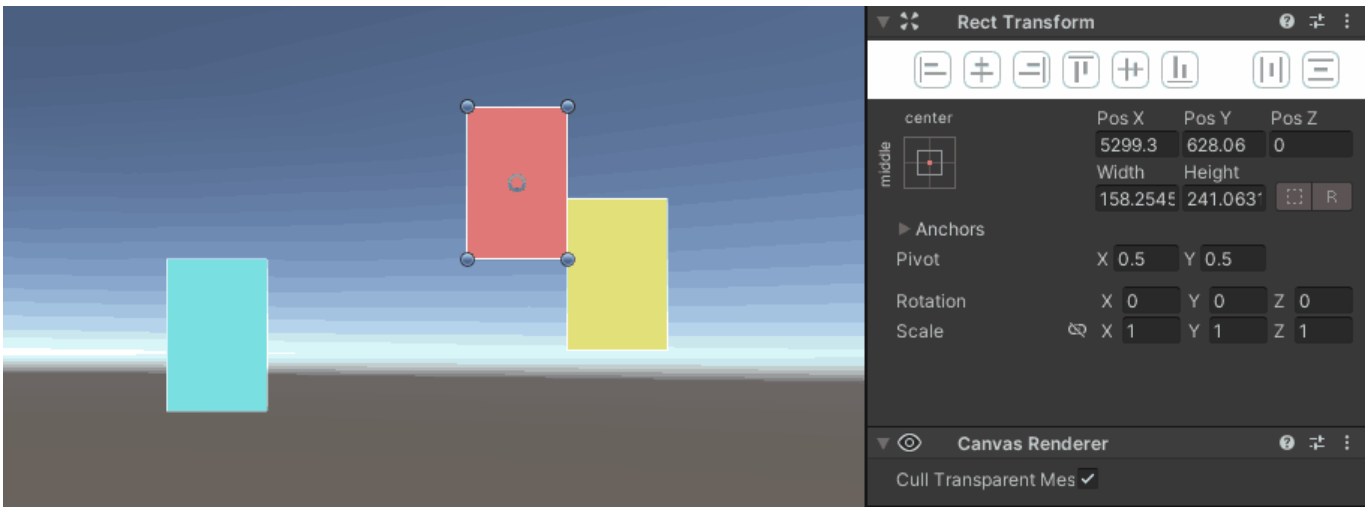
Note: Only when two or more objects are selected, the alignment tool becomes available. When the object-related parameters in the selected objects are locked (such as when they cannot be changed due to Layout settings), the alignment tool is not available.



Distribution Arrange 3 or more objects equally spaced along the selected direction.

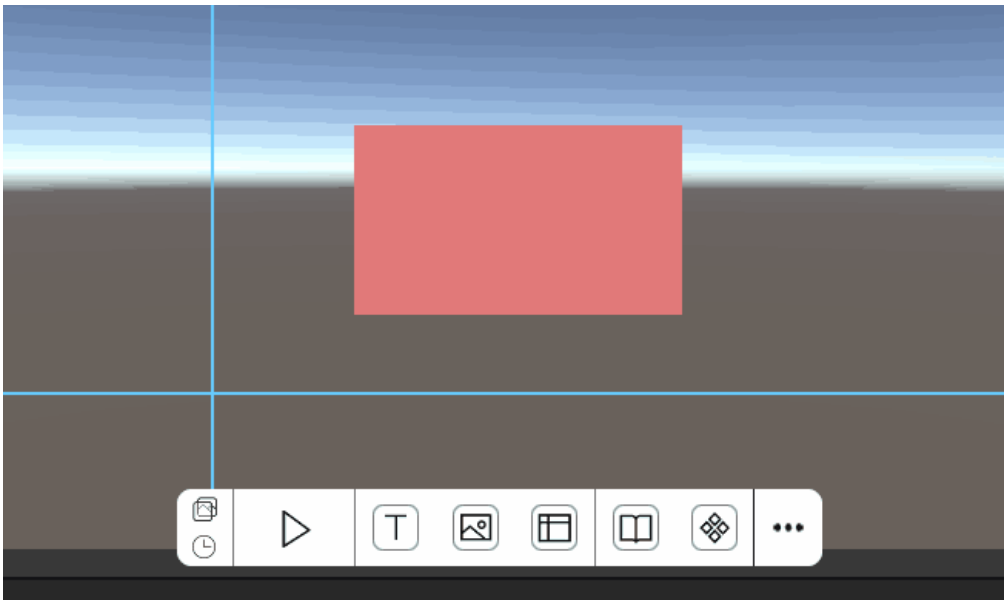
- **Horizontal distribution**: select 3 or more objects while keeping the horizontal alignment order unchanged and the positions of the objects at the left and right ends unchanged. Adjust the position of the middle object to ensure that equal distances are maintained between adjacent objects on both sides.
- **Vertical distribution**: select 3 or more objects, while keeping the vertical alignment order unchanged, the position of the upper and lower ends of the objects unchanged. Adjust the position of the middle object to ensure the equal distance between adjacent objects on upper and lower edges.

Note: Only when two or more objects are selected, the alignment tool becomes available. When the object-related parameters in the selected objects are locked (such as when they cannot be changed due to Layout settings), the alignment tool is not available.



Reference Line

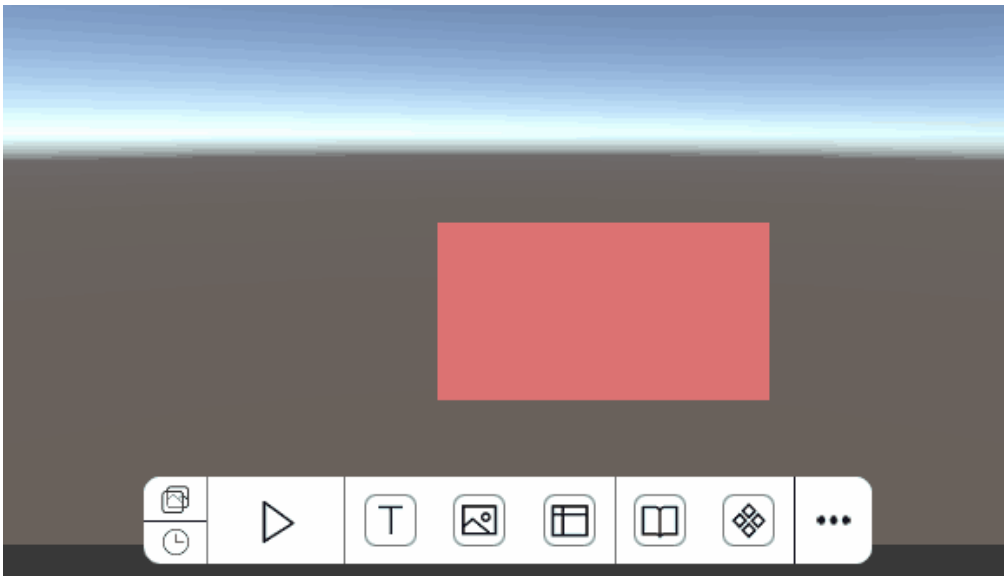
Add a reference line in Scene, and when dragging an object such as a picture or text close to the reference line, the object can automatically attach the edge to the nearby reference line.



Add Reference Lines : In a 2D scene or Prefab, click the Reference Lines button in the ToolBar to add a set of reference lines to the scene.

Move reference lines : Drag a created reference line to change its position. When moving a reference line close to an object in the scene, the reference line will automatically attach to the edge of the object.

Delete reference line : Right click on the created reference line and select "Delete" to delete the reference line in the scene.



Snap-to-Move

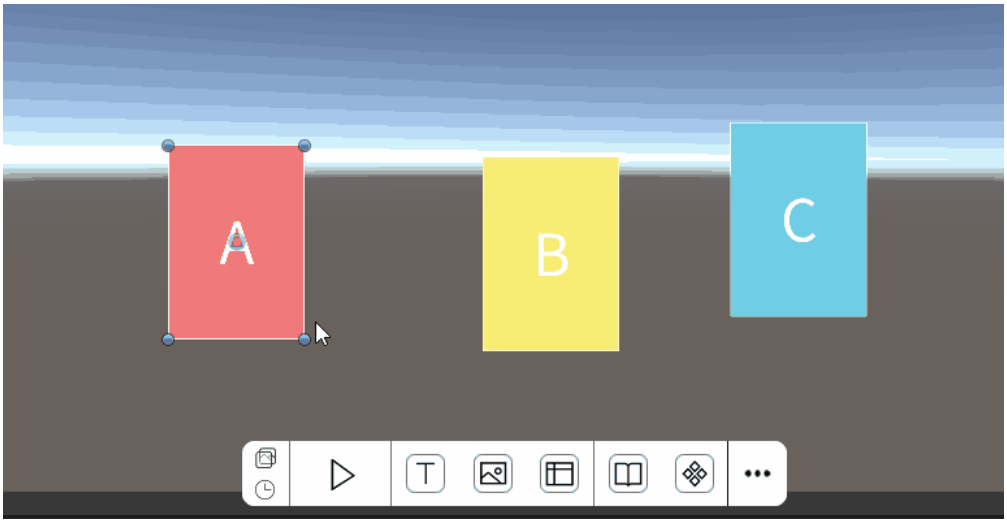
Snap-to-Move is a function that generates multiple automatic alignment effects, including isometric, edge, and center alignment, when editing multiple objects in a row in the Scene.

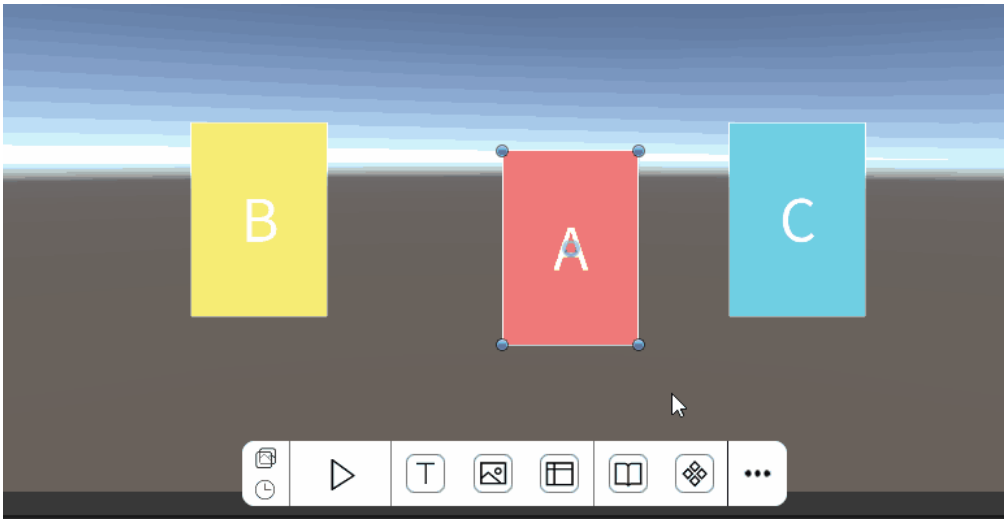
Isometric adsorption

- Select object A to move, when the distance between it and B is close to the distance between B and C on the other side, object A will automatically snap to the exact position of the isometric distance and

display the numerical value of this distance. (Here the distance refers to the distance between the edge lines of two objects relative to each other)

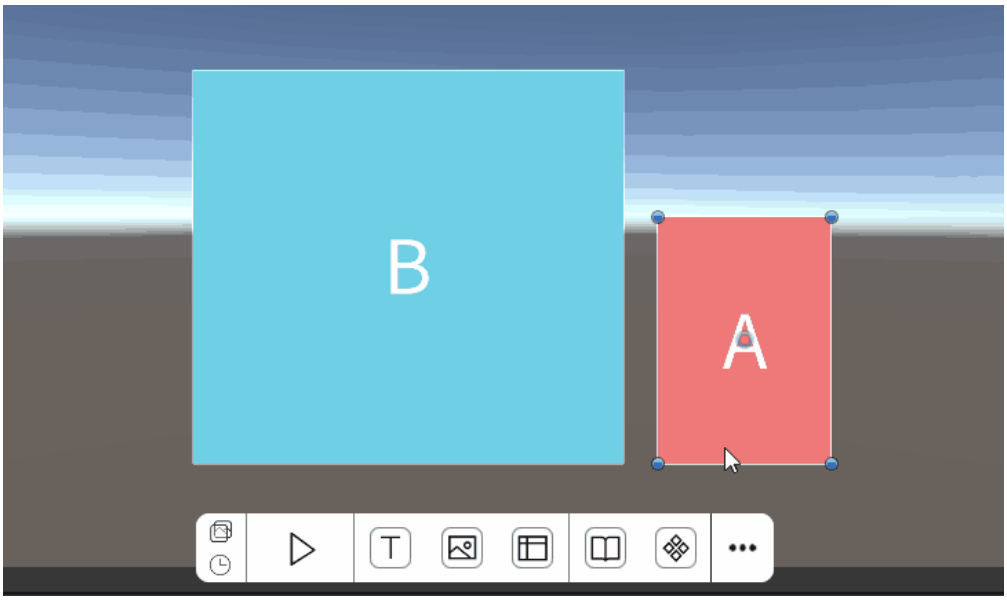
- Select object A to move, when the distance between it and B is close to the distance between A and C on the other side, object A will automatically snap to the exact position of the equal distance, and display the numeric identifier of the distance. (Here the distance refers to the distance between the edge lines of the two objects relative to each other)





Edge/Center adsorption

- Select object A to move, when its horizontal (vertical) center line or edge line is close to B's horizontal (vertical) center line, edge line or edge extension line, the former will automatically snap to align to the latter.



Shortcuts

ThunderFire UX Tool provides a variety of shortcut tools in the interface to make the operation closer to the common design software, thus enhancing the efficiency of interface creation. Shortcuts can be turned on or off in ThunderFire UX Tool - Settings - Function Switches.

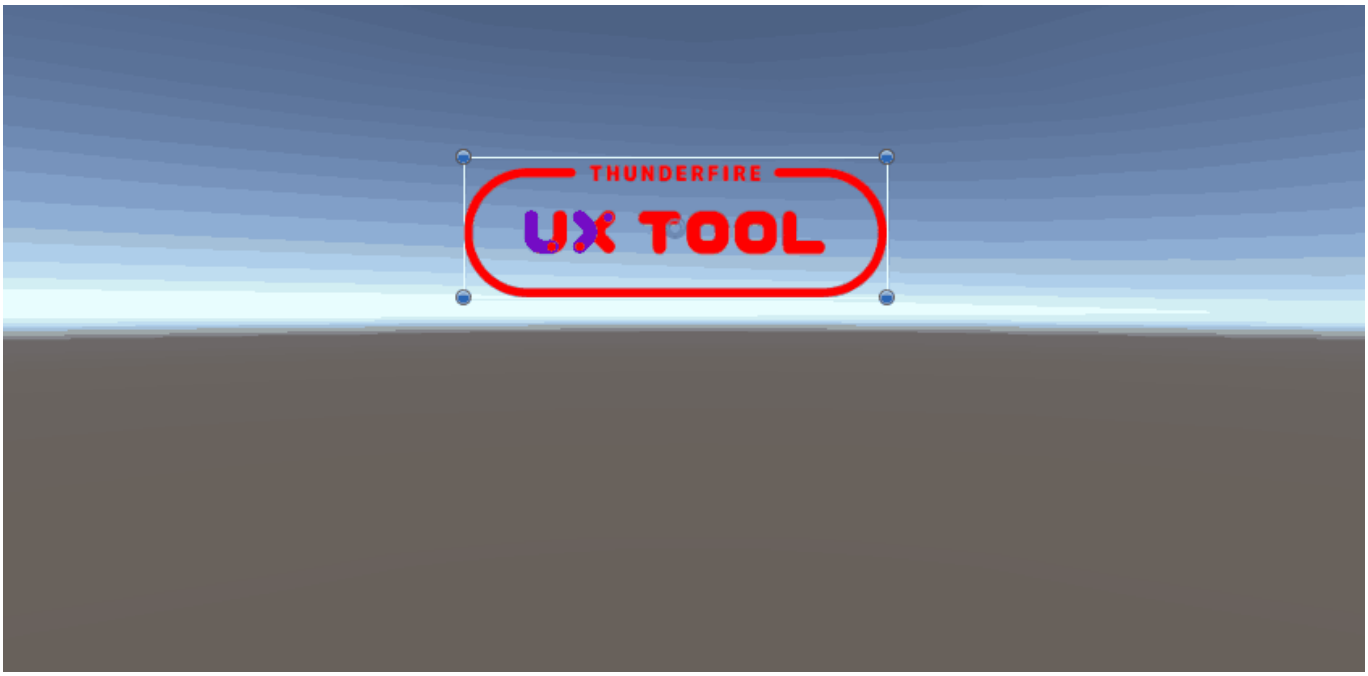
Position Nudging

When one or more objects are selected in the 2D scene or Prefab, use the up, down, left, and right arrow keys on the keyboard to slightly and precisely move the selected objects.

Note: This feature overrides Unity's original feature of "Keyboard Move Increment".

Copy, Paste, Delete and Combine

When one or more objects are selected in a 2D scene or Prefab, the right-click menu allows you to copy, paste, delete, and combine the selected objects.



Copy : Copies the selected object to the clipboard.

Paste : Creates the object in the clipboard under the currently selected node.

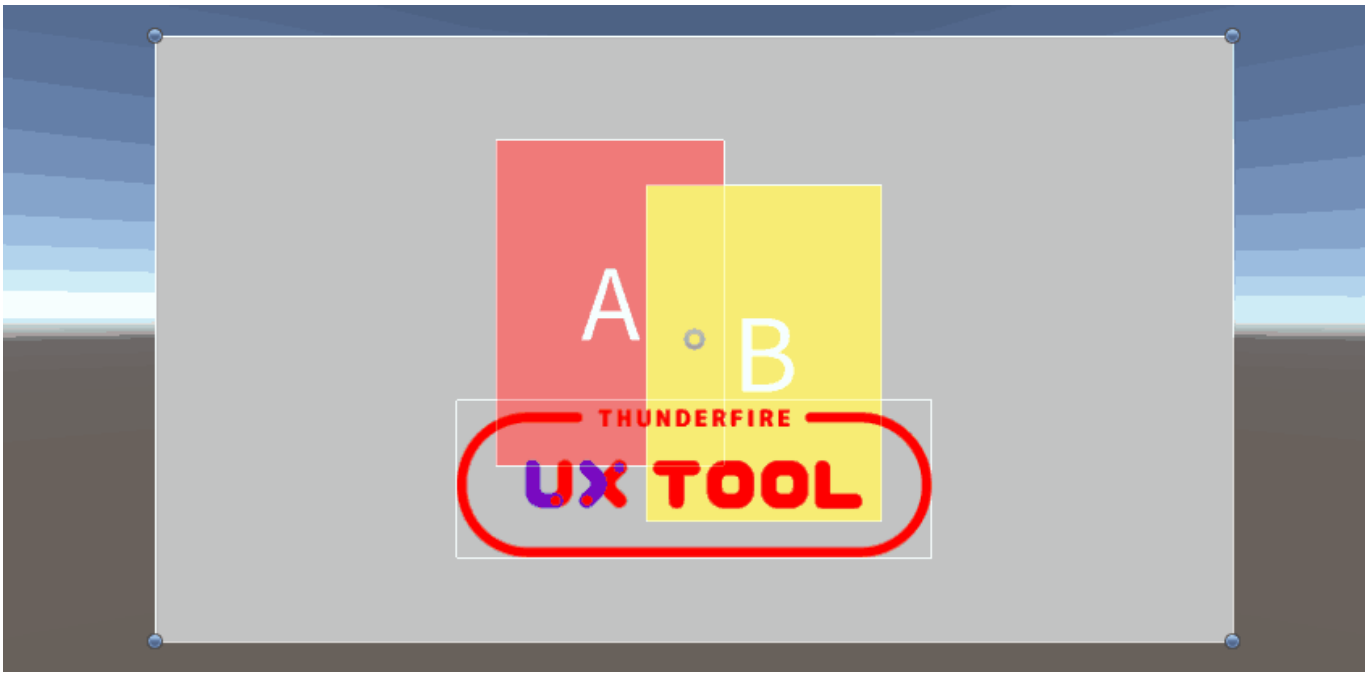
Delete : Delete the selected objects.

Combine : Quickly add a parent node named root to the selected 2 or more objects, and the selection box of the parent node is consistent with the selection box formed by the outer edges of all child nodes.

Quick select of objects

In 2D scenes or Prefab, objects can be selected by right-clicking on the list.

Depending on the position of the mouse at the time of right-clicking, all objects under that pixel point will be displayed in the list in order from top to bottom, and the object can be selected directly after clicking on it. If there are objects with the same name in the list, the objects closer to the bottom will be distinguished by the "[n]" suffix.

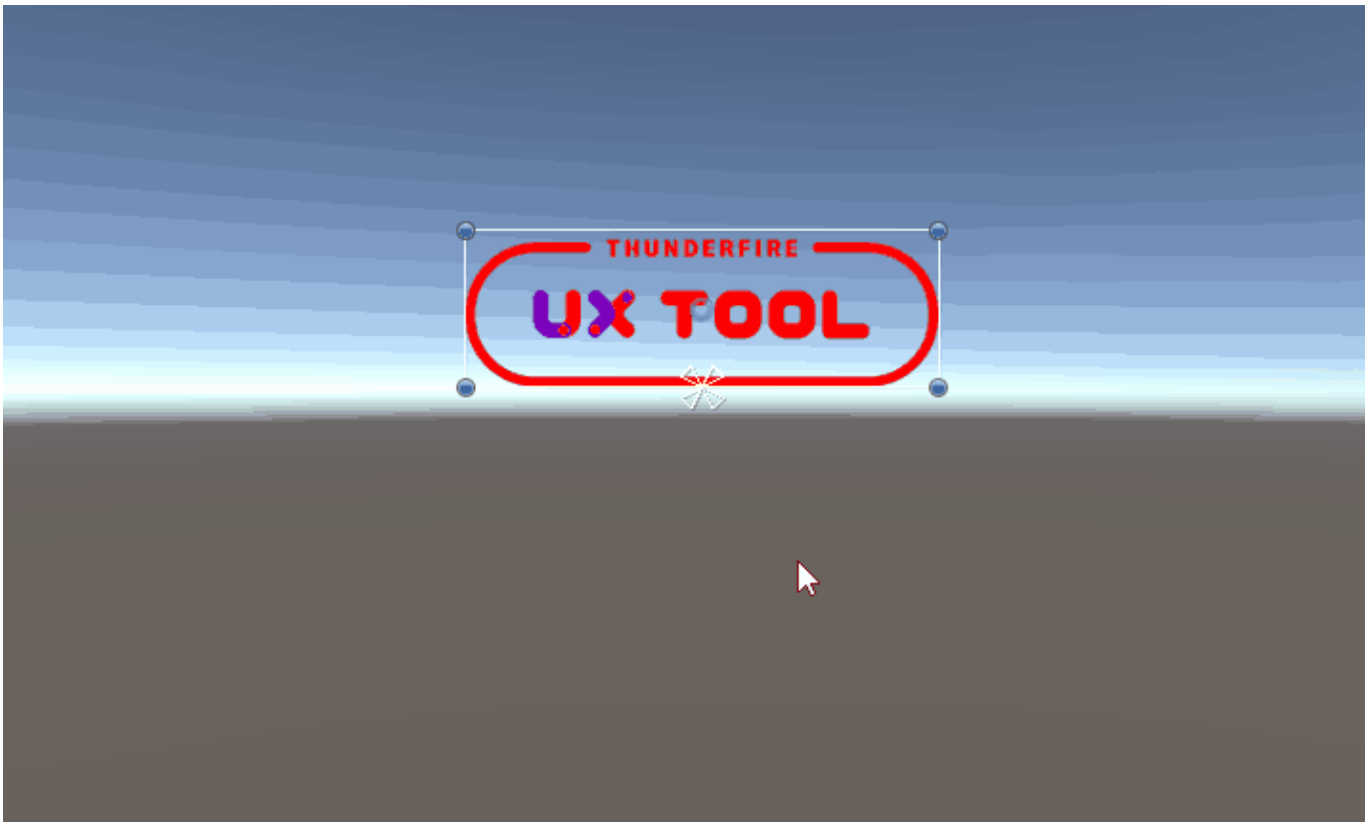


Quick Copy

In 2D scene or Prefab, when one or more objects are selected, hold down Alt during dragging with the left mouse button, the currently dragged object will be copied. Release the left mouse button to paste.

Note:

- If the left mouse button is not released before Alt during copying, the copied objects will be lost and the copying process will be interrupted.
- If the Layout tool settings are enabled, the distance and automatic alignment between the copied object and other objects in the scene will also be displayed during the quick copy process.



Widget Library

The Widget Library in ThunderFire UX Tool is a tool that organizes, categorizes and manages the user's commonly used prefabs in the form of widgets.

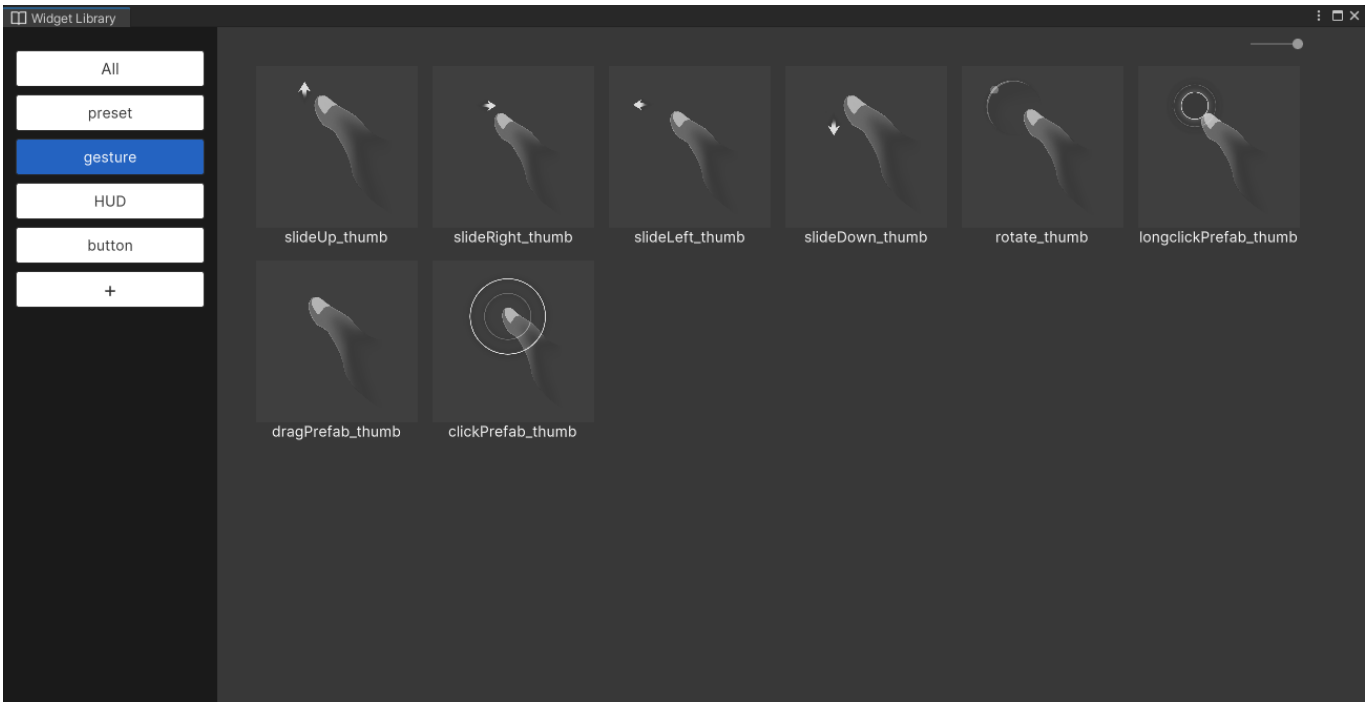
Terminology

Widget

A widget is a generic and reusable prefab created by the designer during interface stitching in Unity, for example, a button that appears repeatedly in multiple interfaces can be stored as a widget for quick and easy reuse.

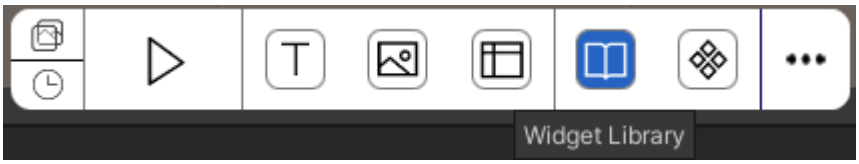
Widget Library

The widget library is used to categorize and organize widgets to make it easier for users to manage the widgets they create.



How to Use

Open via ToolBar : Enable ToolBar via the Unity top drop-down menu [ThunderFireUXTool-工具栏 (ToolBar)], then click the Widget Library button on ToolBar to open the Widget Library panel.



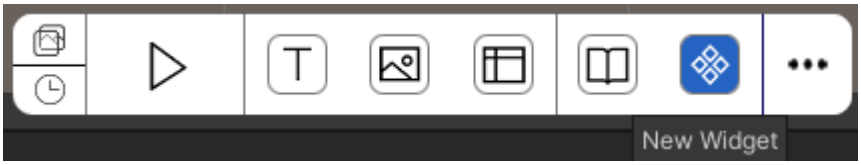
Open through the top drop-down menu : Click the Unity top drop-down menu [ThunderFireUXTool-组件库 (Widget Library)] to open the Widget Library panel as well.

In the Widget Repository panel, the widget category tab is on the left side, click the corresponding tab to see all widgets under that category on the right side. Click the "+" sign to create a new widget category. Right-click on an existing category tab to modify the category name or delete the category.

Creating widgets

Widgets can be created in three ways. Any Prefab or non-Prefab node can be created as a widget.

Create by ToolBar : Enable ToolBar through the drop-down menu [ThunderFireUXTool-工具栏 (ToolBar)] at the top of Unity. After selecting a prefab or node in the Scene, click the [New Widget button] on the ToolBar, a New Widget pop-up window will appear, fill in the information and the creation is complete.



Create by dragging and dropping : Open the Widget Library panel, directly drag and drop the prefab or node from Hierarchy or Project to the Widget Library panel, a New Widget pop-up window will appear, fill in the information to create it.

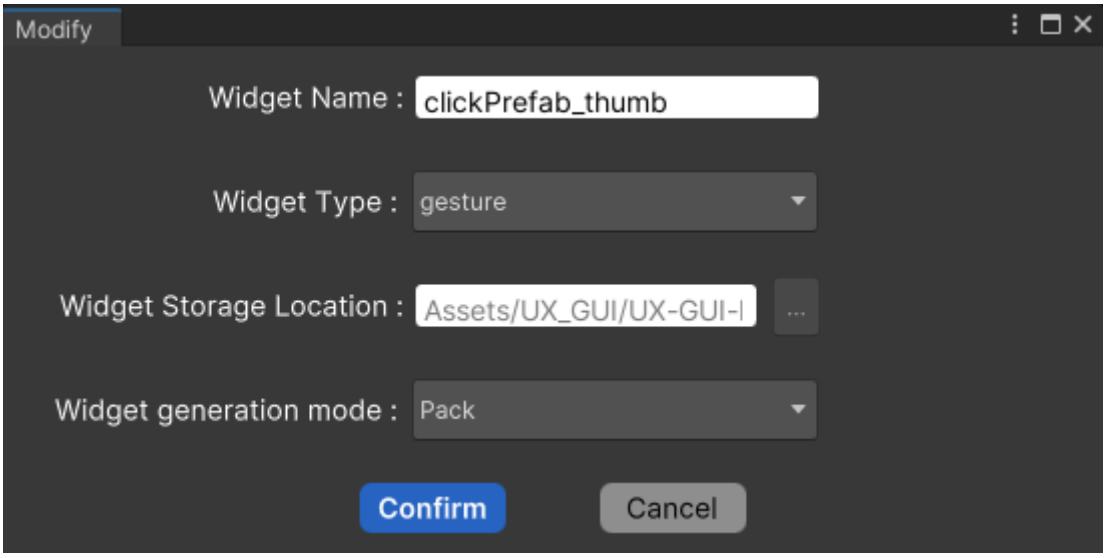
Create by right-click :Select a prefab in Project and right-click [设置为组件(Set As UXWidget)], a New Widget pop-up window will appear, fill in the information to create it.

Managing widgets

The Widget is displayed in the Widget Repository as a thumbnail, double-click it to open the corresponding prefab of the widget directly.

Modify information

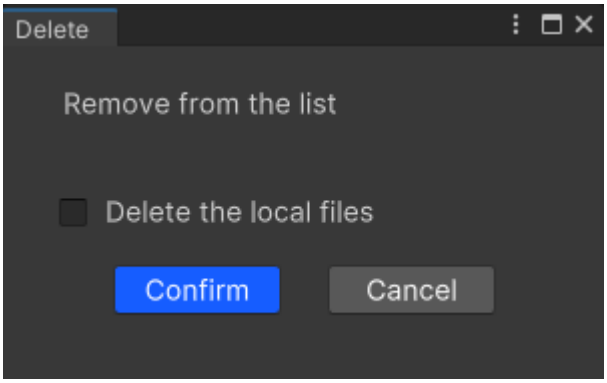
Right-click the widget in the Widget Repository to modify the widge properties, including widget name, widget type, widget storage location and widget generation mode.



- **Widget Name**: This refers to the name of the prefab that corresponds to the widget. If you make any changes to the widget name, it will also modify the name of its corresponding prefab.
- **Widget Type**: By default, this is set as "All", and when you create a new widget by dragging and dropping it, its current classification will be automatically recognized. You can also create a new type in the Widget Information panel by clicking on "+" sign.
- **Widget Storage Location**: The location of the corresponding prefab in the project, which cannot be modified after setting once.
- **Widget Generation Mode**：
    - Prefab: i.e. the widget is added to the scene or prefab in the form of prefab.
    - Unpack Prefab: i.e. the widget is unpacked from the prefab and added directly to the scene or prefab as a normal node.

Delete widget

To delete the widget, right-click on it in the Widget Library. You can choose to remove it from the Widget Library list only or also delete its corresponding prefab local file.



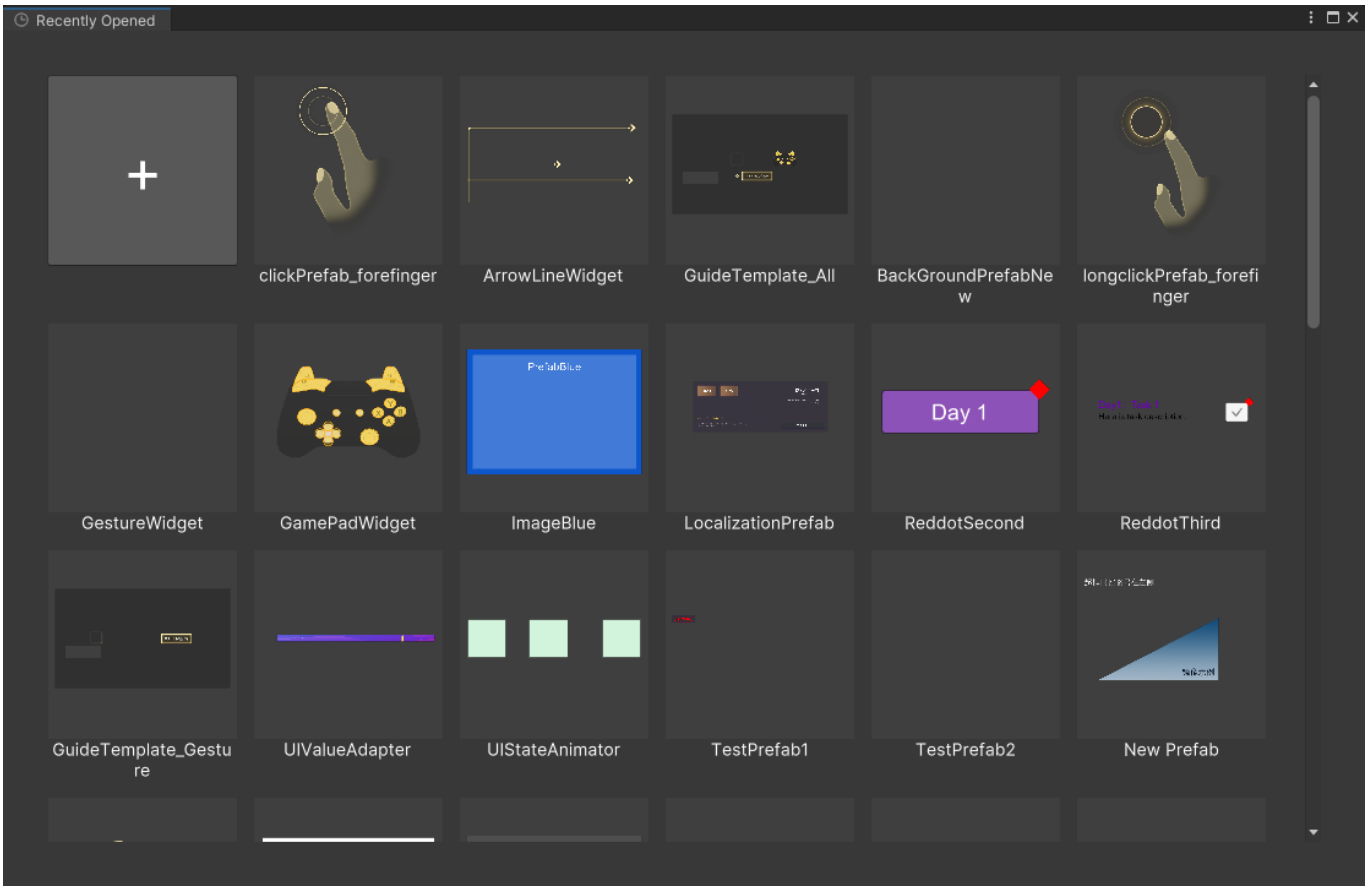
Using widgets

The created widget can be used in the Scene or prefab by opening the Widget Repository panel and dragging the corresponding widget directly to the Scene panel to generate it.

Recently Opened

The Recently Opened Panel in ThunderFire UX Tool is designed to enhance user experience by providing quick access to recently opened Prefabs, thereby reducing the time and effort required for locating them within a Project.

The Recently Opened Panel can be turned on or off in [ThunderFireUXTool - 设置 (Setting) - Function Toggle].



How to Use

Open via menu :Select [ThunderFireUXTool->最近打开 (Recent Opened)] in the menu.

Open via ToolBar:Enable ToolBar via the Unity top drop-down menu [ThunderFireUXTool-工具栏(Toolbar)], and click the [Recently Opened Template] button on the left to open the panel.

Note: The Prefab will be displayed in the Recently Opened panel in the order of recently opened. Double-click the Prefab thumbnail to open it quickly.

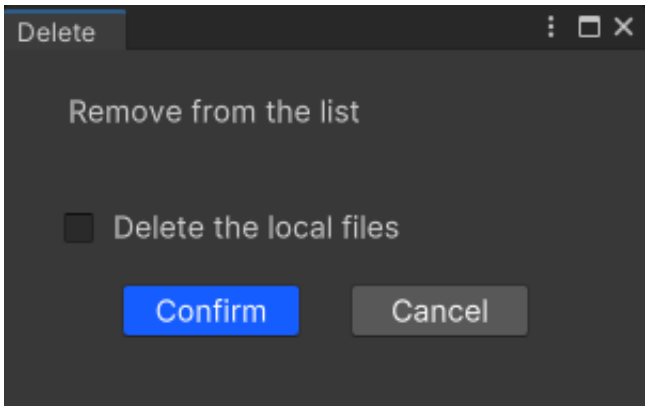
Panel Features

Automatically add the most recently opened prefab

When a Prefab is opened in unity, it is automatically added and displayed at the top of the list in the panel. Double-click on a thumbnail to open the prefab.

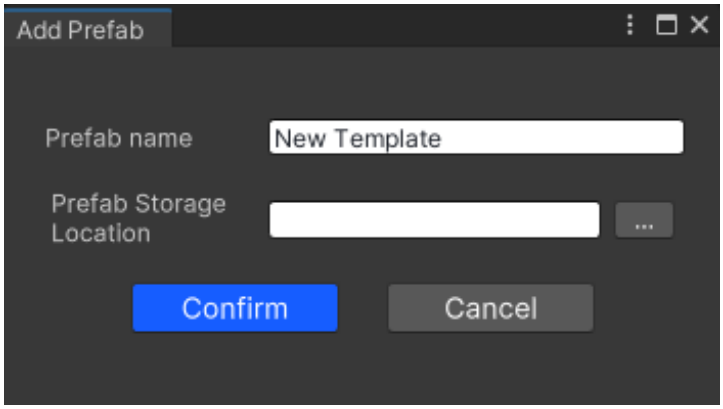
Deleting a prefab from the window

Select a prefab, right-click it and select "Delete", a pop-up window will appear, follow the prompts to delete it (you can choose to delete it only in the recently opened panel or delete the corresponding prefab local file at the same time).



Quickly create a prefab

Clicking "+" in the window will bring up the Add Prefab window. After setting the name and location, you can quickly create an empty Prefab.



Quick Background

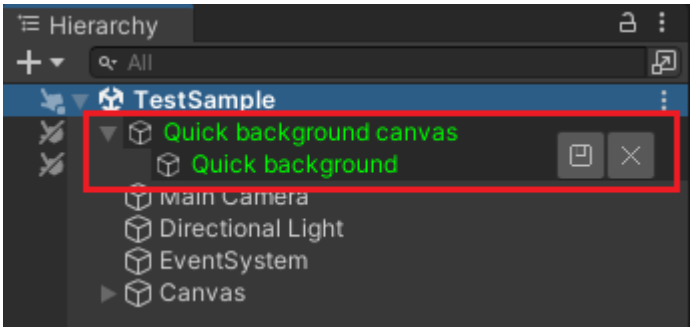
The Quick Background feature in ThunderFire UX Tool is designed to facilitate the rapid creation of a reference image for interface stitching or review, while allowing users to run or save it without impacting the original hierarchical tree structure.

How to Use

Open via menu : Select [ThunderFireUXTool->创建背景图 (Create QuickBackground)] in the menu.

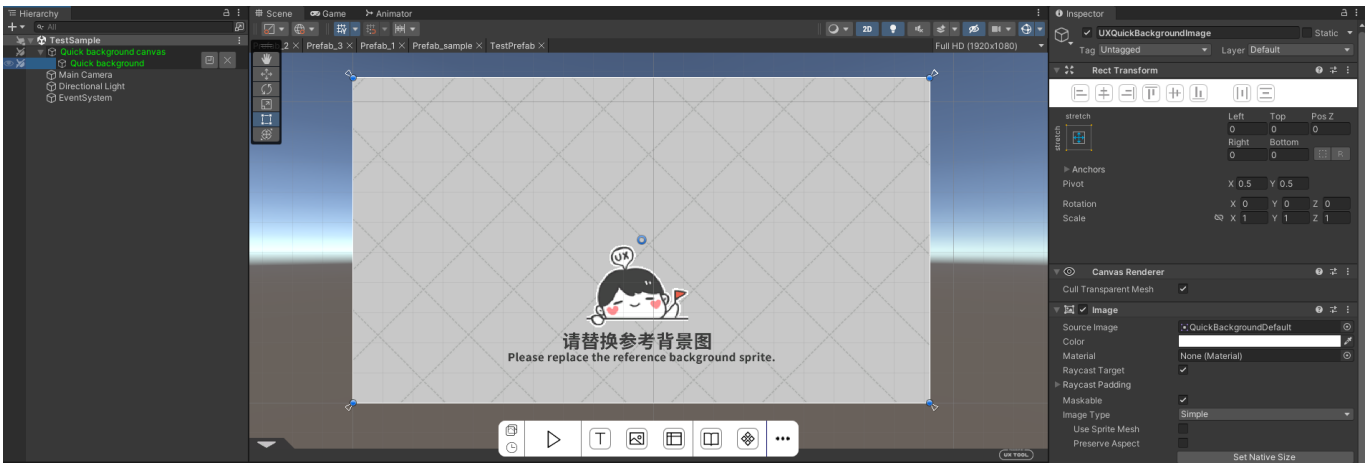
Open via ToolBar : Enable ToolBar via the Unity top drop-down menu [ThunderFireUXTool-工具栏 (ToolBar)], and click the [Quick background] button on the left to open the background image.

Once created, the "Reference Background Canvas" and "Reference Image" nodes will appear at the bottom of the Hierarchy panel under the root node, displayed as temporary objects in green.



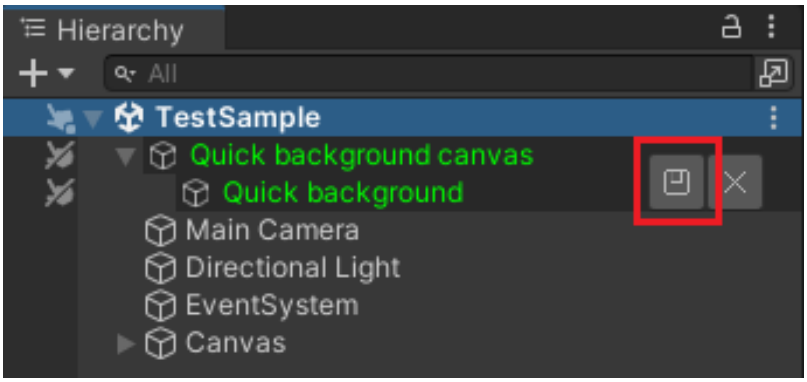
Quick background image in Scene

Settings In the Hierarchy panel, you can choose 'Reference Background Image' and modify its size, relocate it or change its information or replace it with another image in the Scene or Project panel.

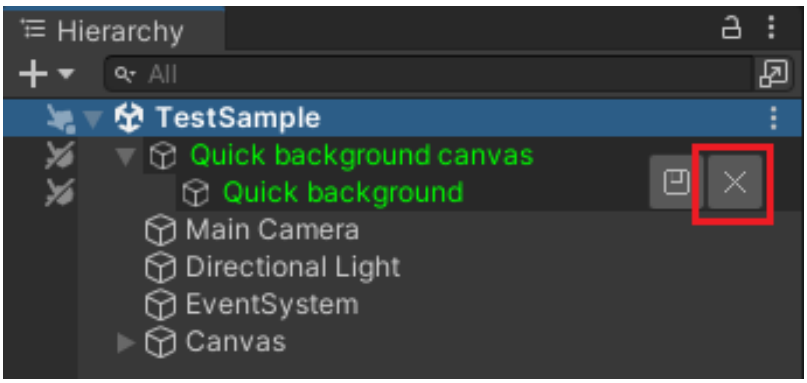


Note: Images cannot be selected directly in the Scene.

Save With the "Reference Background Canvas" expanded, click the Save button to save the changes.



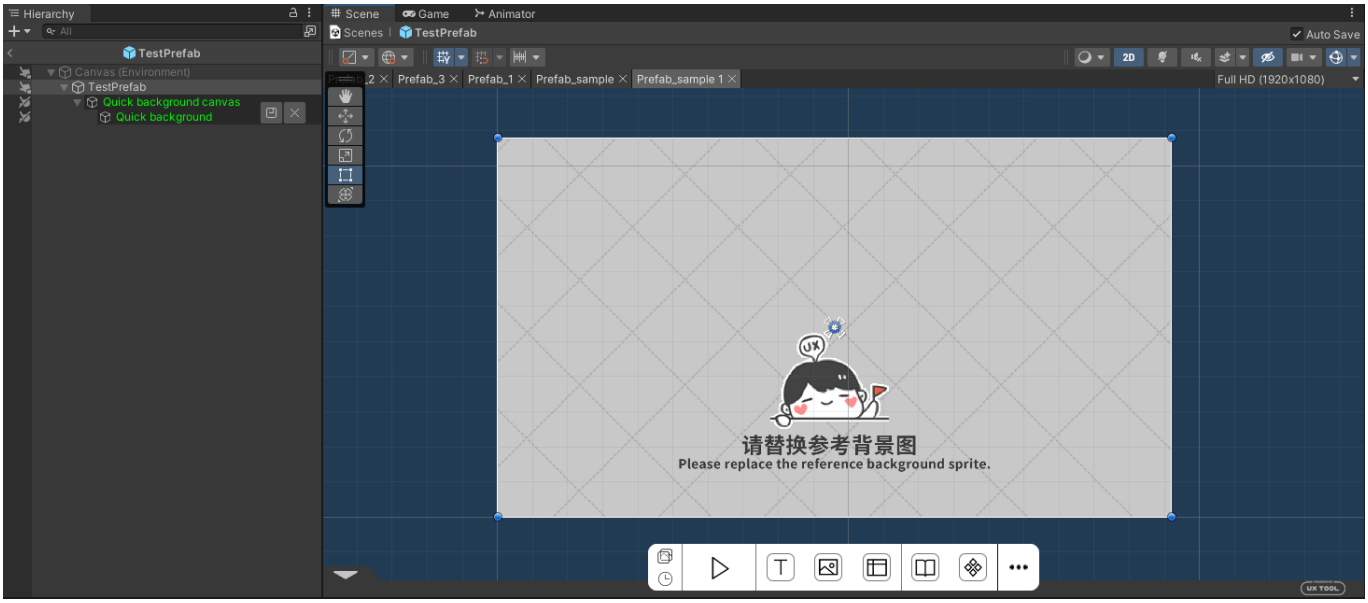
Close If the Reference Background Canvas is expanded, click the Close button to close it, and the Reference Background will be closed in the Scene.



Note: When Play mode starts, if the reference background is still present, it will be automatically removed to avoid affecting the scene; when exiting Play mode, the reference background will be restored unless manually turned off.

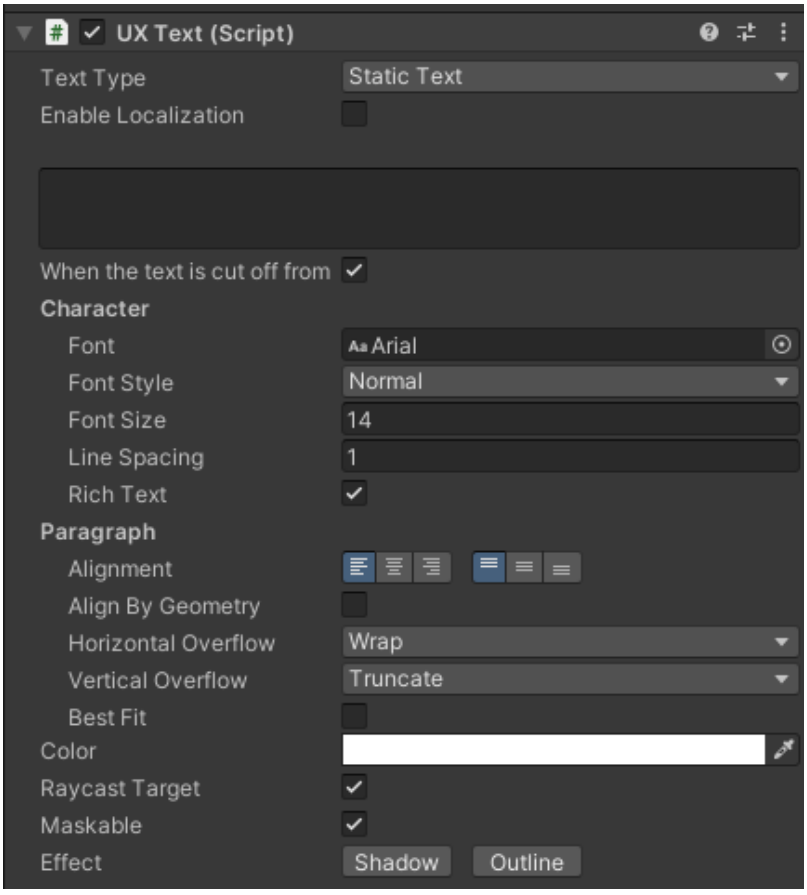
Quick background image in Prefab

The other operations are the same as the quick background image in the scene, but different Prefab can choose different background images and they will be recorded after saving.



UXText

ThunderFire UX Tool extends Unity's original Text in UXText, adding new features to help interaction designers work more easily with text content and providing a partial interface for engineers to change in code.



How to Use

Hierarchy window Click [Right click->UI->UXText] in the Hierarchy window to create a UXText.

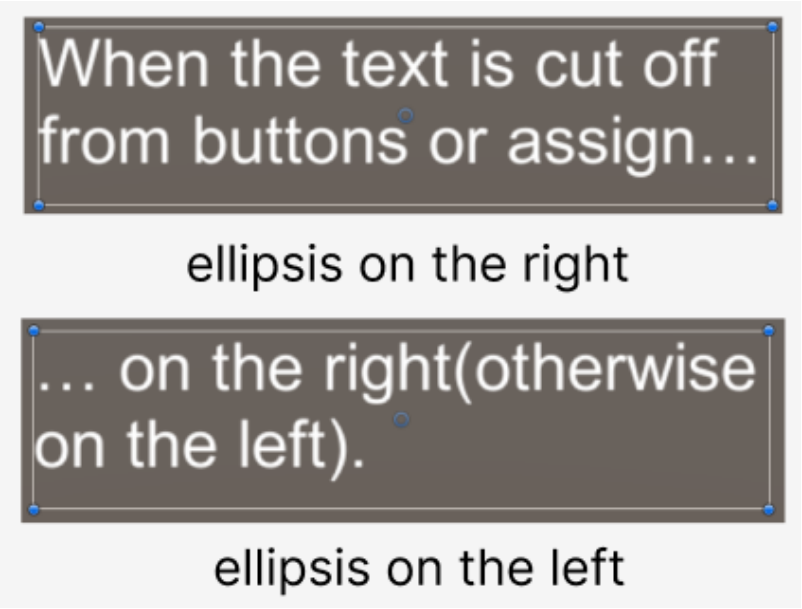
Menu bar Click [GameObject->UI->UXText] in the menu bar to create a UXText as well.

Text Localization

UXText supports text localization function, refer to the **Localization feature document** for details.

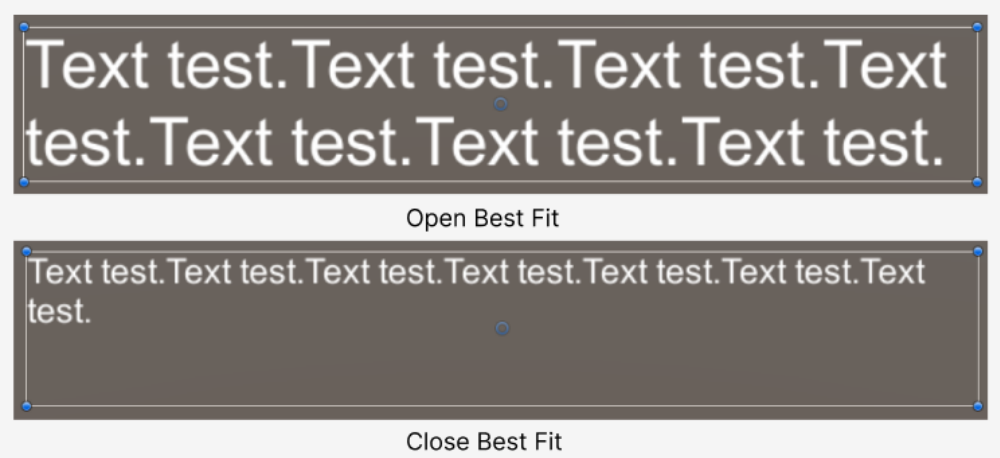
Ellipsis

When the text content exceeds the size of the checkbox, the ellipsis will be displayed. You can set the ellipsis on the left or right side by checking the box.



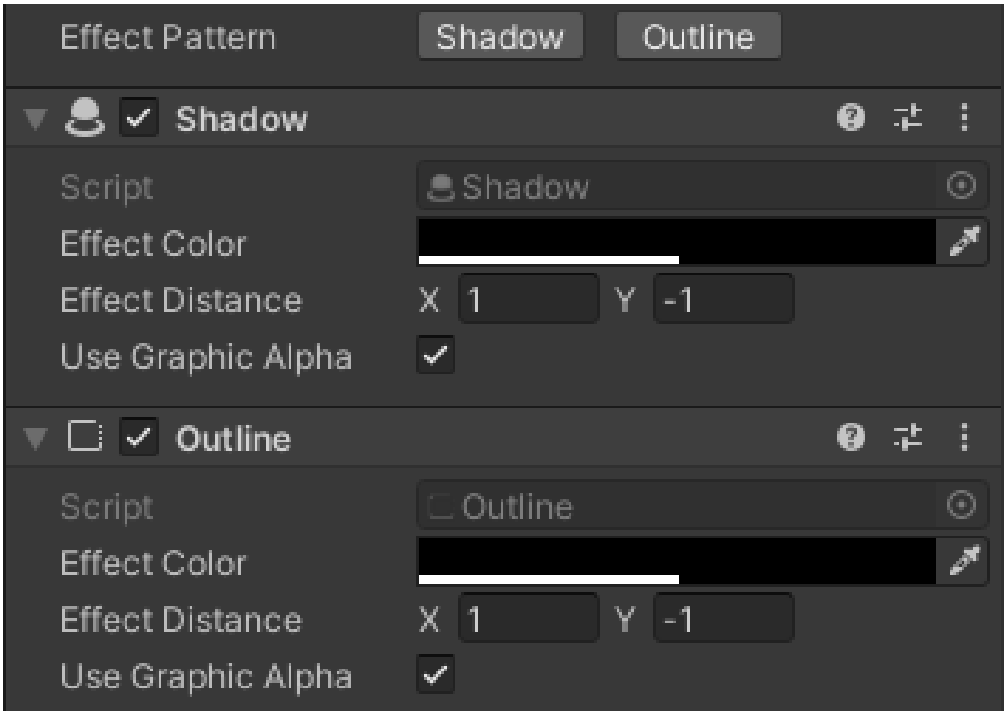
BestFit Adaptive Text Box

Enabling the Adaptive Text Box feature will can adjust the size of the text box to fully accommodate its contents. If this option is disabled, priority will be given to filling the horizontal space until the minimum text size is reached.



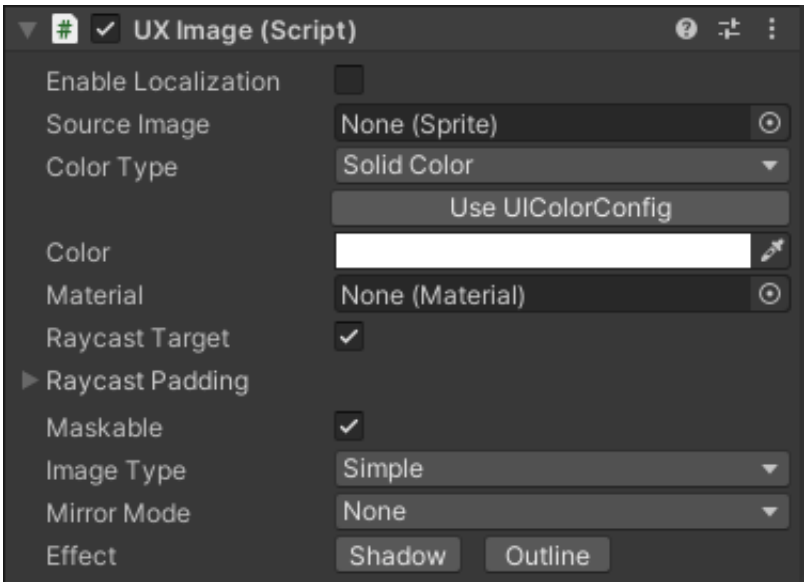
Effect

You can quick-add Shadow and Outline components to style the texts by clicking the Shadow and Outline buttons.



UXImage

ThunderFire UX Tool extends Unity's original Image in UXImage, adding new features to help interaction designers work more easily with image content and providing a partial interface for engineers to change in code.



Use path

Hierarchy window Click [Right click->UI->UXImage] in the Hierarchy window to create a UXImage.

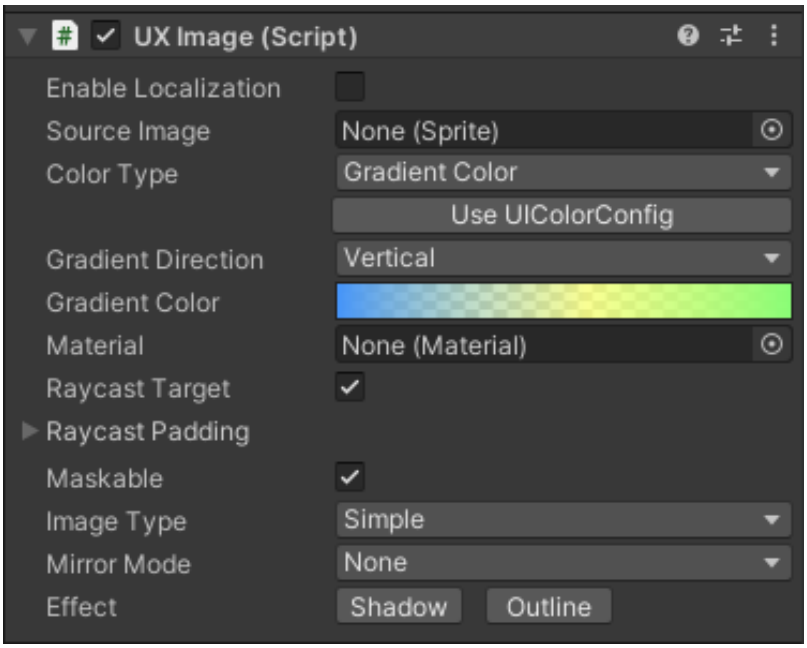
Menu bar Click [GameObject->UI->UXImage] in the menu bar, you can also create a UXImage.

Image Localization

UXImage supports image localization and can switch the image display according to the selected language, refer to **Localization Features** for details.。

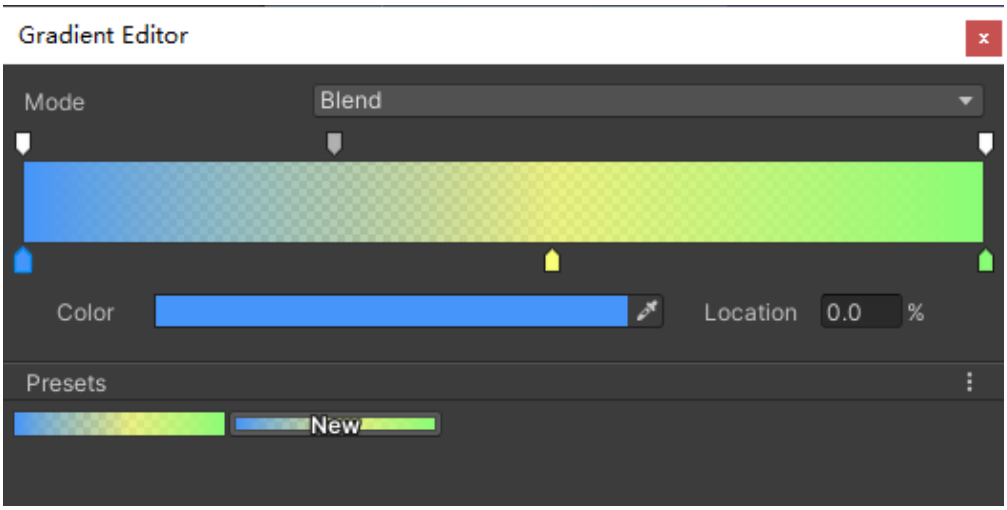
Gradient color

UXImage supports filling the Image with a gradient color selected by the color type drop-down box, as shown in the figure.



Gradient direction the gradient fill direction, you can choose horizontal or vertical gradient.

Gradient color the gradient preview box, click it to open the gradient editor.



In the gradient editor, the effect of gradient can be adjusted.

Gradient mode there are two modes, Blend and Fixed, Blend is a normal gradient, Fixed is a fixed color without transition effect.

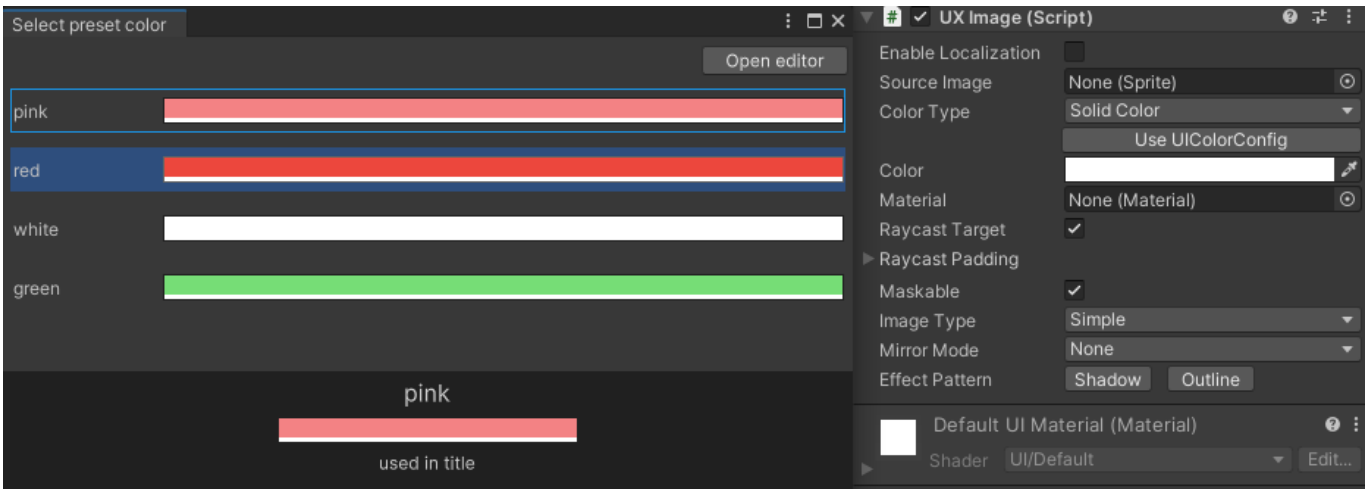
Color scale located at the top and bottom of the gradient color bar, used to indicate the key color used in the gradient, every two adjacent color scale between the smooth and uniform composition of the gradient. Click the upper and lower area of the color bar to add a new color marker, select and press delete to delete the color marker.

- **Color color scale**: located below the gradient bar, with two properties: color and color scale position.
- **Transparency color marker**: located above the gradient bar, with two properties of transparency and color marker position.

Gradient preset you can store the current gradient in the preset.

Using color/gradient configuration

Click the [Use UIColorConfig] button to open the preset selection interface of UIColorConfig and use the configured color or gradient directly, you can refer to the **Color and Gradient Configuration** for details.

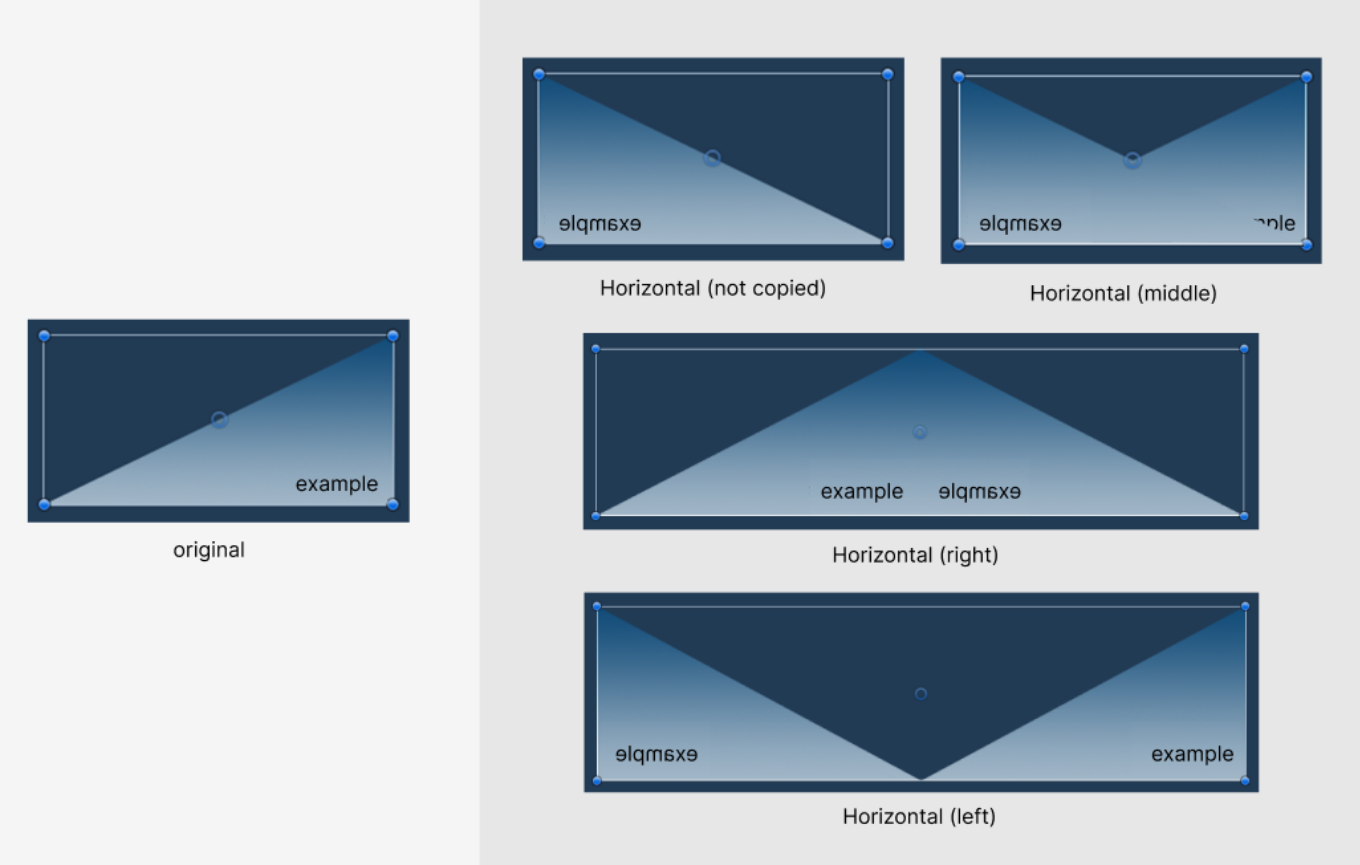


Mirroring

UXImage provides four mirroring modes, namely no mirroring, horizontal mirroring, vertical mirroring and surrounding mirroring, which can help save image resources by using mirroring.

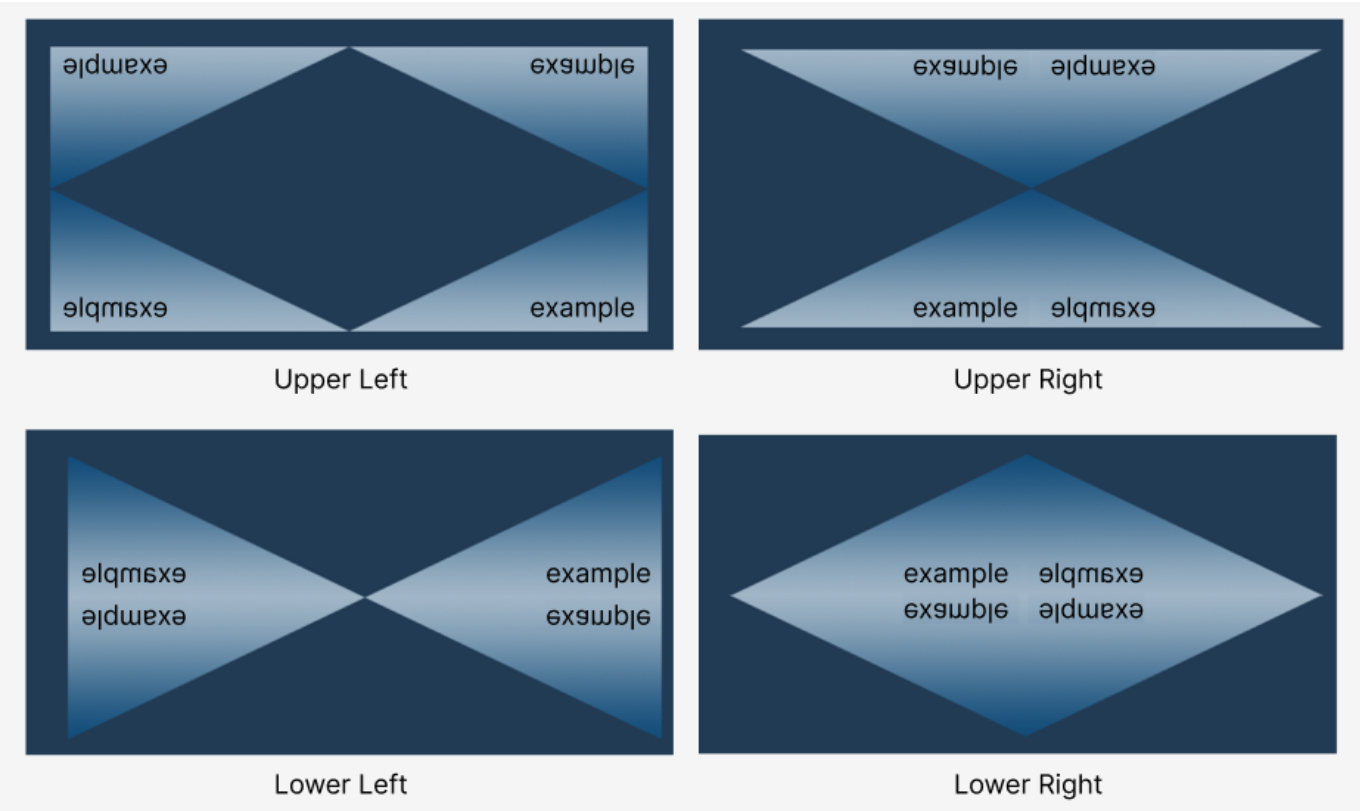
Horizontal Mirroring and Vertical Mirroring

Horizontal mirroring and vertical mirroring flip the original image in the horizontal and vertical directions respectively, and you can choose whether to copy the original image or not, and you can also choose the copying direction when copying. Take horizontal mirroring as an example, the effect of mirroring in different directions is shown in the following figure.



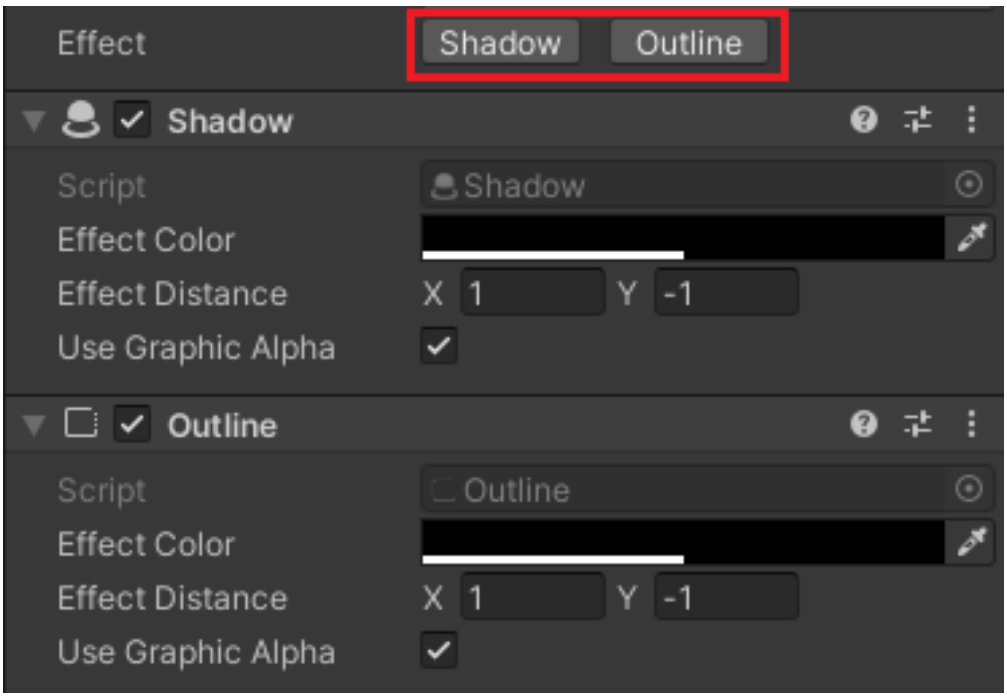
Surrounding Mirroring

Surrounding mirroring is a combination of horizontal mirroring and vertical mirroring, the effect is equivalent to the first horizontal mirroring and copy and then vertical mirroring. the center point of filling can be chosen, as shown in the following figure.



Effect Styles

You can quick-add Shadow and Outline components to style the images by clicking the Shadow and Outline buttons.



UXScrollView

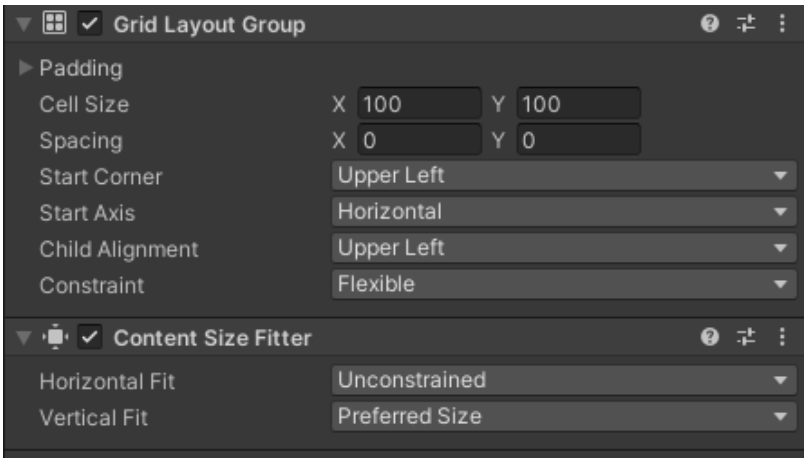
ThunderFire UX Tool extends Unity's original ScrollView in the UXScrollView. Compared to ScrollView, UXScrollView mainly provides **grid element preview** and **auto add component**.

How to Use

Hierarchy window Click [Right click->UI->UXScrollView] in the hierarchy window to create a UXScrollView.

Menu bar Click [GameObject->UI->UXScrollView] in the menu bar to create a UXScrollView as well.

After UXScrollView is created, two components are automatically created on the Content object: Grid Layout Group and Content Size Fitter, and users can modify the parameters according to their actual needs.



Grid Element Preview

In UXScrollView, in addition to the original properties of ScrollView, the following properties are added.

Grid element the preview prefab that is populated in the Viewpoint.

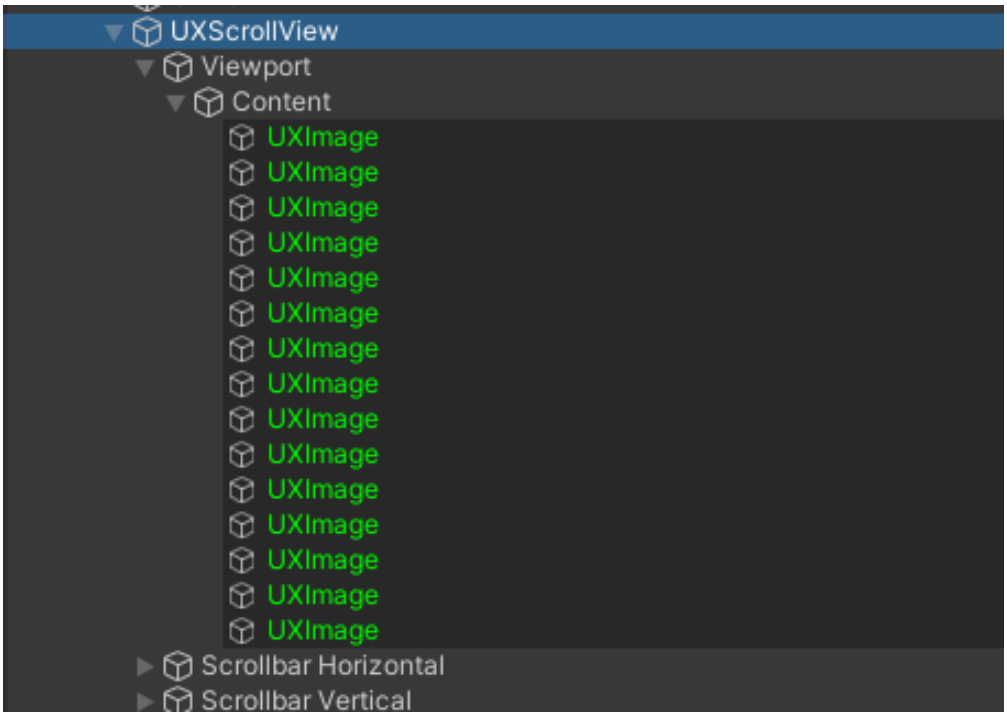
Grid count the number of grids to be previewed, default is 0, i.e. no preview is enabled.

Grid layout the arrangement of the grids in Viewpoint, including grid layout, horizontal layout and vertical layout.

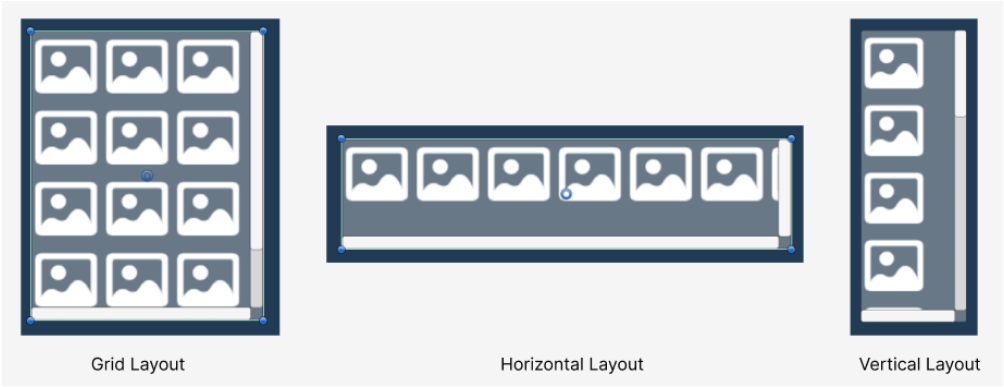
Preview effect

You can preview the effect under Hierarchy and Scene by modifying the grids.

In Hierarchy



In Scene



Color and Gradient Configuration

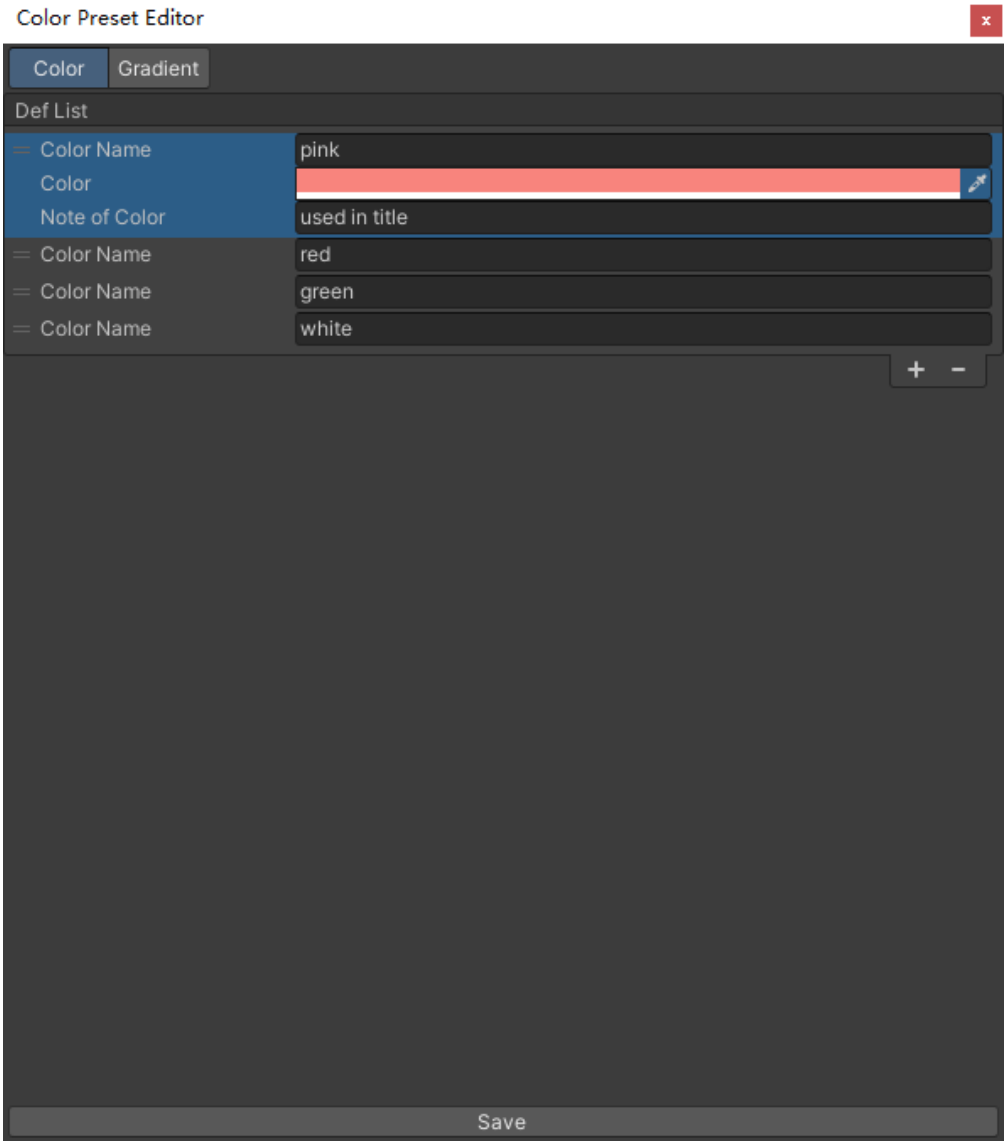
ThunderFire UX Tool offers a convenient solution for configuring colors and gradients, allowing users to apply the preset directly in the Inspector panel or through code.

How to Use

Menu bar By selecting [ThunderFireUXTool -> Create Assets -> UIColorAsset or UIGradientAsset] from the menu bar, you can generate a new configuration file for colors or gradients. Afterwards, click on [ThunderFireUXTool-> UIColorConfig], which will lead you to the following configuration interface (using color as an example; gradient follows suit).

Select a column to expand the detailed configuration, you can configure the color or change the name and notes and other information. Click the "+" or "-" sign to add or delete colors.

To access the generated code file and invoke the color configuration in your code, you can save the data to UIColorGenDef.cs (click on the Save button below). The gradient configuration will be automatically created in UIGradientGenDef.cs file. (Please refer to the program interface below for details on file format.)

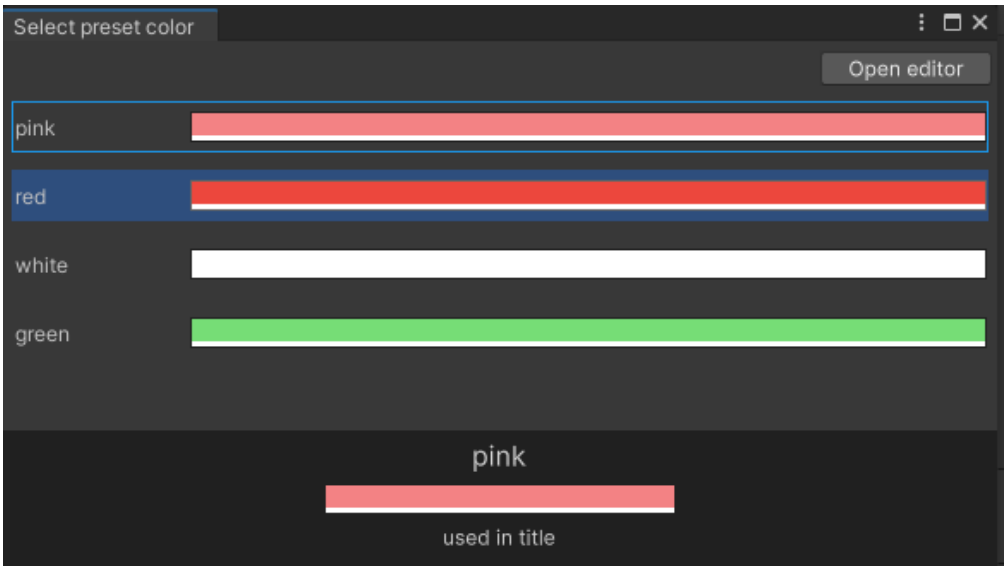


Use Configuration in UXImage Component

Clicking on the UIColorConfig button in UXImage directs you to the following interface.

The window will list all configured colors. Double-click a color entry to select that color as the color used by UXImage. Click the button in the upper right corner to open the Preset Editor to edit the color.

When the color type of UXImage is set to Gradient, it opens the Gradient configuration, which is basically the same as the operation related to the Color Configuration window.



Program Interface

CS files

```
// The following files are automatically generated by the program and can be
called by the user

// File path： Assets/UXTools/Runtime/Feature/UIColor/UIColorGenDef.cs
public sealed class UIColorGenDef {
    public enum UIColorConfigDef {
        Def_Color1 = 0,
        Def_Color2 = 1330857165,
        Def_Color3 = 888517903,
    }
}

// File path： Assets/UXTools/Runtime/Feature/UIColor/UIGradientGenDef.cs
public sealed class UIGradientGenDef {
    public enum UIGradientConfigDef {
        Def_Gradient1 = -1179191585,
        Def_Gradient2 = 549431141,
    }
}
```

Function

```
// Description: Load the generated cs file into the project and call this function
first when using the other functions below
// Class: UIColorUtils
// Parameters: None
// Return value: None
public static void LoadGamePlayerConfig()
```

```
// Description: Get the configured color Color
// Class: UIColorUtils
// Parameters：
//     def the enumeration value in the enumeration of Color automatically
generated in the above cs file
// Return value:
//     The configured color, type Color
public static Color GetDefColor(UIColorGenDef.UIColorConfigDef def)

// Description: Get the configured gradient Gradient
// Class: UIColorUtils
// Parameters：
//     def the enumeration value in the enumeration of the Gradient automatically
generated in the above cs file
// Return value:
//     The configured gradient, type Gradient
public static Gradient GetDefGradient(UIGradientGenDef.UIGradientConfigDef def)
```

Hierarchy Manage

The Hierarchy Manage in ThunderFire UX Tool is designed to visually manage the hierarchy of Prefabs in a project. There are certain problems with the current hierarchy management system, which include:

- Extremely complex interface hierarchies when the number of interfaces is too large.
- Difficulties in communication between management personnel.

This tool hopes to solve the above problems through visual expression, thereby reducing management costs. Based on this expectation, the tool mainly includes the following functions.

How to use

Open via menu Select [ThunderFireUXTool->Hierarchy Manage] in the menu.

Hierarchy Panel Structure

As shown in the figure below, the Hierarchy Manage is structured into 3 layers: **the first-level channels** represented by "HUD, Interface, and Pop-up," **the second-level hierarchy** represented by numbers, and **the third-level Prefabs** represented by gray squares. The number of channels and the number of hierarchies within each channel will be controlled by the program code (see the **program interface** below for reference).

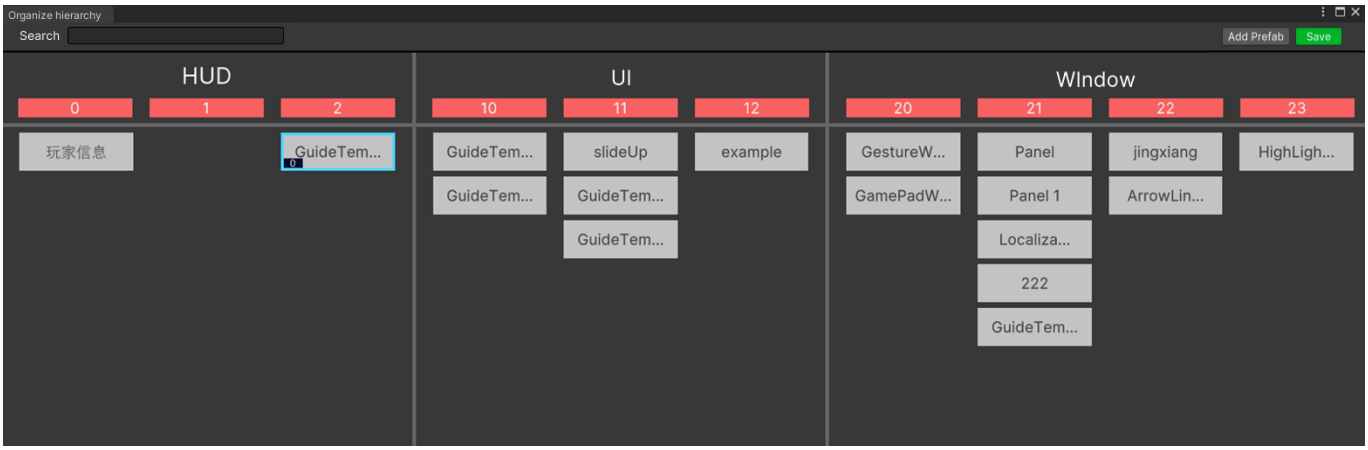
The first level channels divide all hierarchies into major categories, with the number of hierarchies per category set by the project, defaulting to three: "HUD," "Interface," and "Pop-up." If adjustments are needed, the program development team can modify the code.

The second level hierarchies indicate the hierarchical rendering relationship at play mode, the larger the number the closer to the top (the project can be redefined), by default a channel has 10 layers.

The third level prefabs corresponds to the Prefab in the project, and is rendered based on their hierarchy. Among Prefabs in the same hierarchy, those loaded later are closer to the top.

Prefab tab information The tabs in Prefab are intended to keep track of the associated Prefab for easy management later. After the tags are added, fixed data is generated indicating the position of the current Prefab among all Prefabs with the same tag.**Different tags can have different colors**, which are configured in the HierarchyManagementSetting.asset file. If this is the first time a tag is added, it will default to black.

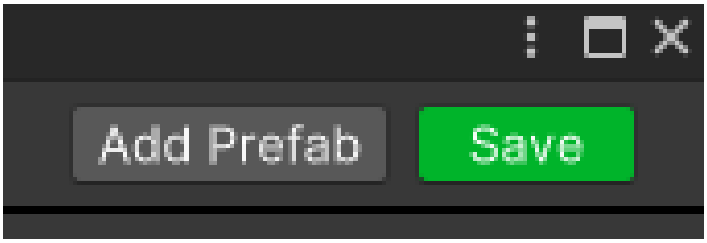
Note: This data **does not change with the change of Prefab hierarchy**. This is to remind users that tag order is predetermined and cannot be changed casually, so as to correspondingly change the Prefabs that come before or after this Prefab.

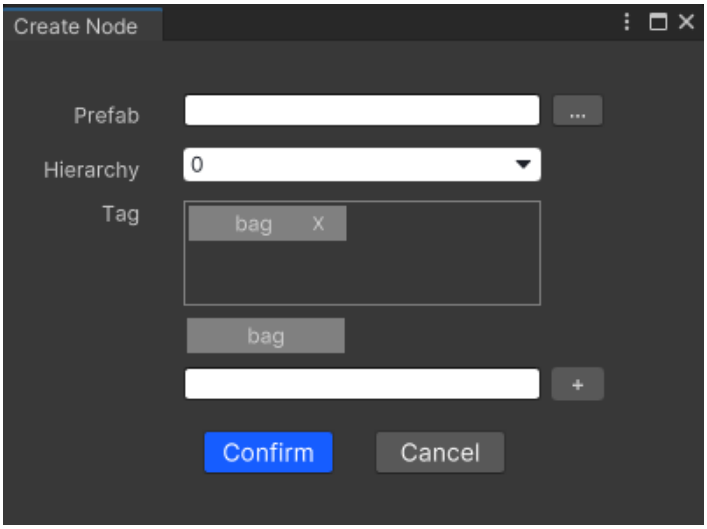


Interface functions

Add Prefab

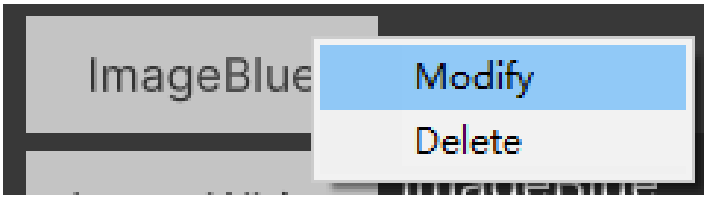
Click "Add Prefab" in the top right corner of the window to bring up the Add window. You can select a prefab by path and choose an existing hierarchy to place it in and click "Save" to save.





Modify And Delete Prefabs

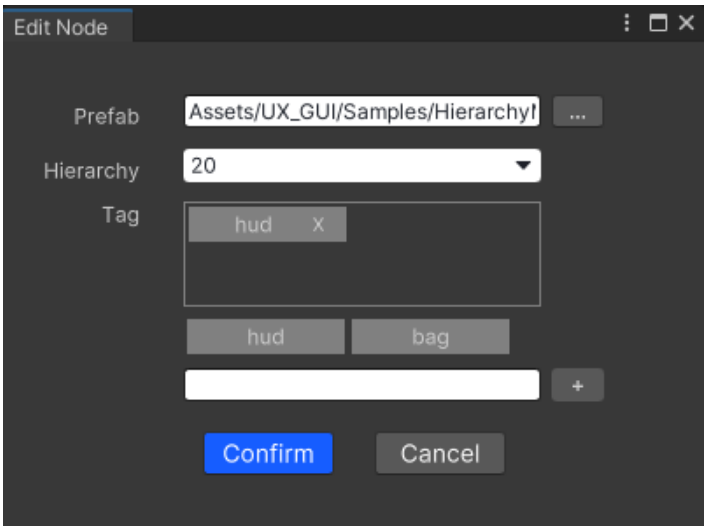
- Modify: Right-click on the Prefab and select Modify to bring up a modification window similar to Add.
- Delete: Right click on the Prefab and select Delete to delete the Prefab from the management window.



Add And Delete Tags

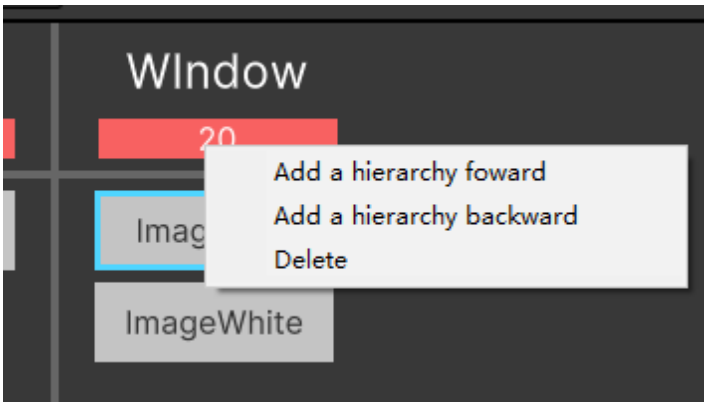
- Add: In the modify or add window, there is a tag editing section. Enter the desired text in the text box and click the "+" button. The new tag will be displayed in the upper box area, and the two most recently added tags will be displayed in the middle area. Clicking on either of these will add them directly to the tag list.
- Delete: Click "×" in the box line area to delete it.

Note: Remember to click "OK" after making any changes to save them.



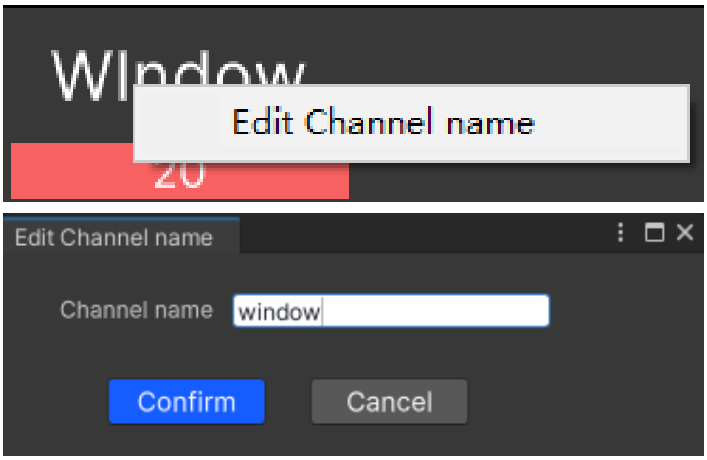
Add And Delete Hierarchies

Right-click on a hierarchy to add a new hierarchy in front of or behind it. You cannot add more hierarchies than the maximum allowed for that channel, and you will receive a prompt if you try. Click "Delete" to **remove a hierarchy that does not contain a Prefab**. You cannot delete the last hierarchy in a channel if it contains only one hierarchy.



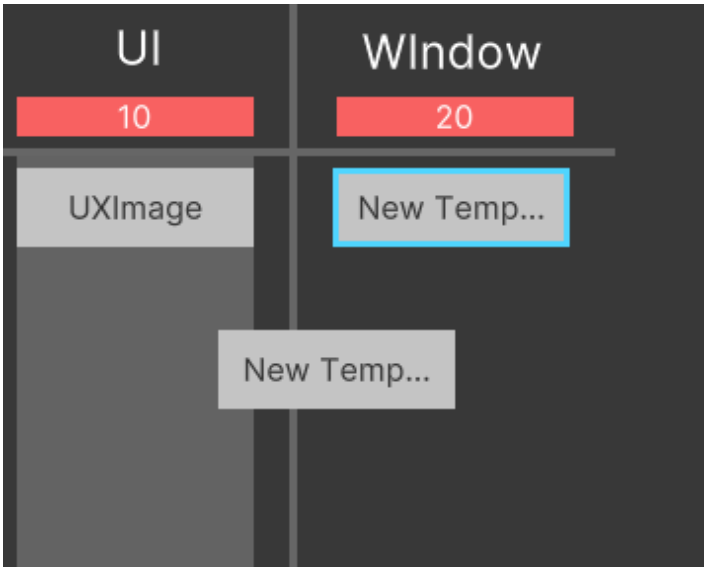
Edit Channel Name

[Right-click] the channel name, select "Edit Channel Name", and then modify it in the pop-up window.



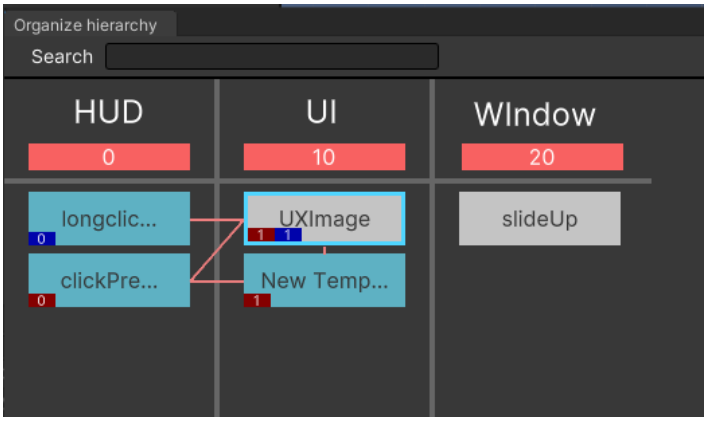
Drag The Prefab

After selecting a Prefab, you can drag it to move it to a different hierarchy. The hierarchy you're dragging it to will be highlighted, and you can release the mouse button to complete the move.



Prefab Relationship Indication

When you select a Prefab, it will be outlined in blue, and the related Prefabs will be highlighted and connected in sequence according to their tag values. (**Related** means that they have the same tags as the selected Prefab or they have the same Prefab as other Prefabs related to the selected Prefab by having the same tags, this can be thought of as a grid expanding).

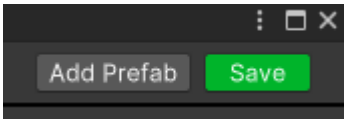


Quick search (Fuzzy Search)

Enter the search content in the search box (only English search is supported in the 2019 version). If there are search results, the nearest (forward) Prefab to the currently selected Prefab will be selected and highlighted. The search box will display "Current Number/Total Number" or "No Results". Click "<" or ">" to select a Prefab forward or backward, and automatically switch the view to the vicinity of the selected Prefab.

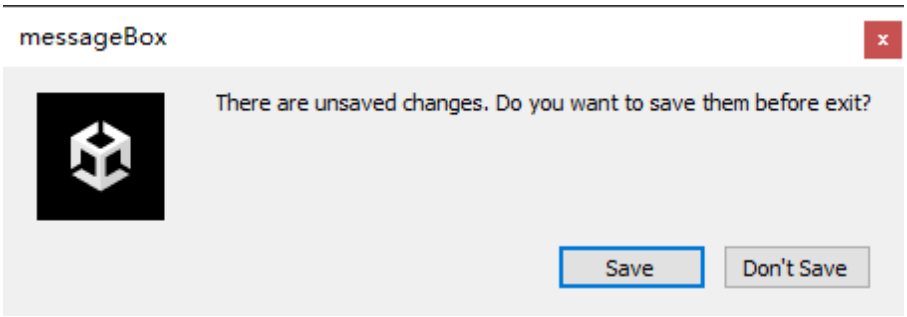
Save Information

Click Save in the upper right corner to save the tag information, Prefab information, channel name, hierarchy level, and other modifiable information on the interface.



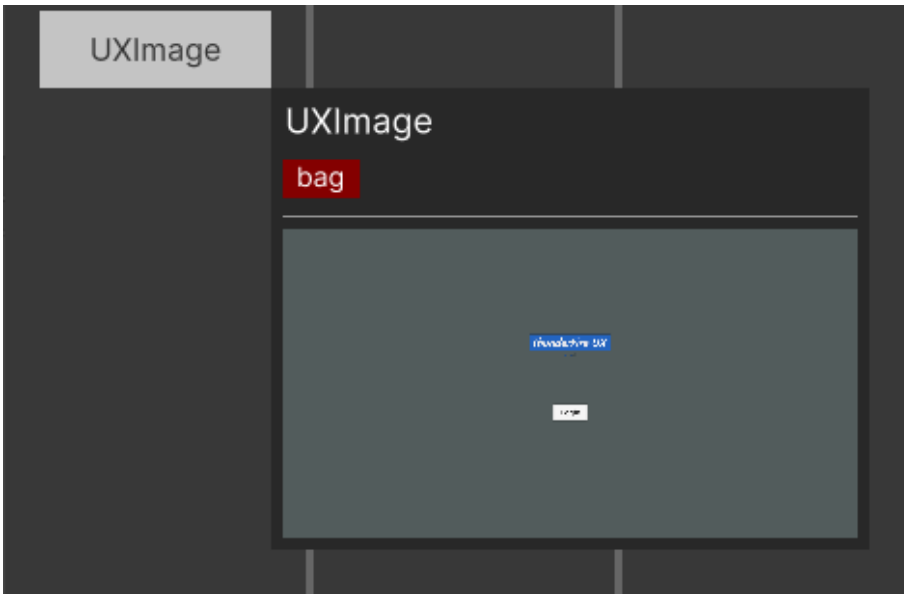
If changes have been made on the page but not saved before closing, a prompt box will pop up.

Note: If you directly click the "×" in the upper right corner of the pop-up window, the changes will not be saved and the management tool will be closed.



Prefab Preview

When the mouse moves over a Prefab, a tooltip will display the name, tabs, and thumbnail images of the Prefab.



Program Interface

Variables

```
// delegate with string as reference and int as return value (used to describe the
method of getting index value by guid)
// Class：HierarchyManagementOutSetting
public delegate int GetIndex(string guid);
```

Functions

```
// Description: Serialize the initial value of the Hierarchy Management tool
// Class：HierarchyManagementOutSetting
// Parameters：
//     guids：list of guids to be serialized
//     getIndex：a delegate to get the index value by guid
// Return value: none
public void CreateGuidList(List<string> guids, GetIndex getIndex)

// Description: Set the number of channels, the number of levels of each channel
// Class：HierarchyManagementSetting
// Parameters：
//     levelRange：number of levels on each channel
//     maxChannelNum：aximum number of channels
// Return value: none
public void SetInitChannelAndLevel(int levelRange,int maxChannelNum)

// Description: Set the callback after saving data
// Class：HierarchyManagementSetting
// Parameters：
//     action：a method with the parameters List<GuidWithIndex>, the list
```

```
    parameter represents the saved data
    // Return value: none
    public void SetAfterSubmit(Action<List<GuidWithIndex>> action)
```

Others

```
// Description: For a Prefab storage method
// Class: HierarchyManagementSetting
public class GuidWithIndex
{
    // Prefab name
    public string Name;
    // Prefab in the hierarchy of the serial number
    public int Index;
    // Prefab's Guid
    public string Guid;
    // Prefab contains the tags
    public List<TagDetail> Tags = new List<TagDetail>();
}

// Description: The properties of the tags
// Class: HierarchyManagementSetting
public class TagDetail
{
    // Tag name
    public string Name;
    // Tag serial number
    public int Num;
}
```

Beginner Guide

ThunderFire UX Tool provides a complete set of Beginner Guide features, including onboarding editing tools and play mode feature, to help UI designers quickly style and adjust the Beginner Guide interface in the Unity editor and allow engineers to quickly integrate onboarding into their projects.

Terminology

GuideWidget A guide control that represents only one kind of guide effect, such as Hollow Highlightings, Gestures, Guide Text, etc.

GuideWidgetData Records various data in the GuideWidget. Used to save data for the guide interface edited in the editing tool.

UIBeginnerGuide A template for a guide interface that includes multiple different GuideWidgets.

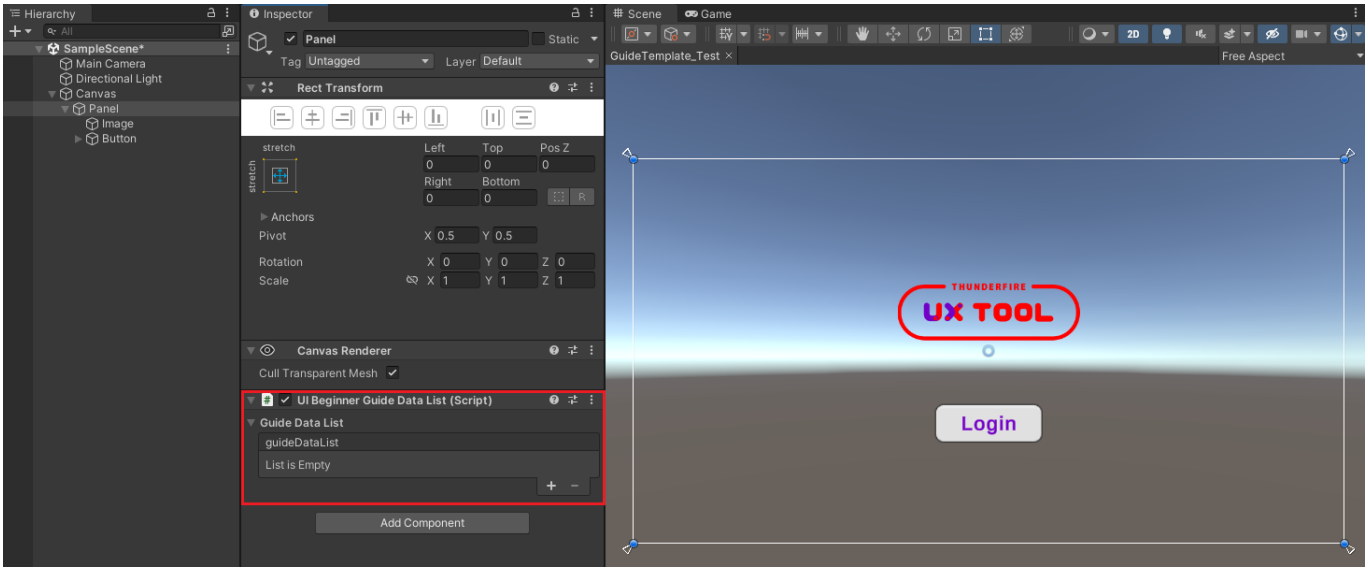
UIBeginnerGuideData Records various data in the UIBeginnerGuide. Includes Guide ID, Guide Type, Guide Duration, Guide Template, and GuideWidgetData included in the UIBeginnerGuide (see **Parameter Details** below).

UIBeginnerGuideDataList A guide list that includes a group of UIBeginnerGuideData. After one of the guides in the list is completed, the next guide in the list will be automatically activated until the entire guide list is completed.

How to use

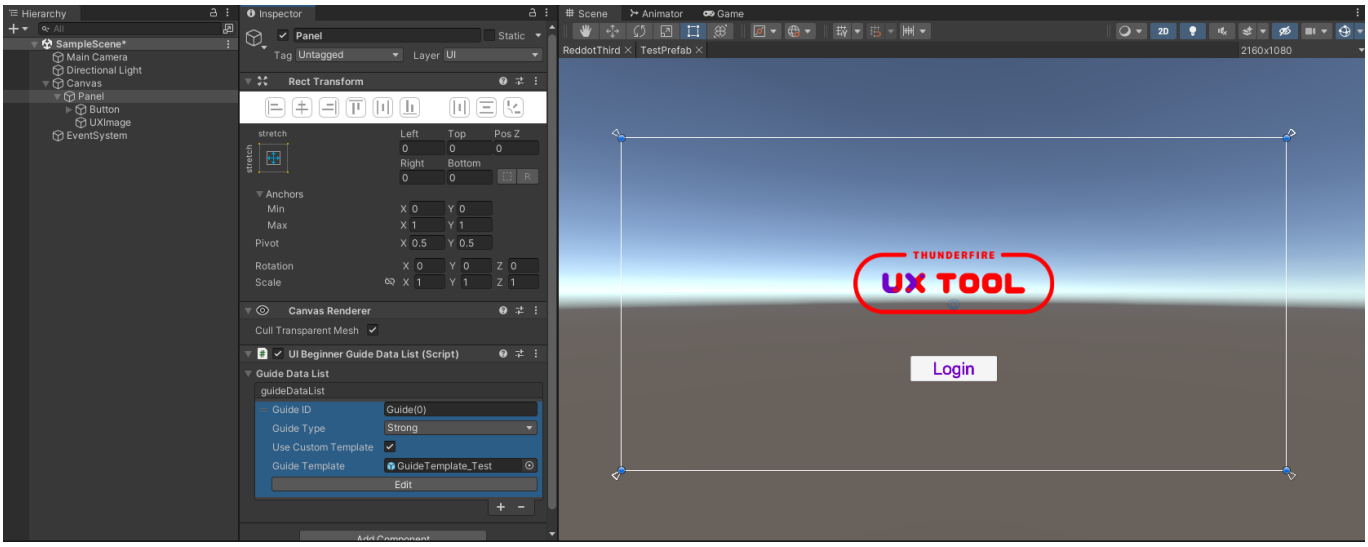
1.Create The Guide Data List

Select a node in the Ccene or Prefab (usually a persist object), Click [ThunderFireUXTool-Create-Create BeginnerGuide] in the top menu bar. This will add a UIBeginnerGuideDataList component to the object, which represents a sequence of guide steps. Alternatively, you can manually add this component via the Inspector panel.



2.Create New Guide Steps

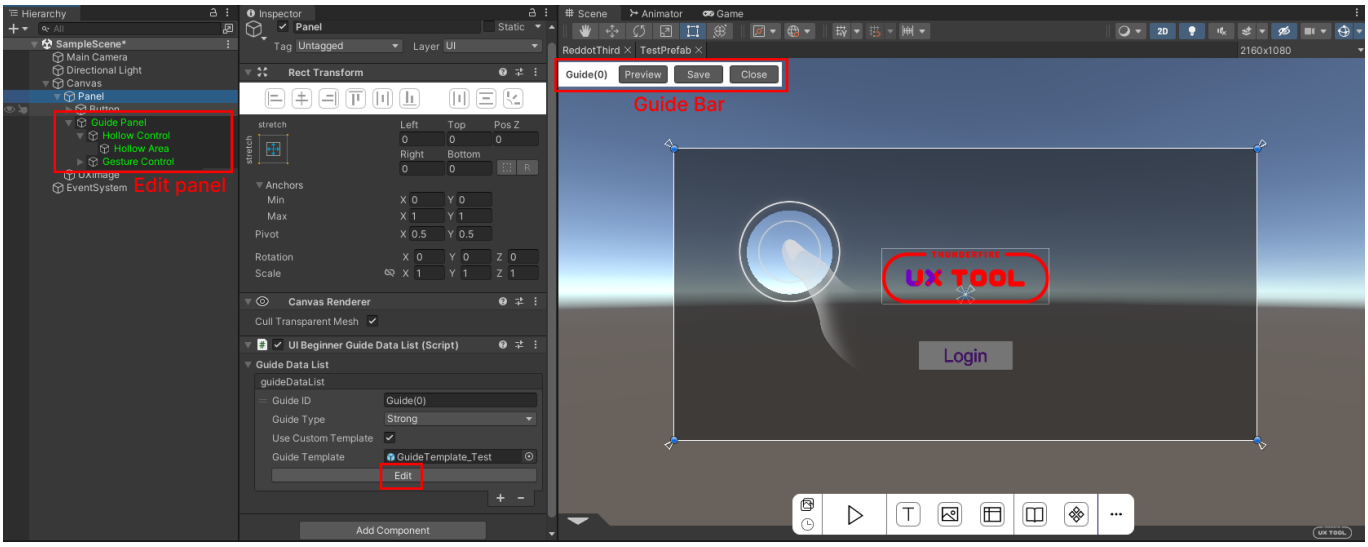
The Guide Data List is empty by default. To add a new guide step, click the "+" button in the UIBeginnerGuideDataList component. To configure the guide step, you can use a preset template or a custom template. Here, we will use a custom template. (You can refer to the section **"Creating a Custom Guide Template"** below.)



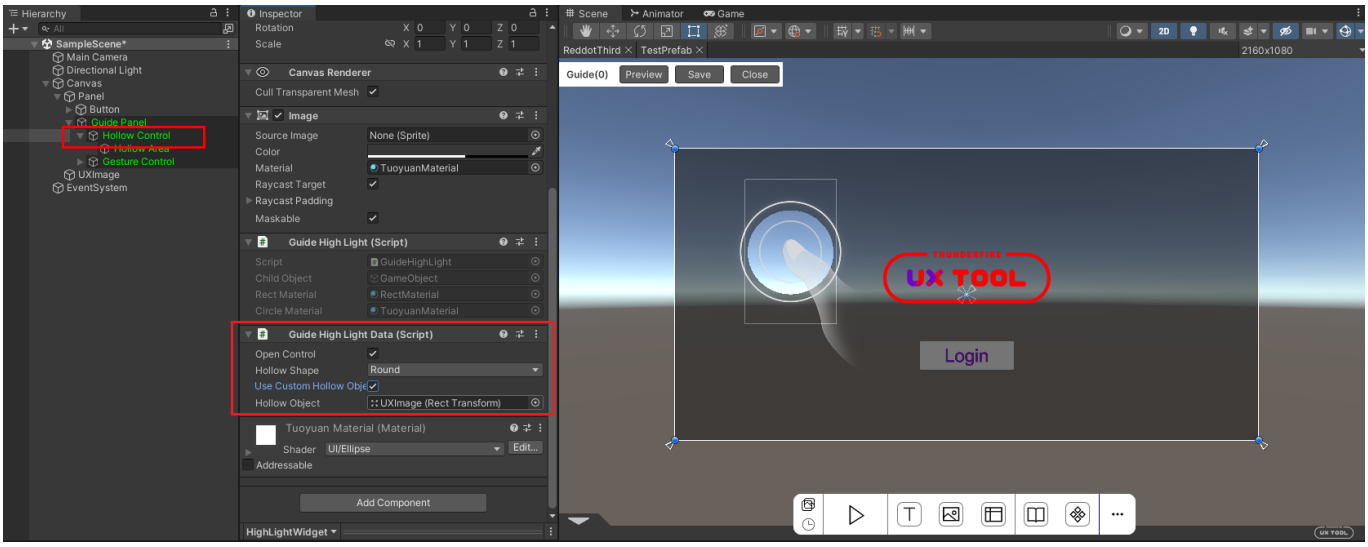
3.Edit GuideData List

Click the [Edit] button to create a guide editing panel in the Hierarchy window. The objects that can be edited will be displayed in green. Additionally, a guide bar will appear in the top left corner of the Scene, displaying the Guide ID, Preview, Save, and Close buttons.

The guide editing panel is a temporary object that is instantiated from the "Guide Template", and is only provided as a visual editing tool. It will be deleted if you switch scenes, start running play mode, or click the close button.

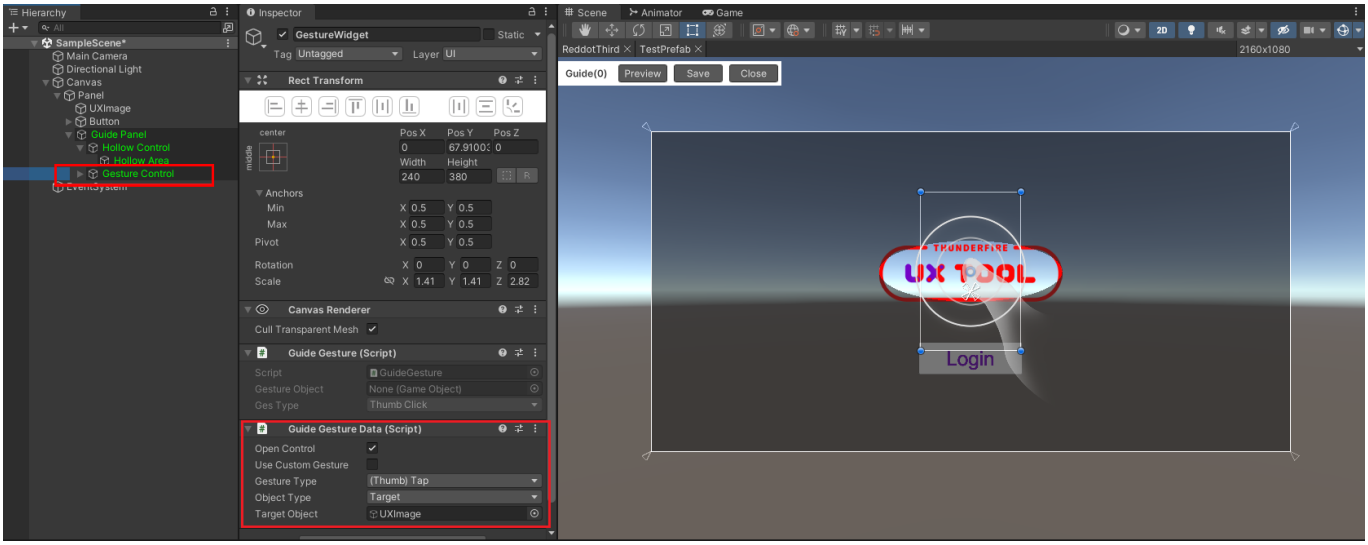


Edit the Hollow object. Click on the "Hollow Component" object in the Hierarchy and configure it in the GuideHighLightData component as shown:



The above steps are to select "Round Hollow" and the hollow object is "UXImage" (i.e. the UX Tool image in the figure). After editing is complete, click the "Save" button in the guide toolbar to save your changes.

To edit the Gesture component, open the guide editing interface and select the "Gesture Component" object. In the GuideGestureData component, configure the settings as shown in the picture:

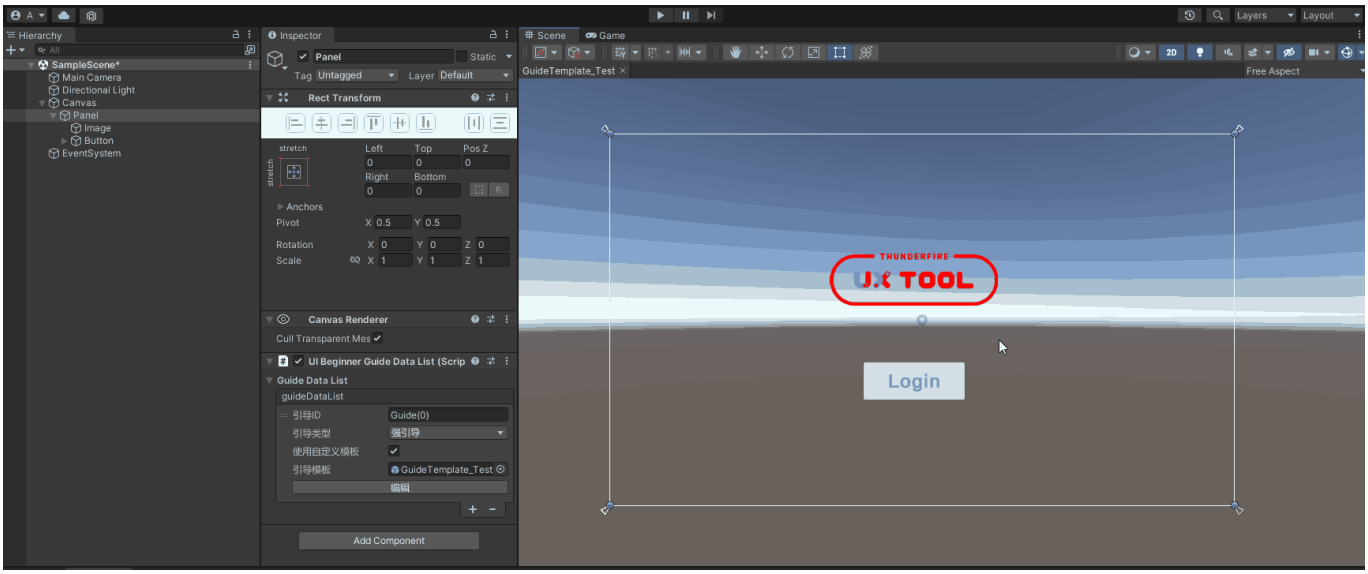


Add a guide animation for a tap gesture on the UXImage (i.e. the UX Tool image in the picture) object. After editing is complete, click the "Save" button in the guide toolbar to save your changes.

4.Preview Guide

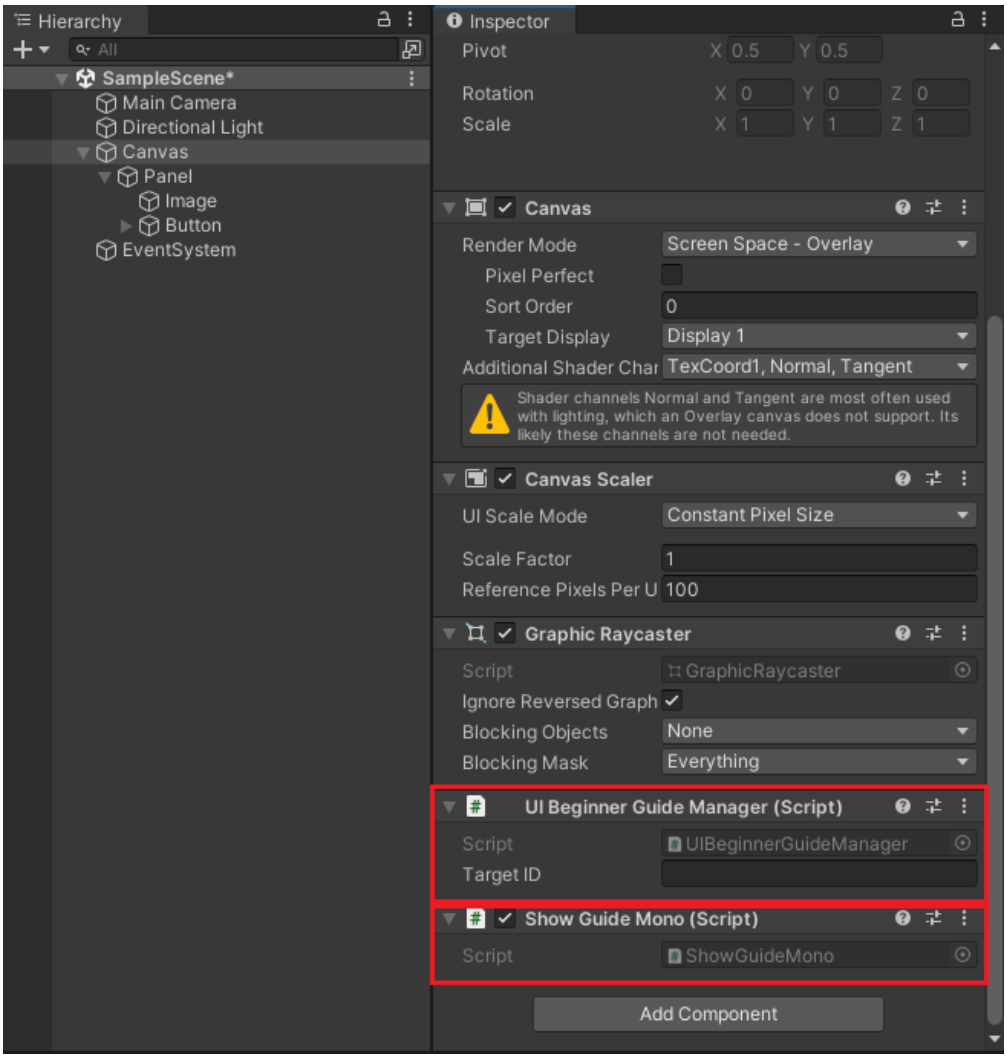
Click the Preivew button in the Toolbar to see the hollow highlighting and gesture animation effect you just edited.

Note: Because the hollow and gesture configurations have just been configured to follow the UX Tool image, the guide effect will also be automatically positioned when the UX Tool image position changes.



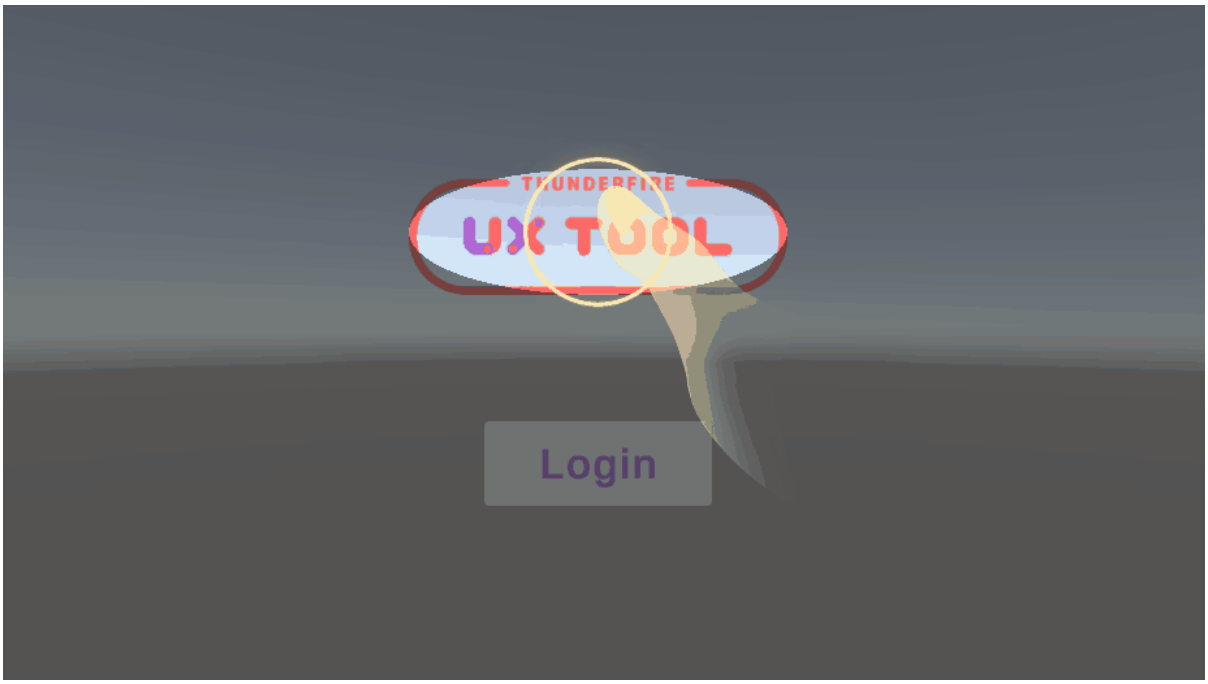
5.Guide At Play Mode

To properly display the guide content at play mode, the scene must include the UIBeginnerGuideManager component. Add the UIBeginnerGuideManager component to the Canvas object in the scene, and all guide interfaces will be loaded under this Canvas. The UIBeginnerGuideManager is a singleton object, so only one is required globally.



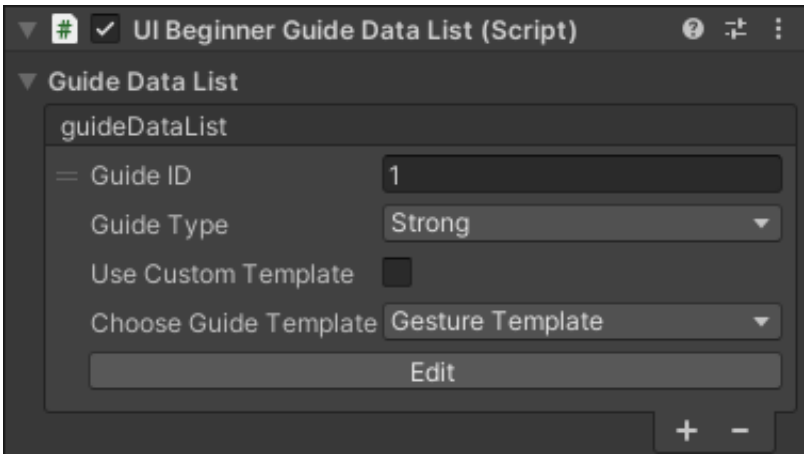
In addition, there should be a program script to control the timing of the Beginner Guide's playback (the specific interface can be seen in the Program Interface below). In the ShowGuideMono script example in the sample, two operations are executed: adding the UIBeginnerGuideList to the Manager and then playing the guide in the Manager.

The guide created by the UIbeginnerDataList component will determine the end condition based on the type of guide, and automatically open the next guide in the group until there are no more guides in the group.



Parameter Details

Guide Parameters

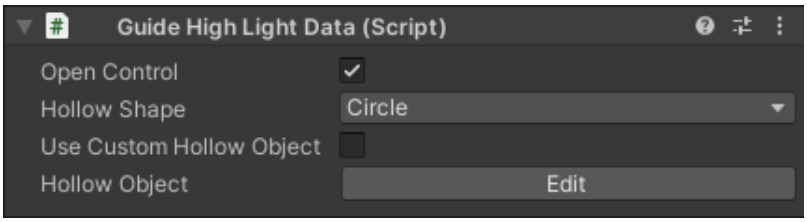


- **Guide ID**: Used to distinguish the names of different guides, generated by default, can be modified according to the content of the guide.
- **Guide Type**: must be used with the skeleton component, and only clicking on the skeleton area will end the guide.
    - Strong Guide: must be used with the skeleton component, and only clicking on the skeleton area will end the guide.
    - Medium Guide: Click anywhere on the screen to end the guide.
    - Weak Guide: Automatically ends guide after a period of time.
- **Guide Duration**: When the guide type is weak, sets the duration of the guide. This setting is not applicable for other guide types.

- **Use Custom Template**: Specifies whether to use a custom guide template. Selecting this option displays the "Guide template" property, while unselecting it displays a dropdown menu for choosing a guide template.
- **Guide Template**: Specify a custom guide template.
- **Choose Guide Template**: Choose from preset guide templates provided by the tool.

Preset Guide Controls

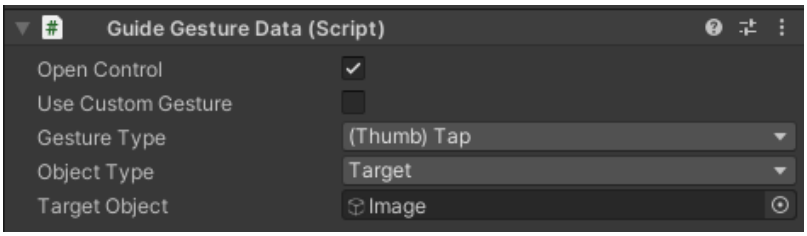
Hollow Contol



Used to add a color mask to a scene or interface and hollow the guided content for display.

- **Open Control**: Control switch, selected by default.
- **Hollow Shape**: You can choose square or round as the shape of the hollow area.
- **Use custom hollow object**: Hollow objects can be used by default or customized.
- **Hollow Area**: Determined by whether to check the use of custom hollow objects.
    - If not selected, the control's built-in hollow area object is used. Click the "Edit" button of the hollow object to directly edit the size and position of the hollow area in the scene.
    - If selected, other UI objects in the scene, such as buttons and images, can be dragged and dropped as hollow objects. The hollow area will automatically recognize the size and position of the object and will be visible at play mode.

Gesture Control



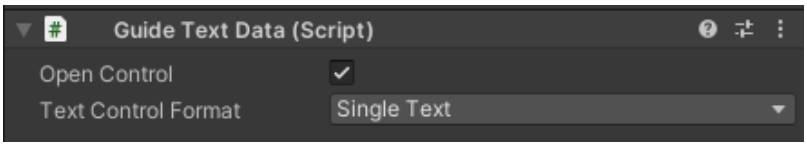
Provide commonly used gesture guide presets with animations and customization options.

- **Open Control**：Control switch, selected by default.
- **Use Custom Gestures**: Gestures can be default or customized.
- **Gesture type**: Determined by whether Use Custom Gesture is selected.

    - If not selected, the default gesture presets are used, and the desired gesture type can be selected directly from the drop-down list.
    - If selected, drag and drop gesture objects made by the project itself.

- **Drag Curve**: This parameter is displayed when the gesture type is drag. At this time, there will be a start marker and an end marker for the drag gesture in the scene, which is used to set the starting point of the drag gesture. At the same time, adjusting the drag curve can adjust the drag animation.
- **Object Type**: The objects that can be guided by gestures can be either customized or targeted. For customized objects, the gesture can be dragged to the desired guide position in the scene. For targeted objects, other objects in the scene can be selected to be the guided object.
- **Target Object**: Show this parameter when the object type is targeted. Other UI objects in the scene can be dragged as the guided object of the gesture. At this time, the gesture will always follow the position of the object.

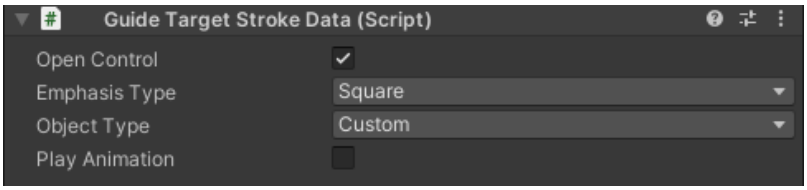
Text Control



Provide basic text objects to be used as guidance tips.

- **Open Control**: Control switch, selected by default.
- **Text Control Format**: Optional text control with or without title, with two options: single text and text with title.

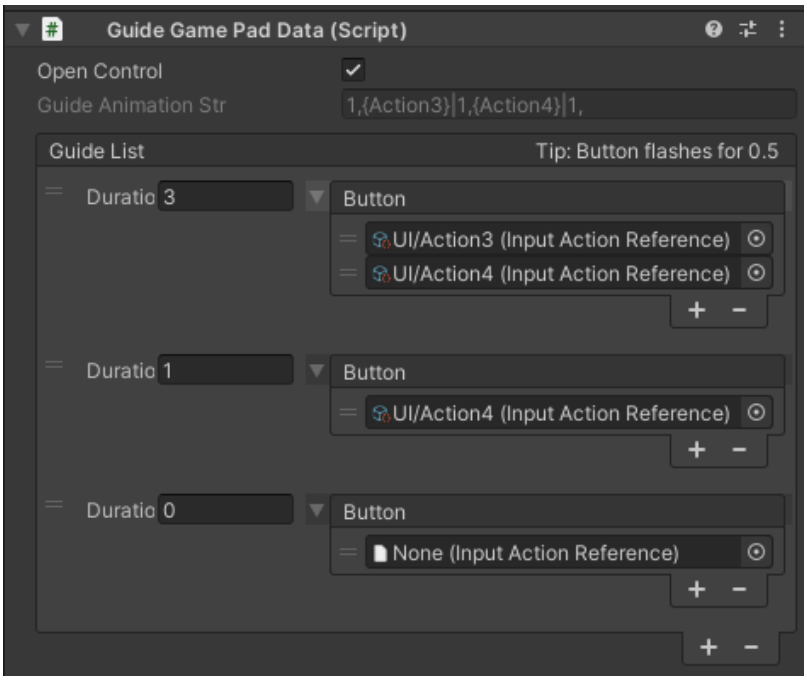
Emphasis Control



Use the emphasis box to frame the lead content for emphasis.

- **Open Control**: Control switch, selected by default.
- **Emphasis Type**: You can choose square or round as the shape of the emphasis box.
- **Object Type**: The object to be emphasized can be customized or targeted. For customized objects, the emphasis box can be dragged to the desired position in the scene. For targeted objects, other objects in the scene can be selected as the emphasized object.
- **Target Object**: This parameter is displayed when the object type is targeted. Other UI objects in the scene can be dragged and selected as the emphasized object. The emphasis box will then always follow the position of the selected object.
- **Play animation**: Whether to play the fade-in and fade-out animation of the emphasis box.

Game Pad Control



Provides game pad guidance.

- **Open Control**: Control switch, selected by default.
- **Guide List**: A sequence of button animation guides. Click the "+" icon to add a new guide, and select an existing guide and click the "-" icon to delete.
    - Duration: The display duration of the guide, which is 0.5 sec for each flash of the button.
    - Button: The button corresponding to the guide. Multiple flashing buttons can be set simultaneously. If no button is set, it will not flash by default.

Preset Guide Templates

When editing the guide, you can select the preset guide templates provided by the tool, including the gesture template and the game pad template, which consists of the above guide controls.

Gesture Templates

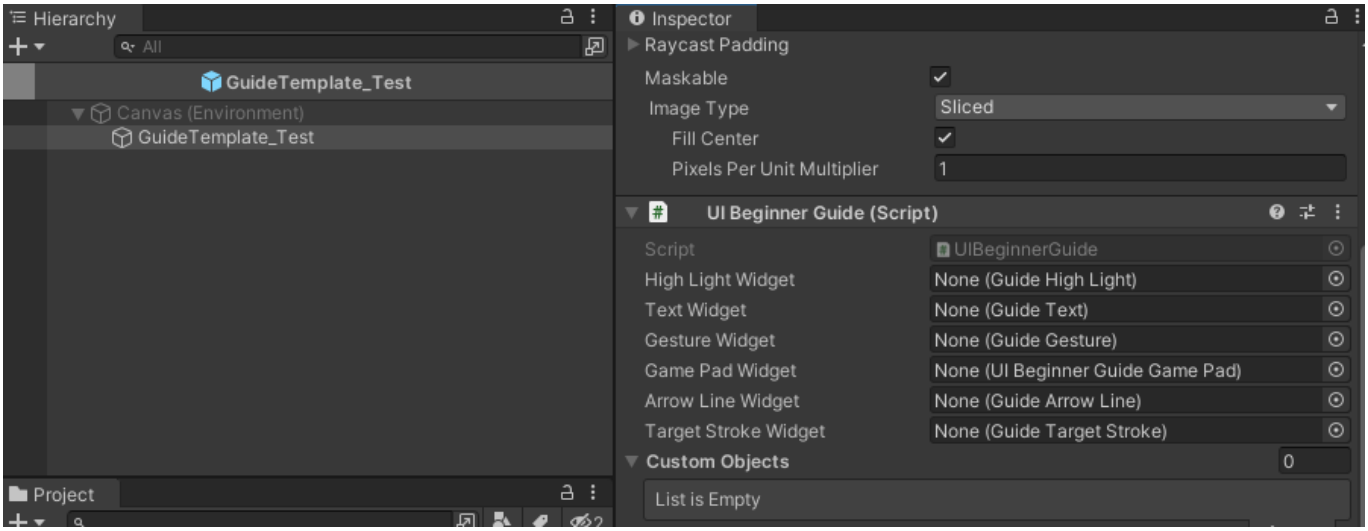
The gesture template contains hollow controls, gesture controls, emphasis controls and text controls. When using the template, you can turn on or off various controls and adjust their parameters as needed, and save them to take effect.

Game Pad Template

The gesture template contains skeleton controls, handle controls, emphasis controls and text controls. When using the template, you can open or close various controls and adjust the parameters in them as needed, which will take effect after saving.

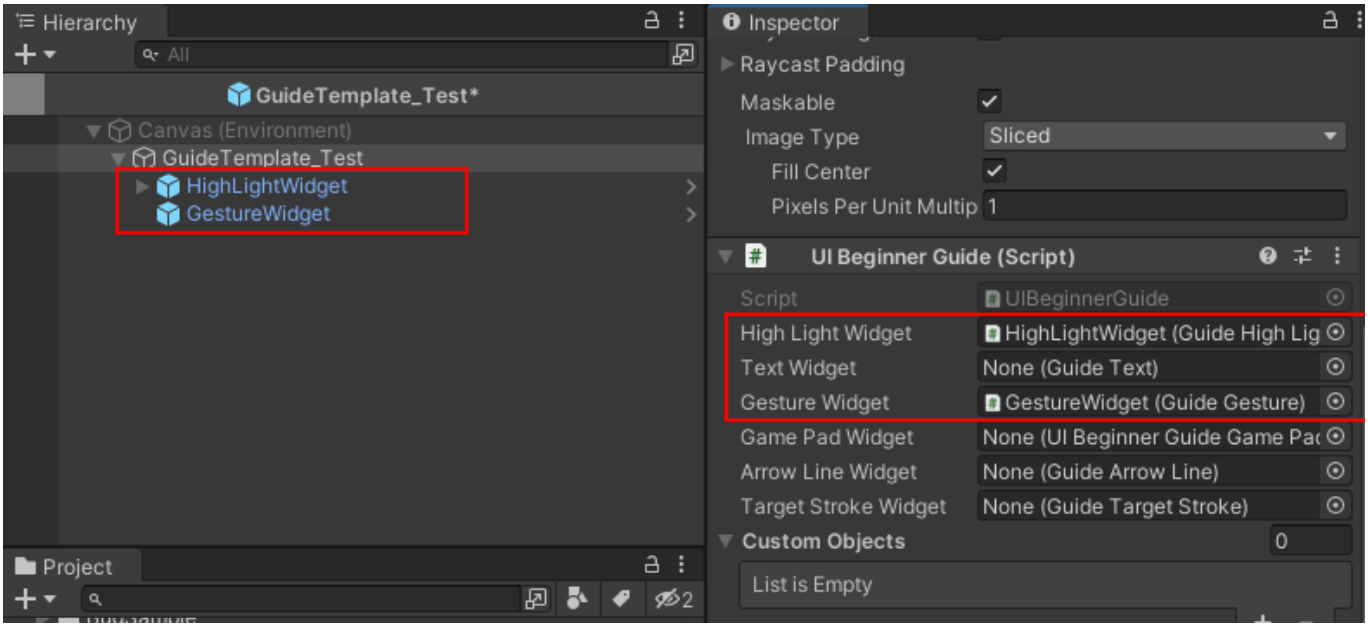
Create Custom Guide Templates

Step 1 Create a new UI Prefab named it GuideTemplate_XXX, XXX can be any name (here it is named GuideTemplate_Test), add UIBeginnerGuide component to the root object of this Prefab.



Step 2 In the folder of preset controls, add the guide controls as child nodes of the Prefab according to the requirements, and bind the corresponding widgets in the UIBeginnerGuide component. Here, we add two common types of guides: Hollow and Gesture.

Default folder path for preset controls: Assets/UX_GUI/UX-GUI-Feature/BeginnerGuide/Res/Prefab/BeginnerGuideWidget



Step 3 Save the Prefab so that a guide template is created and the usage is the same as defining a template.

Program Interface

Variables

```
// instance of UIBeginnerGuideManager
// Class：UIBeginnerGuideManager
public static UIBeginnerGuideManager Instance;
```

Function

```
// To use the following function, you need to get UIBeginnerGuideManager.Instance
and then use it

// Description: Adds a boot to the boot list queue
// Class：UIBeginnerGuideManager
// Parameters：
//     datalist: the boot object to add
// Return value: None
public void AddGuide(UIBeginnerGuideDataList datalist)

// Description: Set the next boot list to start with the boot item whose name is
id
// Class：UIBeginnerGuideManager
// Parameters：
//     id: the id to be played next time the guide is played
// Return value: None
public void SetGuideID(string id)

// Description: Play the first boot list
// Class：UIBeginnerGuideManager
// Parameters: None
// Return value: None
public void ShowGuideList()

// Description: Play the specified list of guides
// Class：UIBeginnerGuideManager
// Parameters:
//     datalist: the boot list object to be played
// Return value: None
public void ShowGuideList(UIBeginnerGuideDataList datalist)

// Description: Play the specified ID from the specified boot listD
// Class：UIBeginnerGuideManager
// Parameters:
//     datalist: he boot list object to be played
//     id: the id to be played next time the guide is played
// Return value: None
public void ShowGuideList(UIBeginnerGuideDataList datalist, string ID)

// Description: Ends the current boot
// Class：UIBeginnerGuideManager
// Parameters: None
// Return value: None
public void FinishGuide()
```

```
// Description: ends the boot of a certain ID (if the current boot is not that ID,
the function is invalid)
// Class：UIBeginnerGuideManager
// Parameters:
//     guideID: the guide ID to end
//     Return value: None
public void FinishGuide(string guideID)

// Description: Clear all boot sequences in the list
// Class：UIBeginnerGuideManager
// Parameters: None
// Return value: None
public void ClearGuide()

// Description: Clearly list the specified boot sequence
// Class：UIBeginnerGuideManager
// Parameters:
//     dataList: the boot sequence to be deleted
// Return value: None
public void ClearGuide(UIBeginnerGuideDataList dataList)
```

Localization

ThunderFire UX Tool's Localization supports two categories: UX Image and UX Text.

- UX Image: localization of UXImage.
- UX Text: including the localization of UXText and UXTextMeshPro, the operation is similar, the screenshot of this manual takes UXText as an example.

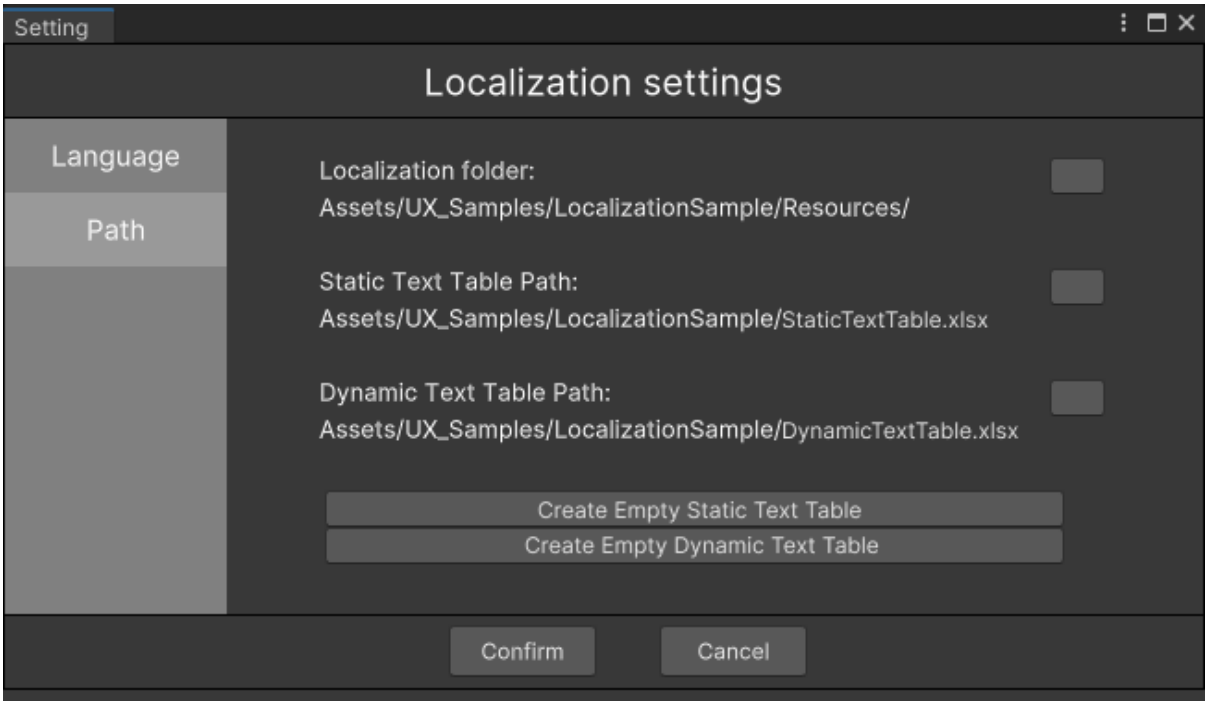
UX Text includes two text types: Static text types and Dynamic text types, which are marked in the subheadings in this manual.

Localization Overview

Configure Localization Language and Path

Click on the menu [ThunderFireUXTool->Localization->Setting] to open the following window:

In the "Language" column you can check all the languages supported by the game, which includes 14 languages in total.



In the "Path" column, you can set the localization folder, the path of the static text table, and the path of the dynamic text table.**The localization folder must be located in a path that can be loaded when running play mode (such as the Resources folder), and all three paths must be located in the "Assets/" directory.**

The localization folder stores all localized image resources and JSON files converted from the text table, named TextLocalization.json.

- The path of the static table is used for localizing static texts and is in the ".xlsx" format.
- The path of the dynamic table is used for localizing dynamic texts and is also in the ".xlsx" format.

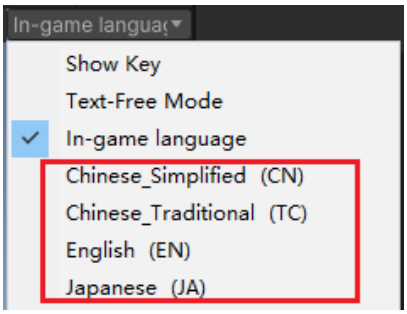
If there is no static text table, click the [Create Empty Static Table] button, select a path in the popup window, and click "Save" to generate an empty static table. It includes keys, paths, source texts, and translations in different languages.

If there is no dynamic text table, click the [Create Empty Dynamic Table] button, select a path in the popup window, and click "Save" to generate an empty dynamic table. It includes keys and translations in different languages.

After setting the above content, click "Confirm" to make the settings effective.

Preview Language Switching In Play Mode

When running play mode, you can set the preview language in the top right corner of Game (this will not affect in-game settings). The preview language will switch all UXImage, UXText, and UXTextMeshPro with enabled localization to the corresponding language.

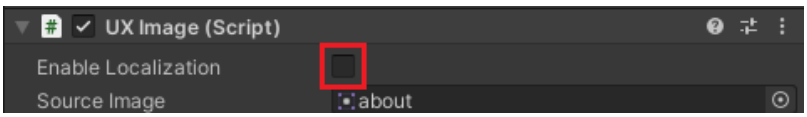


Click "In-game language" means that the preview language will not be used and the in-game settings will be applied. The default setting for the in-game language is "Not set". When the "In-Game language" is clicked while this setting is active, the program will not respond, and users will need to set the in-game language using the code (see Program Interface).

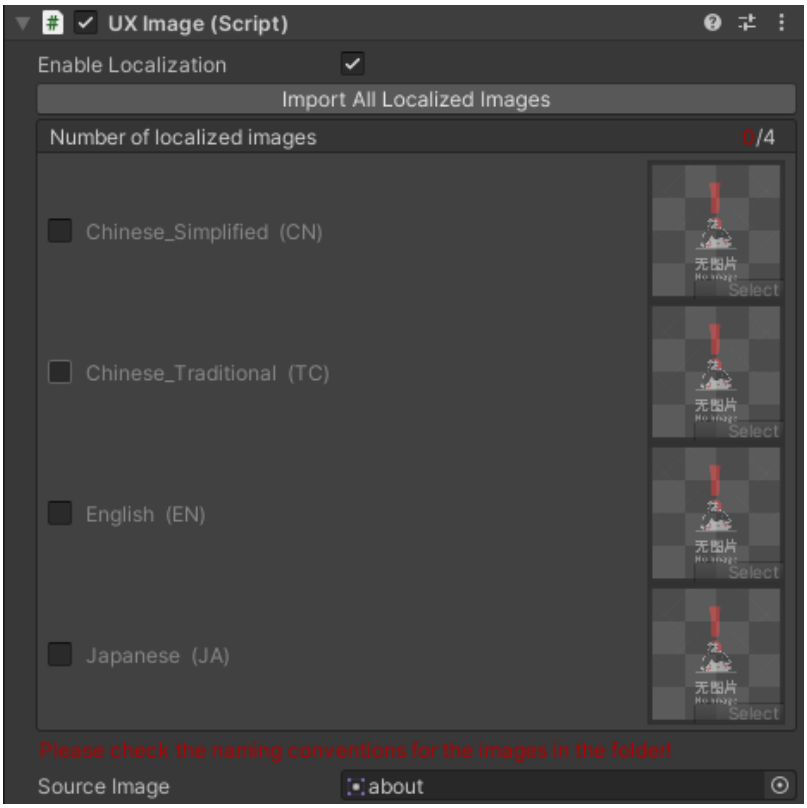
UX Image

Enable Localization

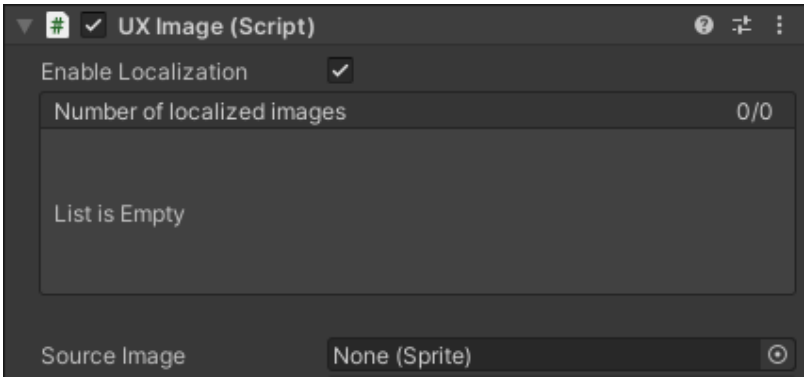
Select the UXImage that needs to be localized and Enable Localization in the Inspector.



After enabled:



Specifically, if the Source Image is empty, the "Import All Localized Images" button will not be displayed, and the list of localized graphics will also be empty.



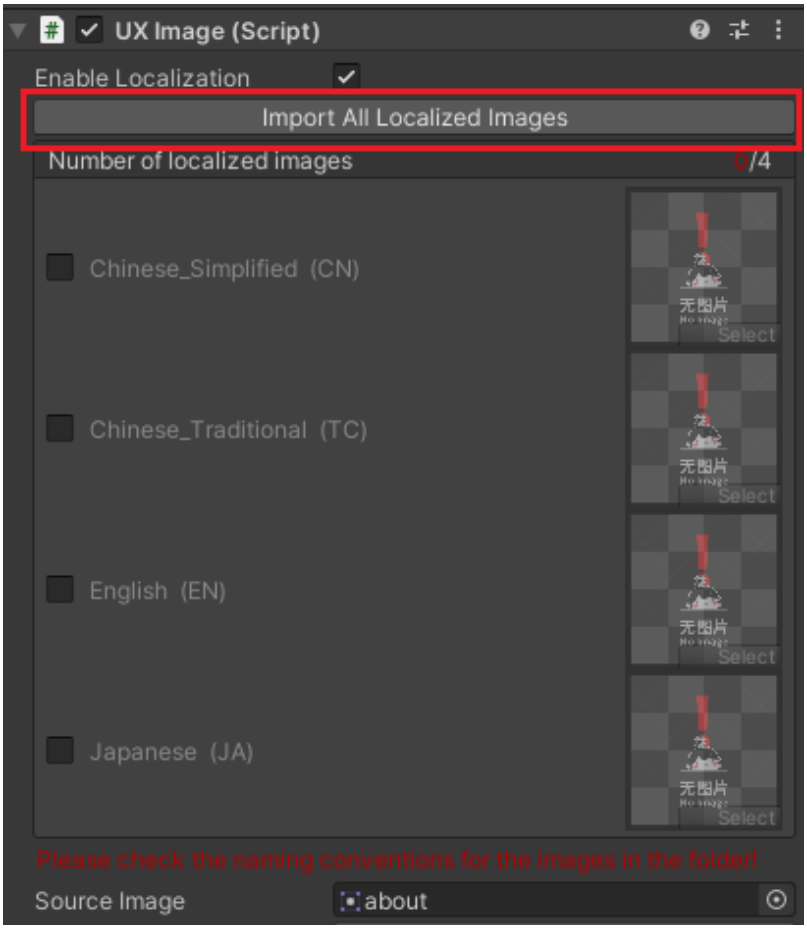
Import Images In All Languages

Clicking the "Import All Localized Images" button and selecting the folder that contains the localized images for the current Source Image will import the localized images and display them in the Inspector's list of localized graphics.

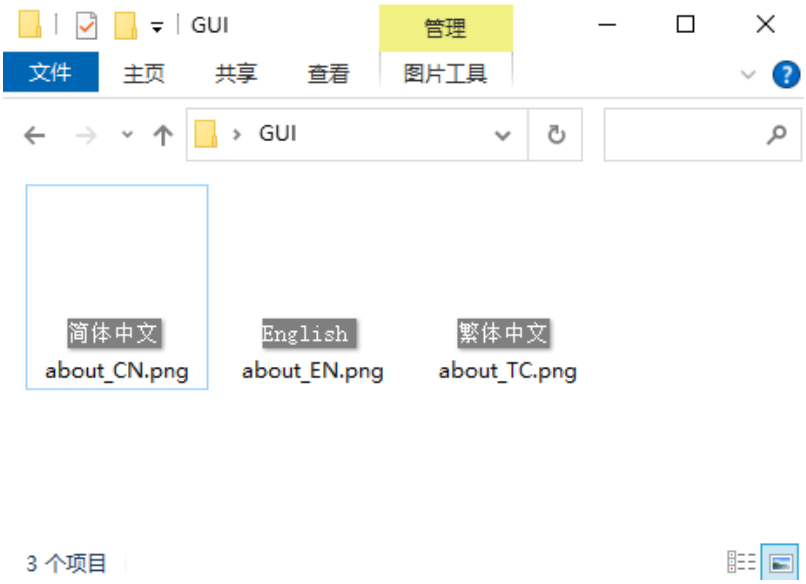
Notes:

- The folder where the localized images are located does not have to be in the "Assets/" directory.
- The program will recursively search for all files within the selected folder.
- The image should be named as "original image name_language suffix", with the extension ".png".
- The path to import is "localization folder/language suffix/image full name". If there is already an image with the same name in that path, it will be overwritten.

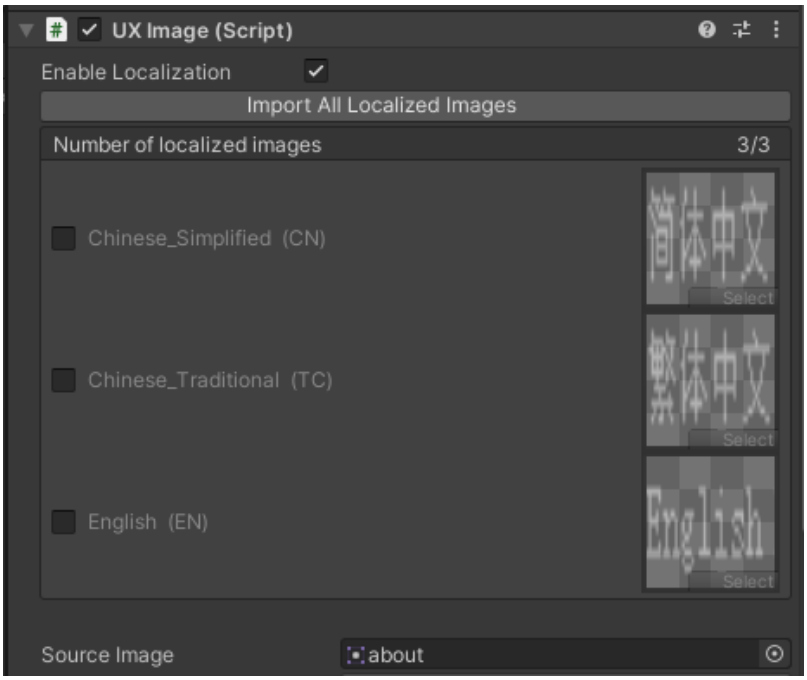
- If the "Automatically Convert Texture to Sprite" option is selected in the UX tool's "Feature Switch" settings, the imported image will be automatically converted to the Sprite.



Selected folder:



After importing:



In the example above, the program will recognize files named "about_CN.png", "about_TC.png", and "about_EN.png" in order and import them to the paths "localized folder/CN/about_CN.png", "localized folder/TC/about_TC.png", and "localized folder/EN/about_EN.png", respectively.

If a certain language's image is missing, a placeholder image will be used instead.

Batch Importing Images

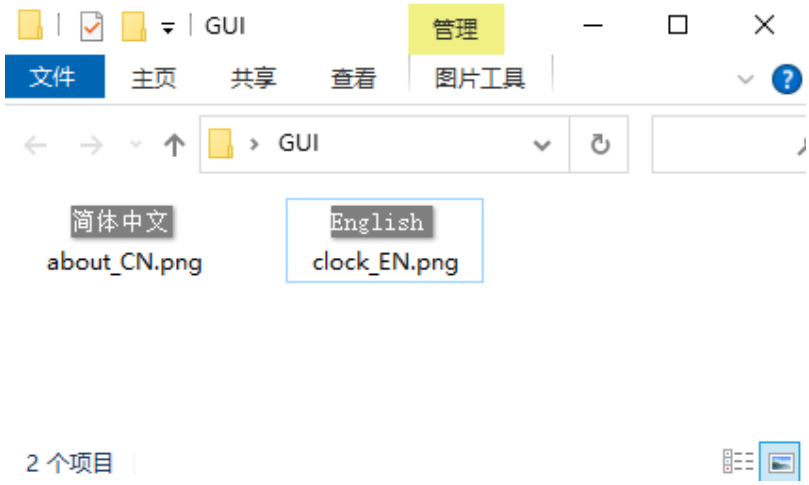
Click on the menu [ThunderFireUXTool-> Localization -> Import Localization Images] and select the localization images folder, you can import images in bulk.

Notes:

- The folder containing the localized images does not have to be in the "Assets/" directory.

- The program will recursively search for all files within the selected folder.
- The image should be named as "original image name_language suffix", with the extension ".png".
- The path to import is "localization folder/language suffix/image full name". If there is already an image with the same name in that path, it will be overwritten.
- If the "Automatically Convert Texture to Sprite" option is selected in the UX tool's "Feature Switch" settings, the imported image will be automatically converted to the Sprite.

For example, import the following folders:



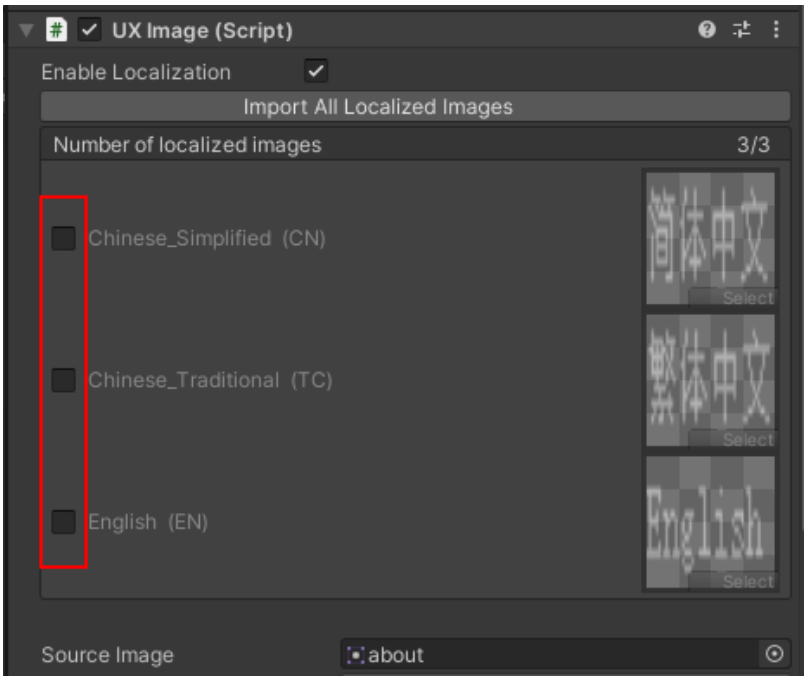
The program will import "about_CN.png" into the path "localized folder/CN/about_CN.png" and "clock_EN.png" into the path "localized folder/EN/clock_EN.png".

Automatic Recognition Of Source Image Modifications

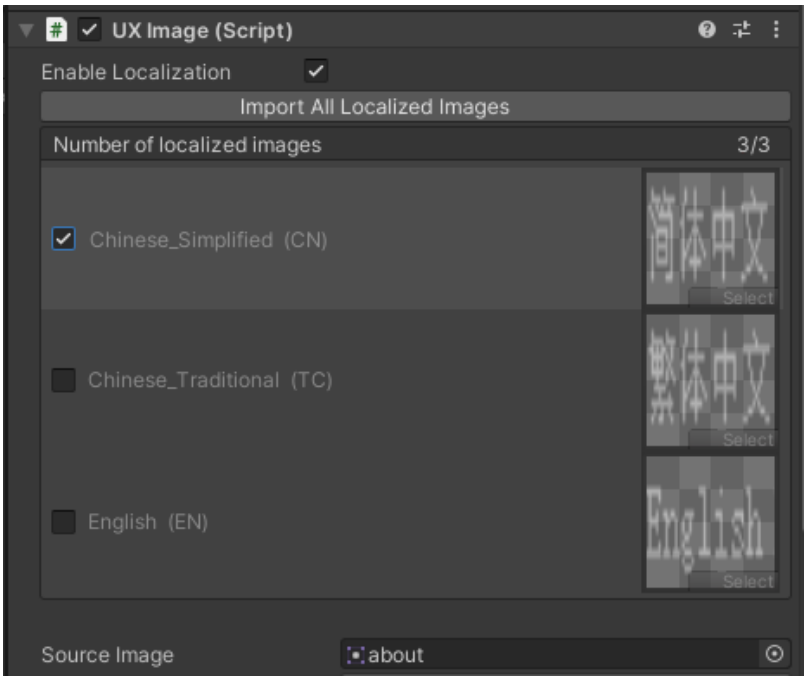
When the Source Image is changed in the Inspector, the list of localized graphics will also be automatically updated.

Preview Images In Different Languages

In the localized graphics list in the Inspector, selecting a specific language will allow you to preview the image in that language in the Scene. The preview does not modify the original file.



After selected:



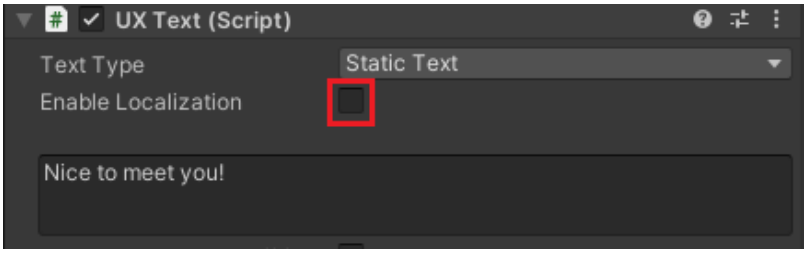
In the Scene:

Note: This feature will automatically set the current object to be invisible in the Hierarchy. If there are overlapping images, please check if  is open in Scene.

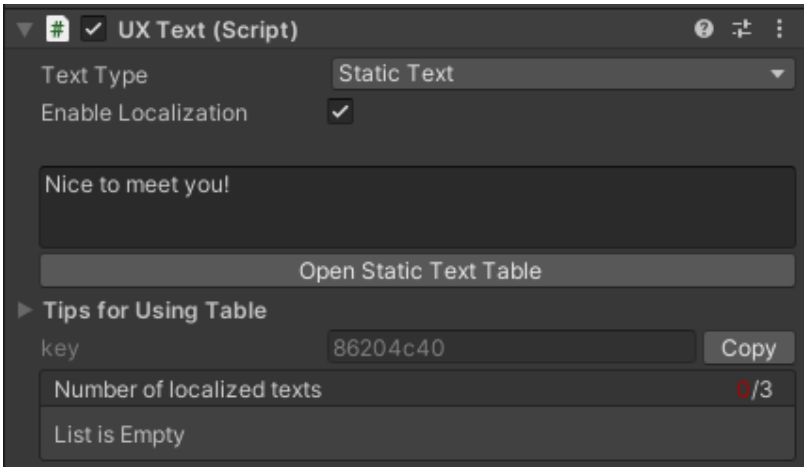
UX Text

Enable localization

Select the UXText or UXTextMeshPro that needs to be localized and Enable Localization in Inspector.



After enabled:

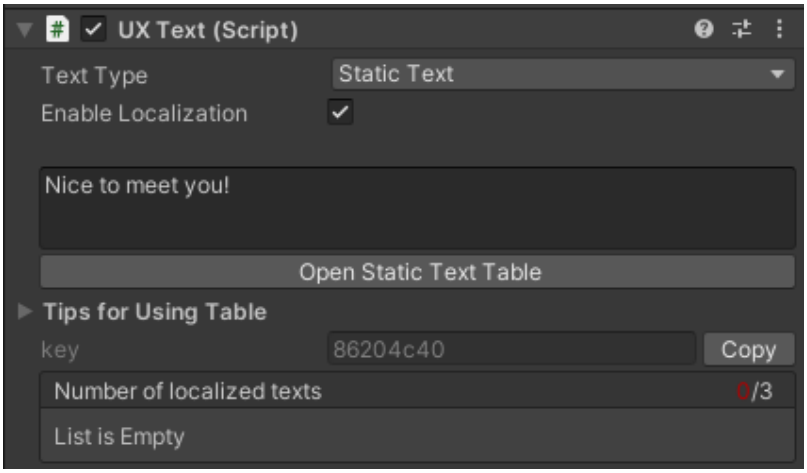


Static Text

Static text is text that is created and edited directly in the Scene or Prefab, and can be previewed directly at play mode. To change the text type to "Static Text" and configure the text content:

Fill in the content to be translated

Click on "Open Static Text Table" will record the UXText and UXTextMeshPro information with localization enabled in the current Prefab to the table, and open the table. If the table does not exist, a window to create a new table will pop up.



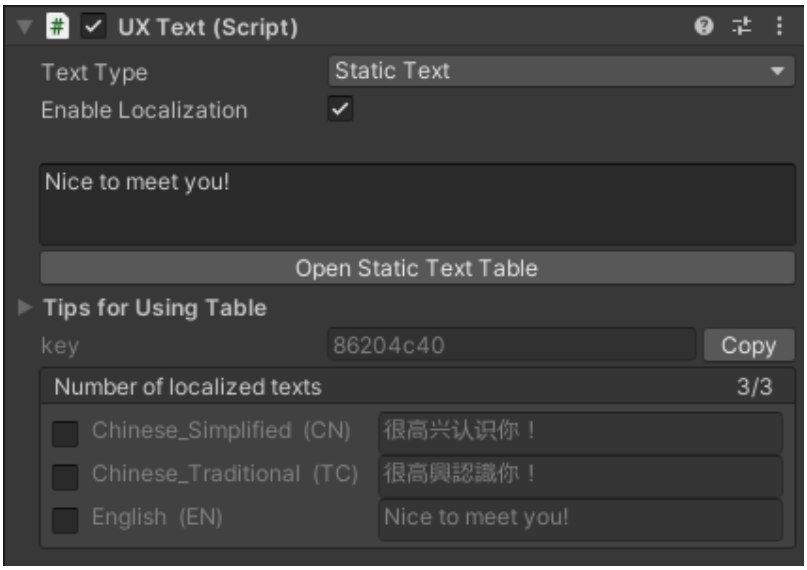
When opened:

	A	B	C	D	E	F
1	key	path	original text	Chinese_Simplified	Chinese_Traditional	English (EN)
2	86204c40	Assets/Resources/Prefab/Sample.prefab/UXText1	Nice to meet you!			

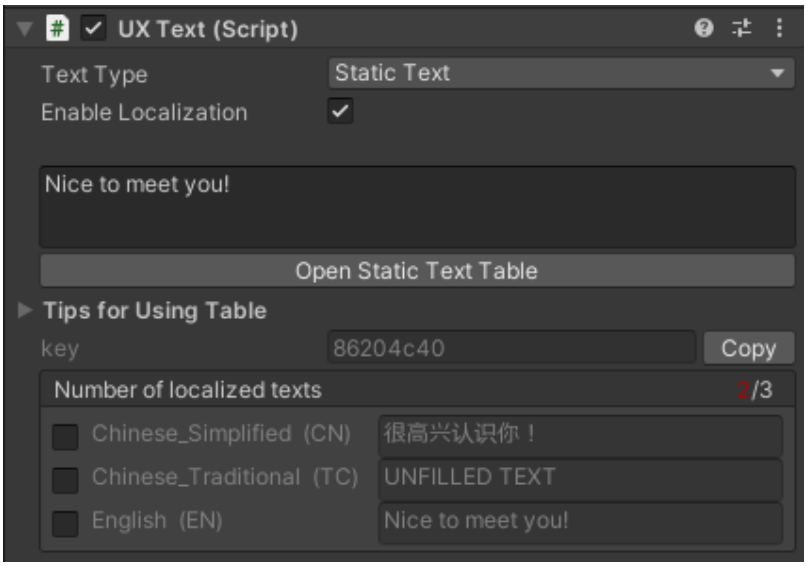
Fill in the translation data (multiple rows of data in one cell are supported).

	A	B	C	D	E	F
1	key	path	original text	Chinese_Simplified	Chinese_Traditional	English (EN)
2	86204c40	Assets/Resources/Prefab/Sample.prefab/UXText1	Nice to meet you!	很高兴认识你!	很高興認識你!	Nice to meet you!

Save and close the table, and then click on the menu [ThunderFireUXTool -> Localization -> Convert Text Table to JSON]. You can then view the effect in the localization text list in the Inspector.



If a language is not filled with translation data, "No text filled" will be displayed.

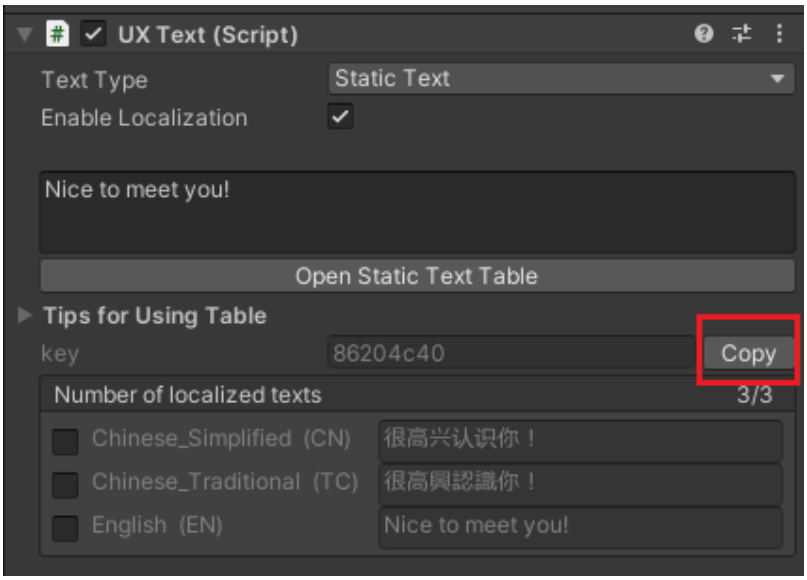


Generate Key Value

For static text types, the key value is automatically generated. When UXText or UXTextMeshPro is first selected for "Enable Localization," a unique key value is generated for it. When the original text is changed, a new unique key value will be generated.

Copy Key Value

Click the "Copy" button to copy the key value to the clipboard for quick positioning in the table.



Path Information

The second column of the static text table is used to record the path information of the corresponding object, which consists of the file name + level name. When the same object has multiple instances in different Prefabs, the path information will record all these paths, separated by "&&".

Stnchronize Table

Click on the menu "ThunderFireUXTool -> Localization -> Refresh Runtime-Use Text Table" to manually synchronize the static text table. Manual synchronization will use the information from all UXText and UXTextMeshPro that have localization enabled and text type set as static text in all prefabs.

Also, when saving a prefab, the information from all UXText and UXTextMeshPro that have localization enabled and text type set as static text in the current prefab will be automatically synchronized.

User Data Protection

When synchronizing the table, the translated data that has already been filled in by the user will be automatically preserved, even if the corresponding object no longer exists (in this case, the path and source text will be empty).

Exception Information

There are two cases where the program will output an exception message.

- An error is reported when the program tries to write to a table but the table is already open.
- When two objects have the same key value, but the source text is different, a warning will be reported (usually this situation is unlikely to occur).

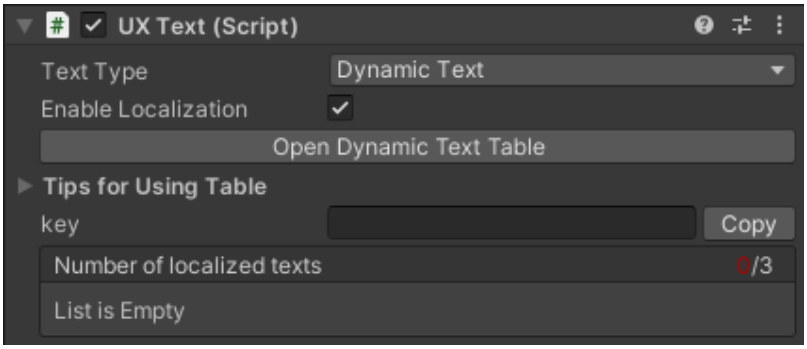
Dynamic Text

Dynamic text is text that is generated in real time when running play mode or edited by the program in code, ThunderFire UX Tool provides a localized preview of such text. Change the text type to "Dynamic Text", and

then configure the text content as follows:

Fill In The Key Value

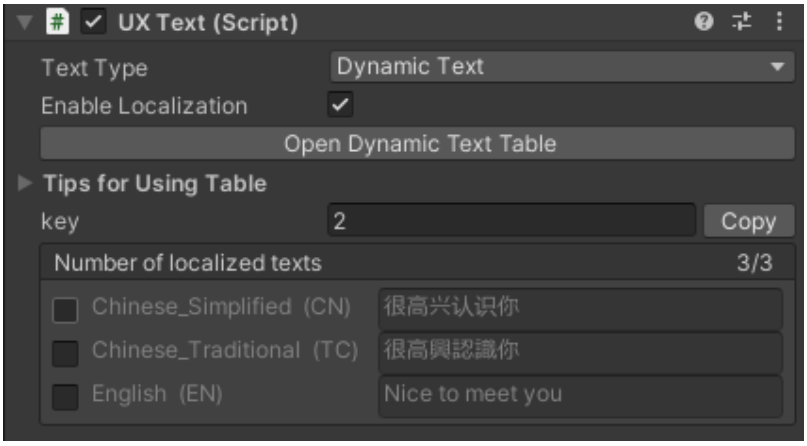
Clicking on "Open Dynamic Table" will open the form according to the "Dynamic Table Path" in the localization settings. If the table does not exist, a window will pop up to create a new table.



When opened:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	key	Chinese_Simplified (CN)	Chinese_Traditional (TC)	English (EN)	Japanese (JA)	Korean (KO)	French (FR)	German (DE)	Spanish (ES)	Russian (RU)	Turkish (TR)	Portuguese (PT	Vietnamese (V	Thai (TH)	Arabic (AR)
2	1	你好	你好	Hello											
3	2	很高兴认识你	很高興認識你	Nice to meet you											
4	3	再见	再見	Goodbye											

Find the required key value and fill it into the Inspector.



If the localized text list is empty, click on the menu [ThunderFireUXTool -> Localization -> Convert Text Table to JSON] to convert the text table to a JSON file.

Copy Key Value

Click the "Copy" button to copy the key value to the clipboard for quick positioning in the table.

Automatic Recognition Of Key Value Changes

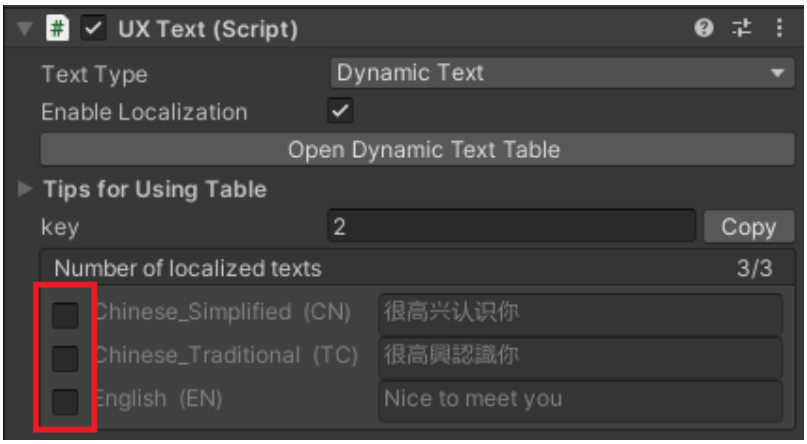
The localized text list is automatically updated when the key value in the Inspector changes.

Convert Text Tables To JSON

Click on the menu [ThunderFireUXTool-> Localization -> Convert Text Table to JSON] to convert static table and dynamic table to a JSON file with the file path "Localization folder/TextLocalization.json".

Preview Of Text In Different Languages

In the Inspector's localized text list, select a language to preview the text in that language in Scene, where the preview does not modify the original file.



After selected:

In the Scene:

Note: This feature will automatically set the current object to be invisible in the Hierarchy. If there are overlapping images, please check if  is open in Scene.

Text-Free Mode

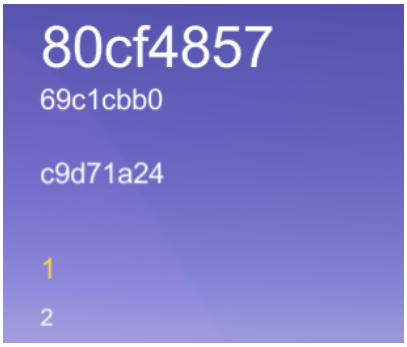
The text-free mode can be set in the upper right corner of the Game at play mode, which will replace all UXText and UXTextMeshPro with box placeholders.

After replaced:

Show key

When running play mode, you can set the display key mode in the upper right corner of the Game, which will replace all localized UXText and UXTextMeshPro with their corresponding key values.

After replaced:



Program Interface

Function

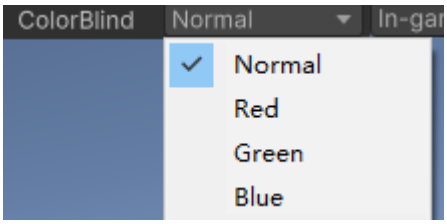
```
// Description: You can switch the in-game language
// Class：LocalizationHelper
// Parameters:
//     type: the language type after switching
// Return value: None
public static void SetLanguage(LocalizationHelper.LanguageType type)
```

ColorBlind Mode

ThunderFire UX Tool can change the color mode of a project when it runs in Unity to preview how the interface will appear for color-blind users.

How to use

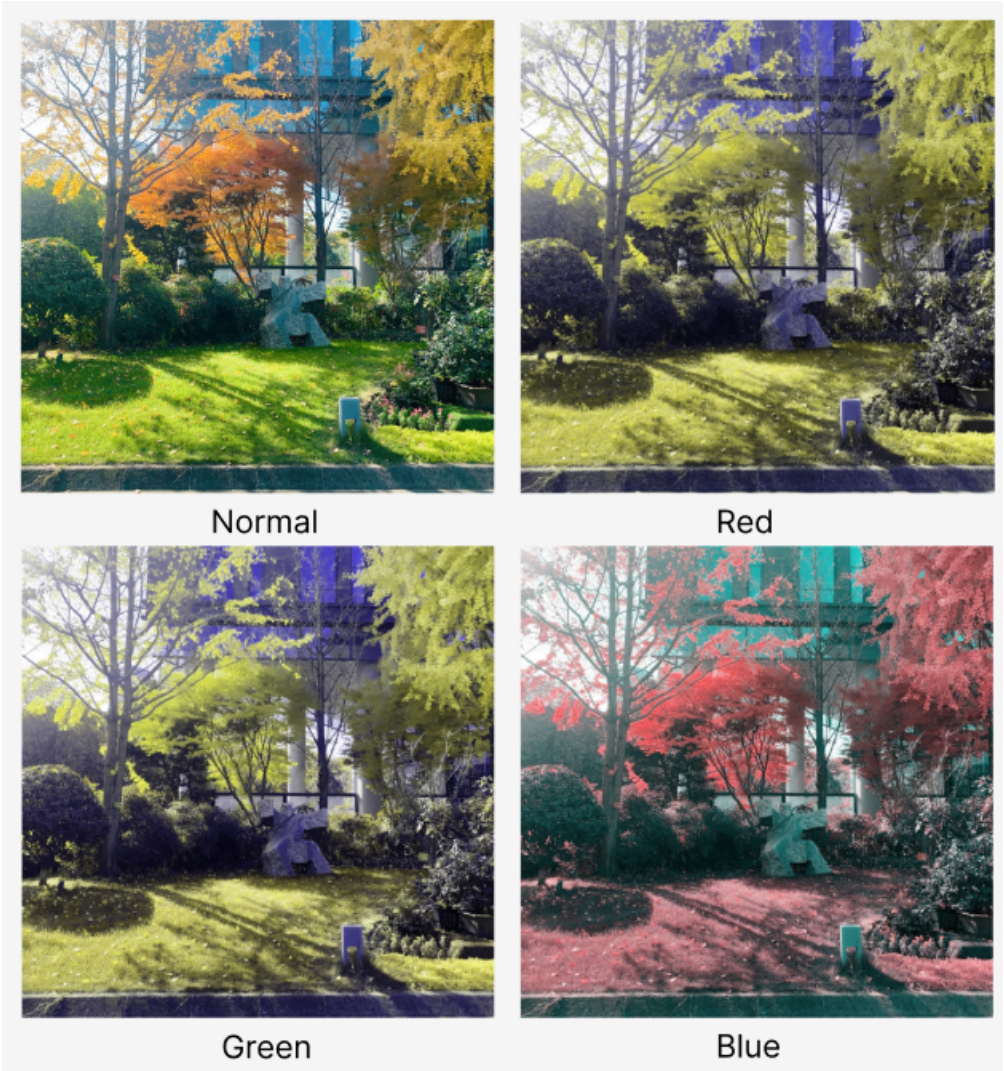
You can set the ColorBlind mode in the upper right corner of the [Game window]. The default colorblind mode is Normal, which is the interface effect in non-colorblind state. The drop-down list can be switched to Red, Green and Blue colorblind modes.



Scope

The ColorBlind mode applies to the entire screen, including 3D objects, 2D objects, UI and more.

Case



Note: The ColorBlind mode of this tool only supports the built-in rendering pipeline and will create a component called "Color Blindness Effect" on the MainCamera.

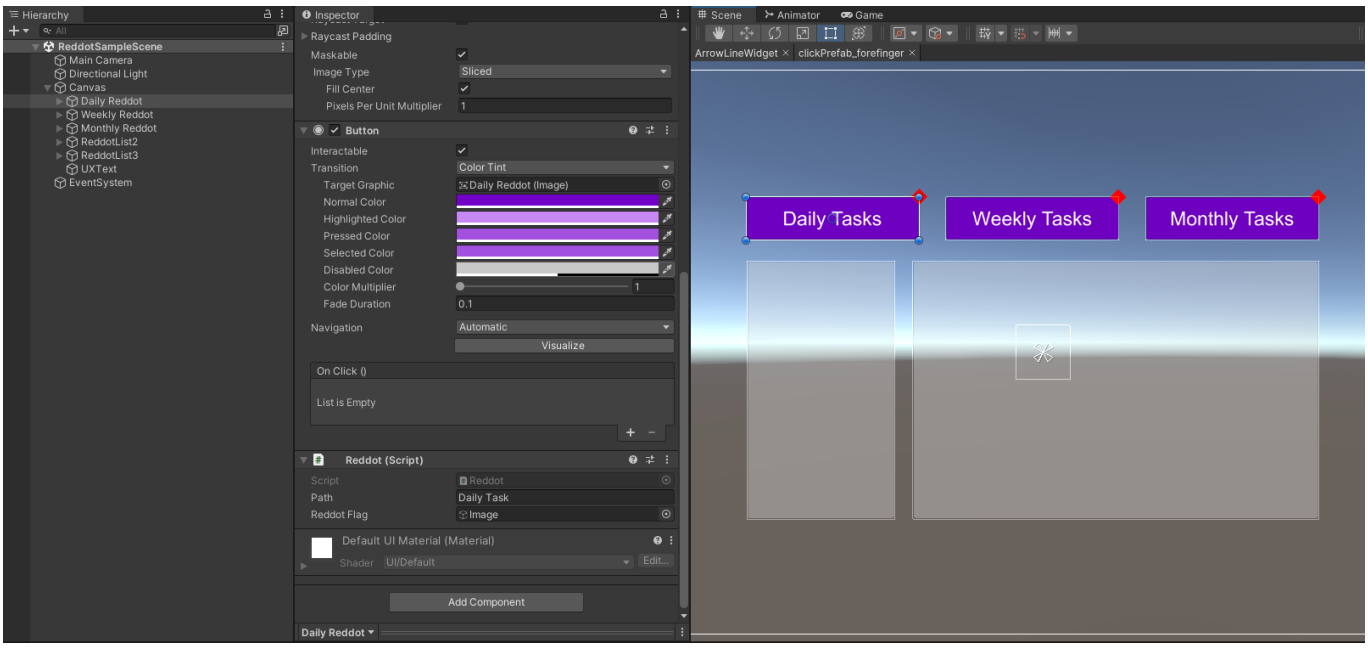
Reddot System

The Reddot system is commonly used for various UI message prompts. During the development process, there are problems such as too many hierarchies of reddots and the need for inter-level linkage settings. ThunderFire UX Tool has implemented a built-in reddot management system, which achieves unified processing of reddot hierarchies by adding paths to each reddot object.

How to use

1.Attach Component

Add a Reddot component to the object to be displayed with reddots.



2.Reddot Flag

The ReddotFlag object is the object that appears or disappears when the reddot status changes. It is recommended to set it as Children of the reddot object.

3.Reddot Path

Path represents the associated path of the reddot data, and the format is set as "stringA/stringB/stringC/stringD".

Note: When setting the Path of the second-layer reddot, the path should be set completely, which should be "stringA/stringB", not just "stringB".

- Static Configuration For reddot objects that have been determined (such as functional entry buttons on the main interface), they can be directly set in the editor.

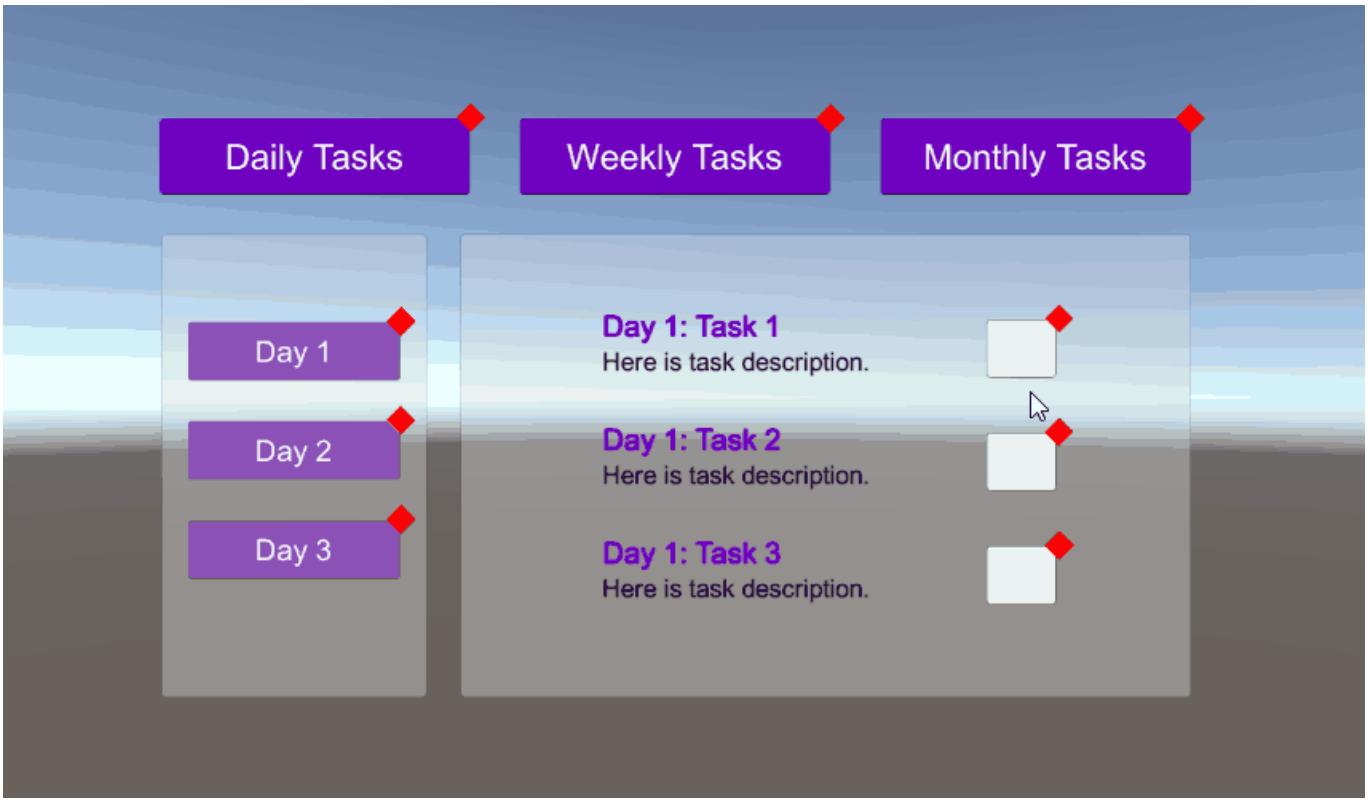
- Dynamic Configuration For reddot objects that are dynamically generated or cannot be determined during the editing phase (such as reddots in the list), they can be set through code. (Please refer to the Program Interface below for the code.)

4.Reddot Status Setting

When running the game, the reddot status can be set through code. (Please refer to the Program Interface below for code.)

After setting the reddot status, the reddot tool will update the display status of all reddots according to the path. **The update rule is that the leaf nodes in the reddot tree use their own display status, while non-leaf nodes determine their own display status based on the display status of their Children nodes**.

Case



Program Interface

Variables

```
// Red dot data correlation path
// Class：Reddot
public string Path;

// Objects that are hidden when the red dot state changes
```

```
// Class：Reddot
public GameObject reddotFlag;
```

Function

```
// Description: Set the red dots under this path to be displayed or not
// Class：ReddotManager
// Parameters:
//     isShown : whether the red dot is displayed
//     Path : red dot path
// Return value: None
public static void SetRedDotData(bool isShown, string Path)
```

Cases

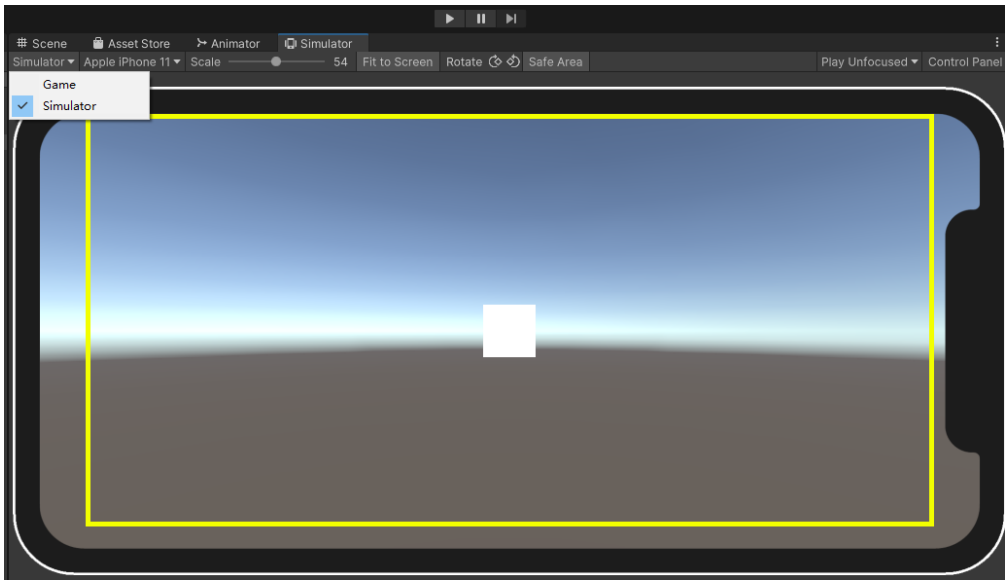
```
// Case 1
// Dynamically add red dot paths to components in the code
Reddot reddot = go.GetComponent<Reddot>();
reddot.Path = "Reddot1/firstChild";

// Case 2
// Set the red dot status
Reddot reddot = go.GetComponent<Reddot>();
ReddotManager.SetRedDotData(true, reddot.Path);
```

Notch Screen Adaptation

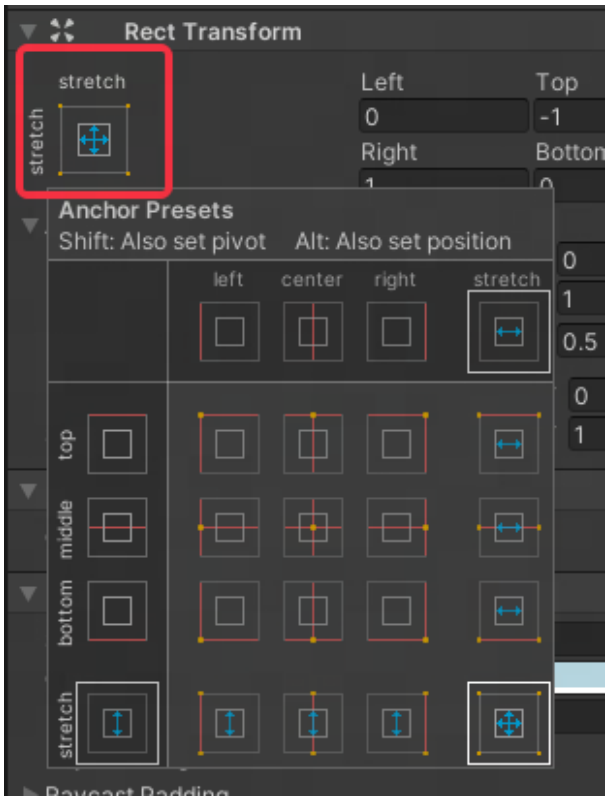
Adapting to notch screen is a technique for dynamically adjusting the position of UI to fit various sizes of the SafeArea on different devices. ThunderFire UX Tool has implemented a feature for quick adaptation to different screen sizes.

How to use

- Switch Unity Game scene to Simulator mode.

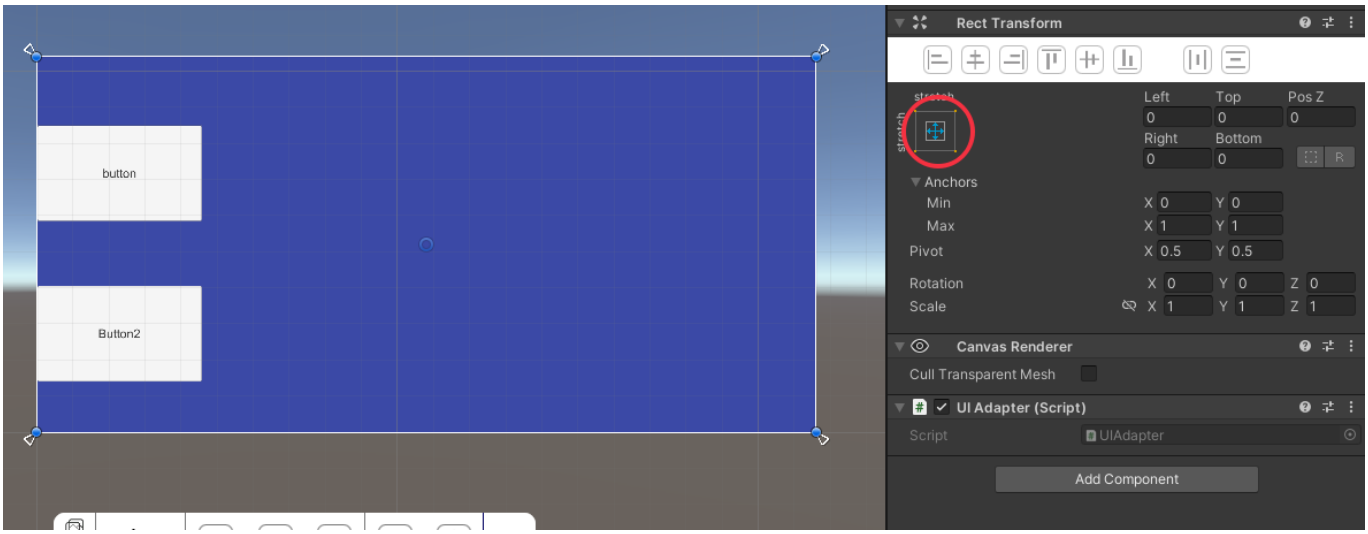


- Create a root object under the Canvas that includes all other UI widgets, and set the Anchor Preset to Stretch to fill the entire space of the Canvas. To set up, refer to the following image:



- Attach the UIAdapter component to the root object, and set the Anchor Preset of the UI widget that requires notch screen adaptation to a type other than Middle Center. For example, if you need to place a button in the top left corner of the screen and ensure adaptation, set the Anchor Preset to "top left".
- If a UI widget under the root object does not need to be adapted, attach the IgnoreUIAdapter component to it, which can ignore notch screen adaptation (mainly used for Quick Background).

UIAdapter Component

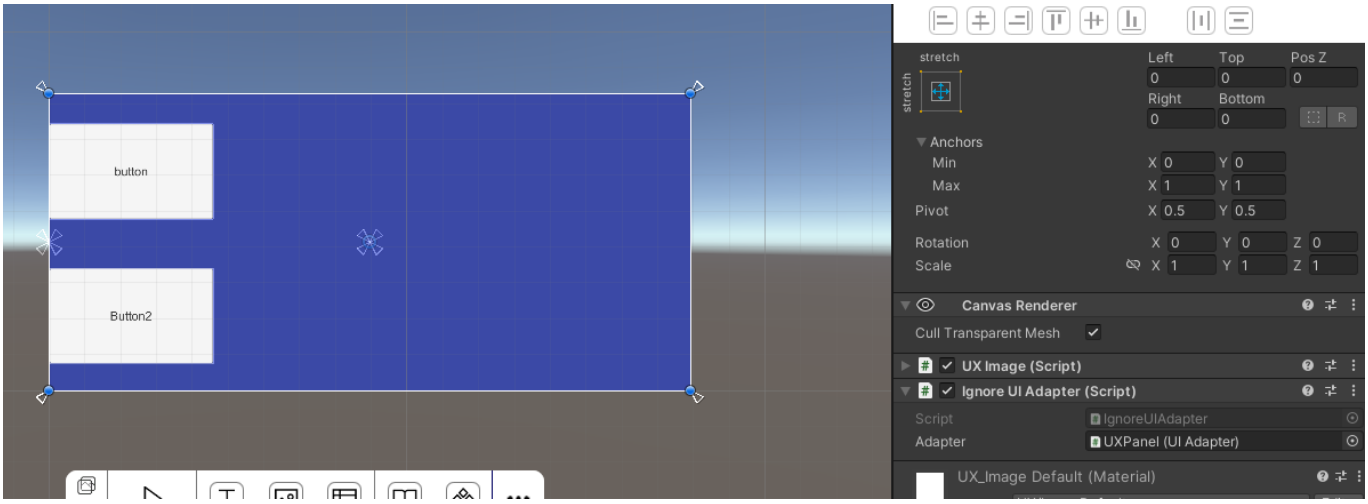


To make the UI root object adaptable to the Canvas size, it is usually necessary to set the Anchor Preset of the root object to Stretch and then adjust the UI based on it, as shown in the red circle in the figure. It is recommended to attach the UIAdapter component to this root object, which will adjust the Anchor of this root object to the appropriate value based on the scale of SafeArea, thereby completing the adaptation function for the notch screen. The steps are as follows:

1. Set the Anchor Preset of the root object to Stretch.
2. Adjust the size of the root object to the appropriate size.
3. Attach the UIAdapter component to the root object.
4. Open the simulator, select the device model, and run to see the effect.

Note: If the child object in UI needs to be adapted to the notch screen, set its Anchor Preset to a type other than Middle Center.

IgnoreUIAdapter Component



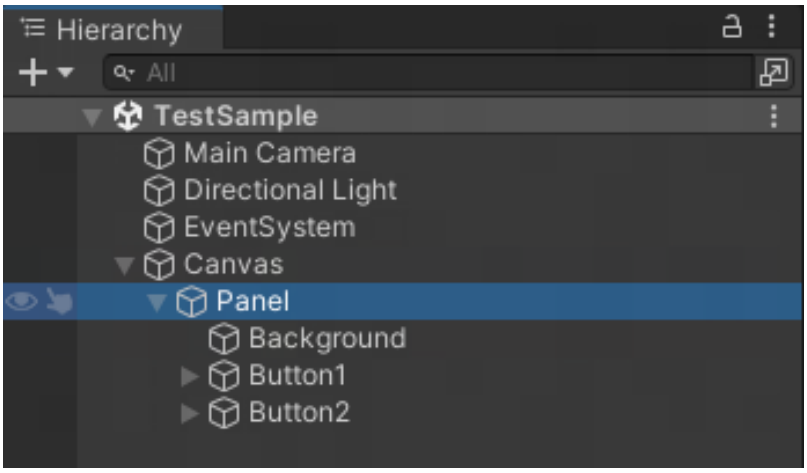
For some UI elements (such as in-game backgrounds) that do not need to be adapted to the notch screen, you can attach the IgnoreUIAdapter component to them and set the UIAdapter of their parent object in the Inspector panel. We will calculate the offset of the UIAdapter and add the original offset to this object to achieve the effect of ignoring the Adapter in the parent object. Generally, the Anchor Preset of these backgrounds should also be set to Stretch. The steps are as follows:

1. Set the Anchor Preset of the background to Stretch.
2. Adjust the size.
3. Attach the IgnoreUIAdapter component. Drag the parent's UIAdapter in the Inspector and assign its values to the component. (If not assigned, the component will be searched in the parent node.)

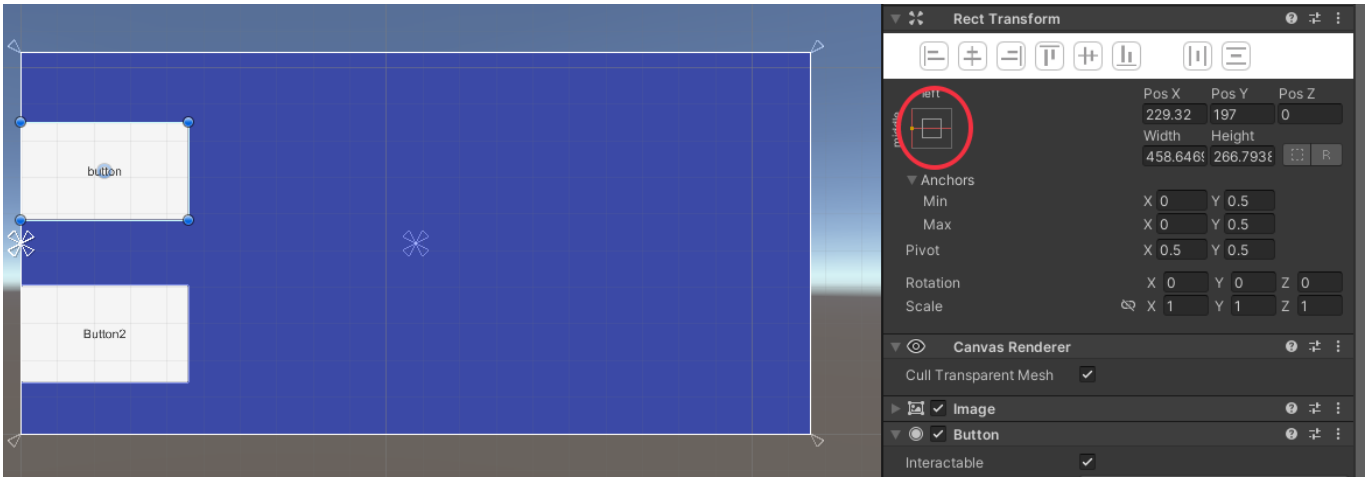
Note: Do not attach this component if the Anchor Preset of this object is set to Middle Center.

Adaptation Cases

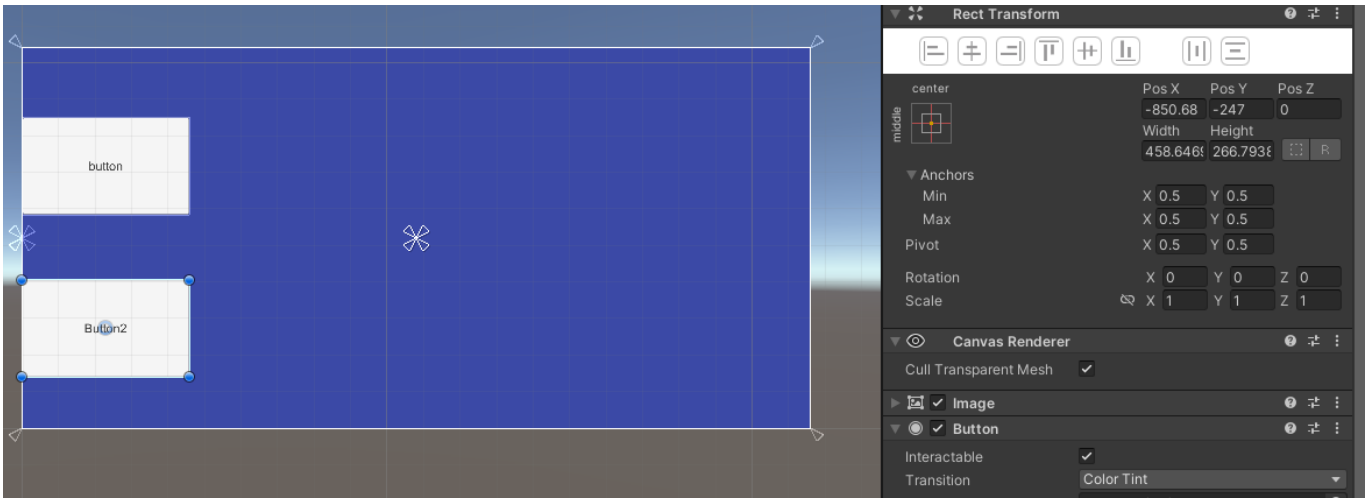
Case scene directory structure: UXPanel is the root object to be adapted, and the UIAdapter component should be attached as required in the previous section, occupying the entire Canvas space. UXImage is the background, which should be ignored according to the method mentioned earlier, and it is best to occupy the space of the root object.



Example of the first Button: Set the Anchor Preset to Middle Left for this button to be adapted to the left edge.



Example of the second Button: The Anchor Preset is set to Middle Center, so when adapting, it is not guaranteed to be within the SafeArea.



Running result: The first Button is adapted correctly, the second Button is not adapted.