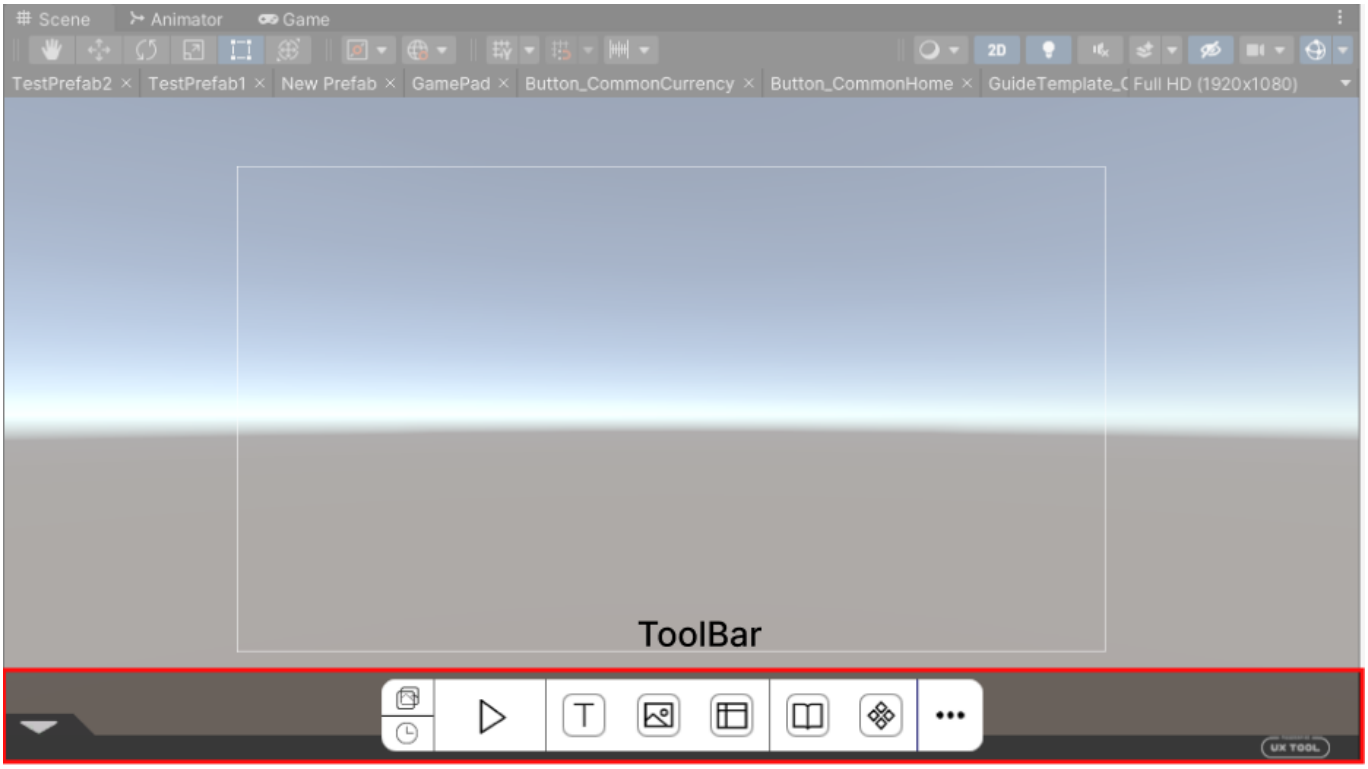
Using Tutorials

ToolBar

The ToolBar is an auxiliary toolbar for quick stitching of interfaces, located at the bottom of the Scene panel, which can be enabled or disabled through the Unity top bar [ThunderFireUXTool-工具栏 (ToolBar)].



Panel Functions

- Background Map: Create a background reference map in the scene or Prefab so that the interface can be based on the reference map.
- 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 the checkbox of UXText directly by dragging and dropping.
- Image: Create a UXImage object in the scene or Prefab, and draw UXImage's checkbox directly by dragging and dropping.
- Reference Lines: Create a set of reference auxiliary lines in the Scene or Prefab.
- Component Library: Open the Component Library panel to view all components.

-  New Component: Stow the currently selected node or Prefab as a component in the Component Library.

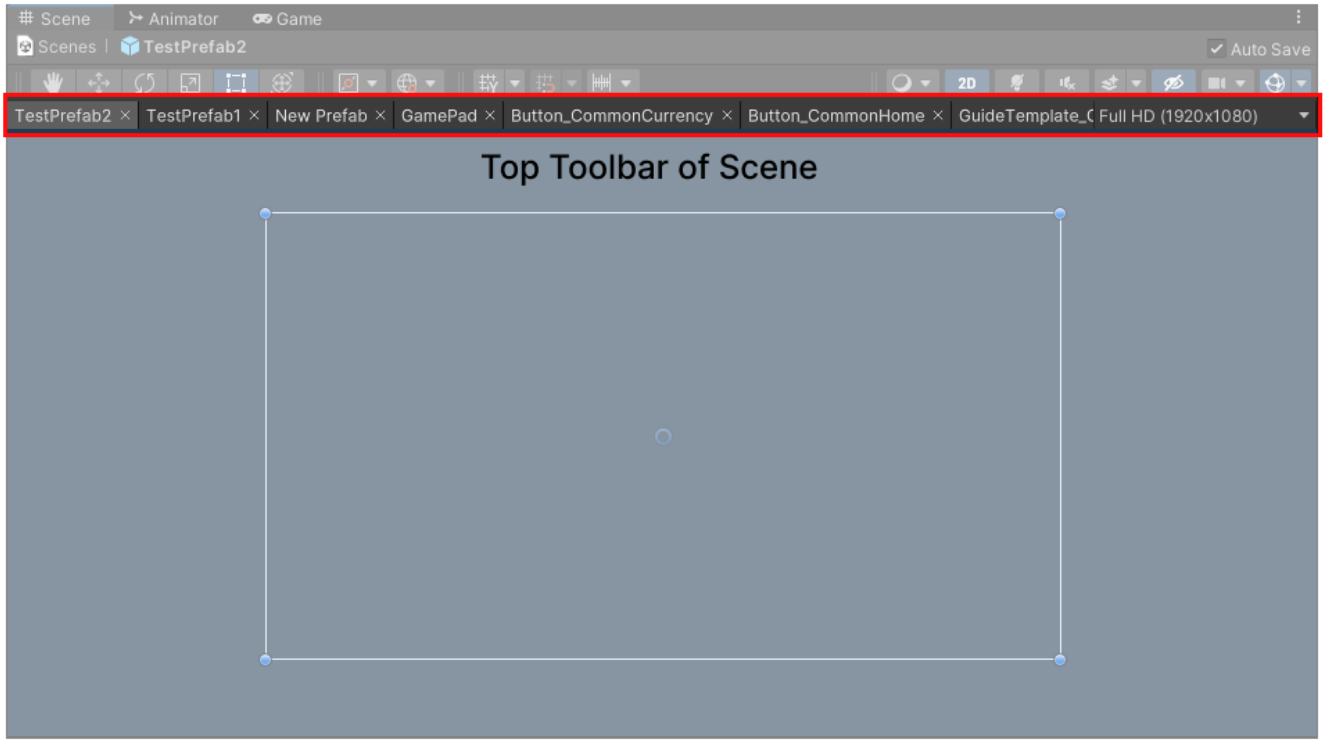
- ••• More: Opens the Tool Information panel or the Settings panel.

Hide Panel

The open ToolBar can be hidden. Click the small triangle button on the left side of the ToolBar to collapse and hide the ToolBar, and click it to open it 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.

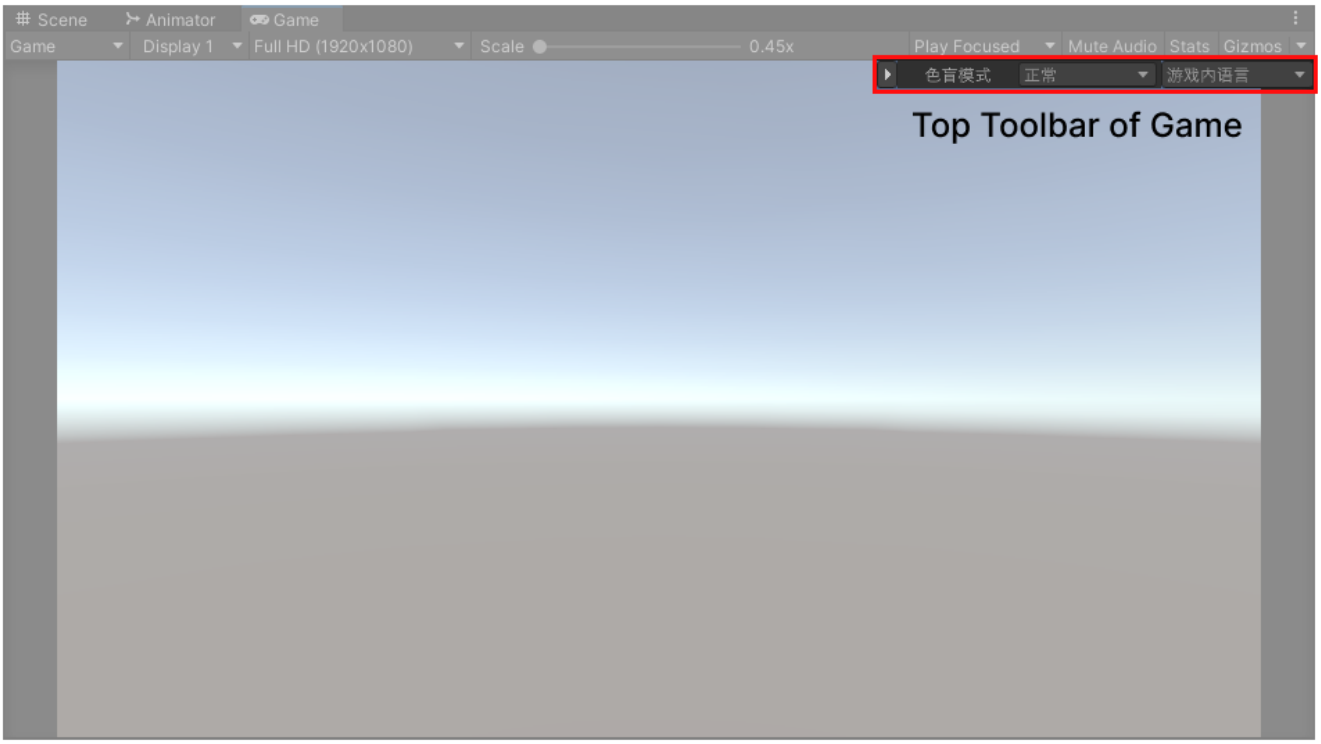
Click the × sign of a tab to close the corresponding Prefab in the tab.

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.

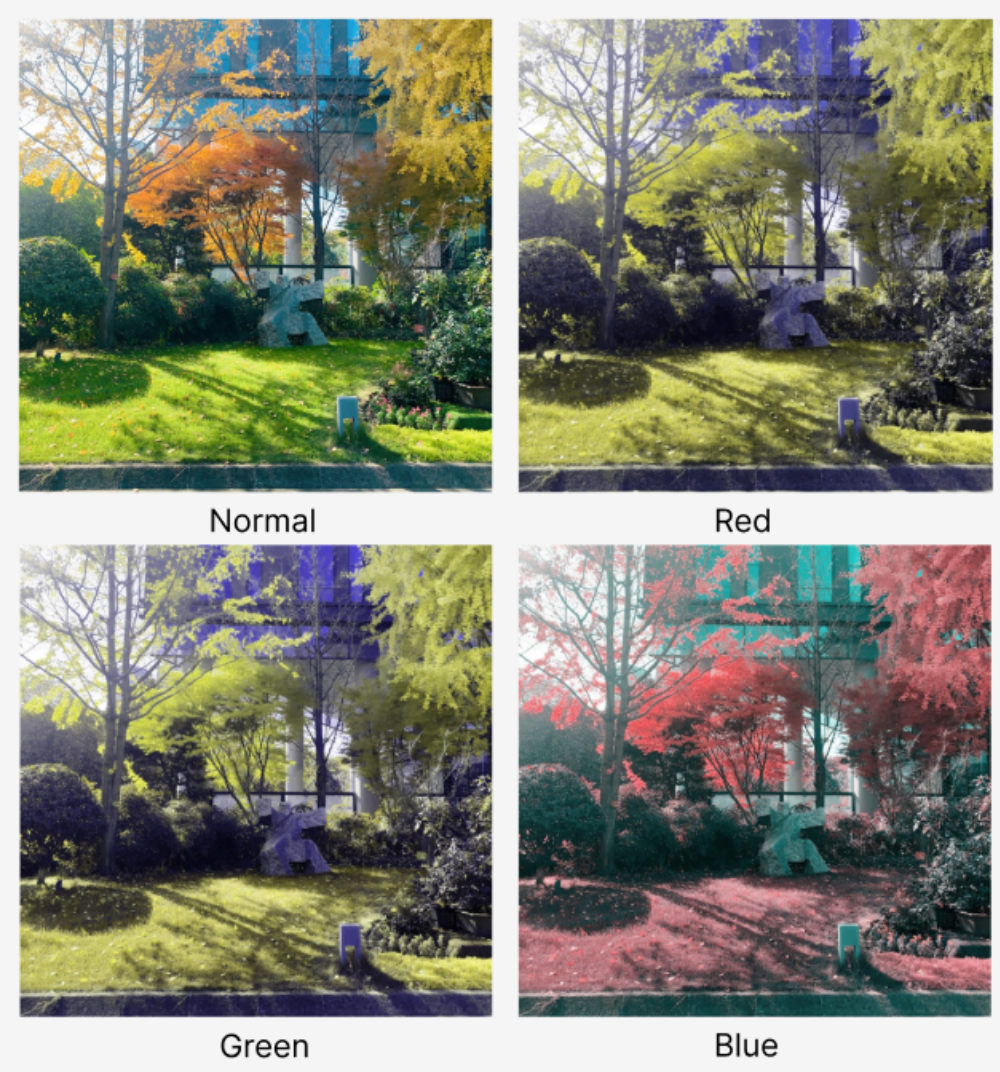


Color Blind Mode

Change the color mode of the project when running in Unity to preview how the interface will be displayed for colorblind people.

The default color-blind mode is normal, which is how the interface looks in non-color-blind states. The drop-down list allows you to toggle between red, green, and blue colorblind modes.

(Refer to the colorblind mode documentation for details)



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 specified by the project itself by default, and the in-game language will be switched as the priority to be displayed at this time.

In addition, there are No Text Mode and Key Mode in the drop-down list of the language switch.

- No Text 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.

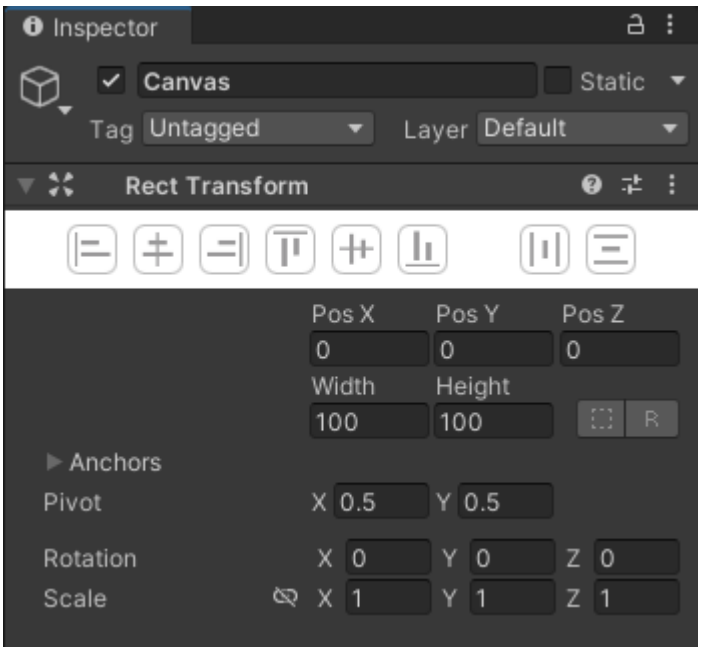
(Refer to the localization function document for more details)

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

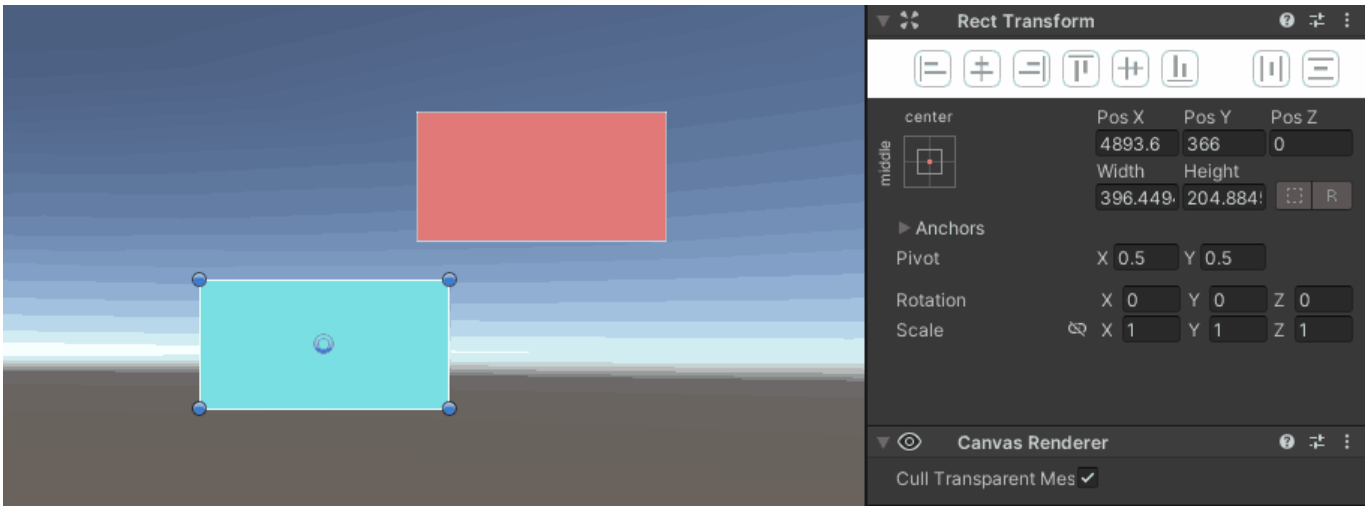
The Alignment and Distribution tool is located in the Inspector panel, under the Rect Transform component, and provides 6 types of alignment: left alignment, horizontal center alignment, right alignment, top alignment, vertical center alignment, bottom alignment, and 2 types of distribution: vertical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

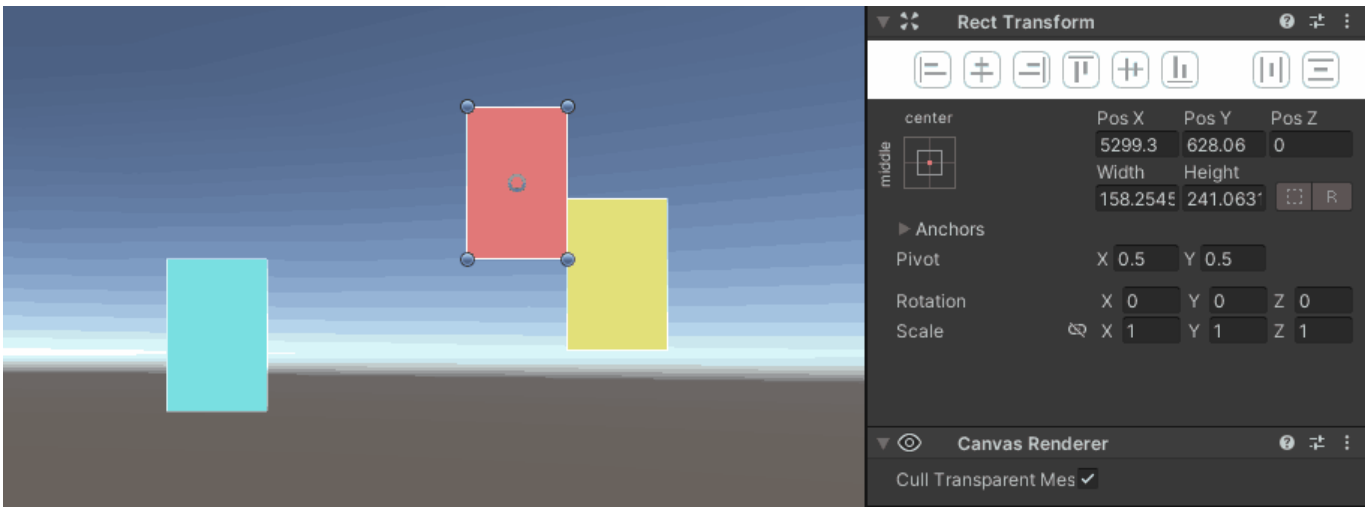
Note: When 2 or more objects are not selected, the alignment tool is not available; when the object-related parameters in the selected objects are locked (such as when they cannot be changed due to Layout component control), the alignment tool is not available.



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

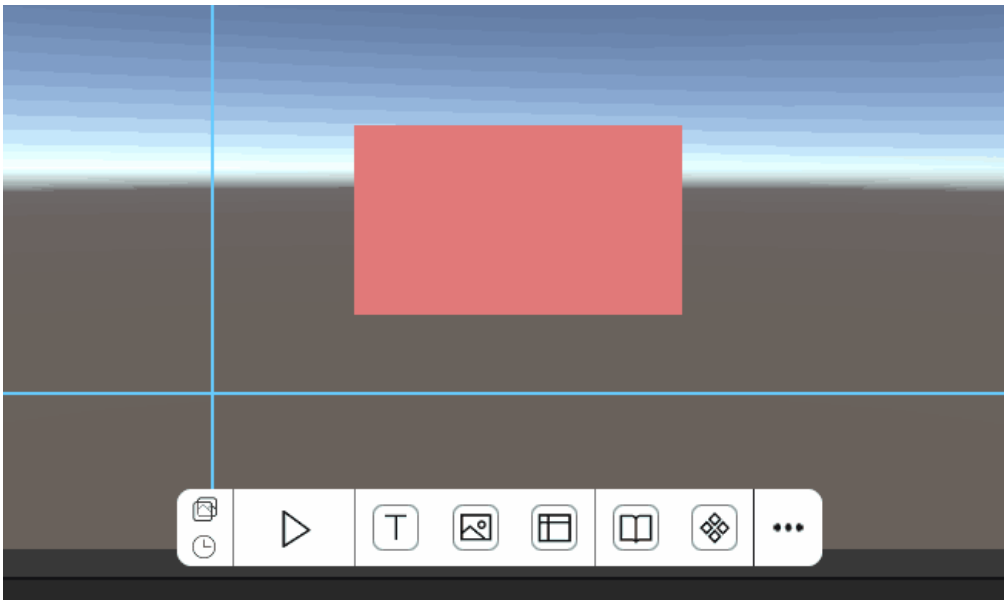
- **Horizontal distribution**:select 3 or more objects, while keeping the horizontal alignment order unchanged and the positions of the objects at the left and right ends unchanged, make the left and right edge distances between adjacent objects equal by changing the middle object position.
- **Vertical distribution**: select 3 or more objects, in the case of keeping the vertical alignment order unchanged, the position of the upper and lower ends of the objects unchanged, by changing the middle object position, so that the distance between the upper and lower edges of adjacent objects is equal.

Note: When 3 or more objects are not selected, the distribution tool is not available; when the selected objects have object-related parameters locked (such as when they cannot be changed because they are controlled by Layout component), the distribution tool is not available.



Reference Line

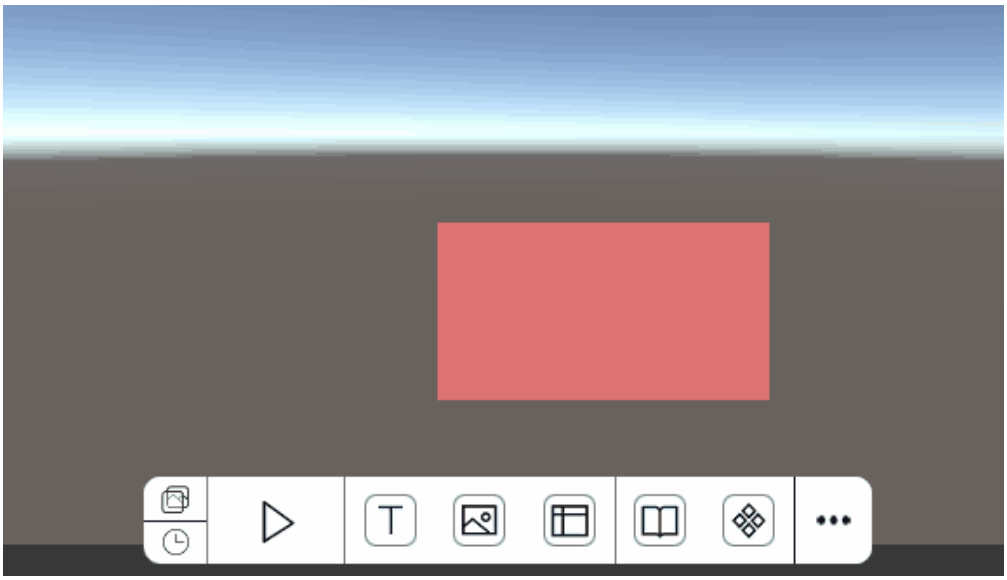
Add a reference line in Scene, and when dragging an object such as a picture or text close to the reference line, the object can automatically attach the edge to the nearby reference line.



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.



Mobile Adsorption

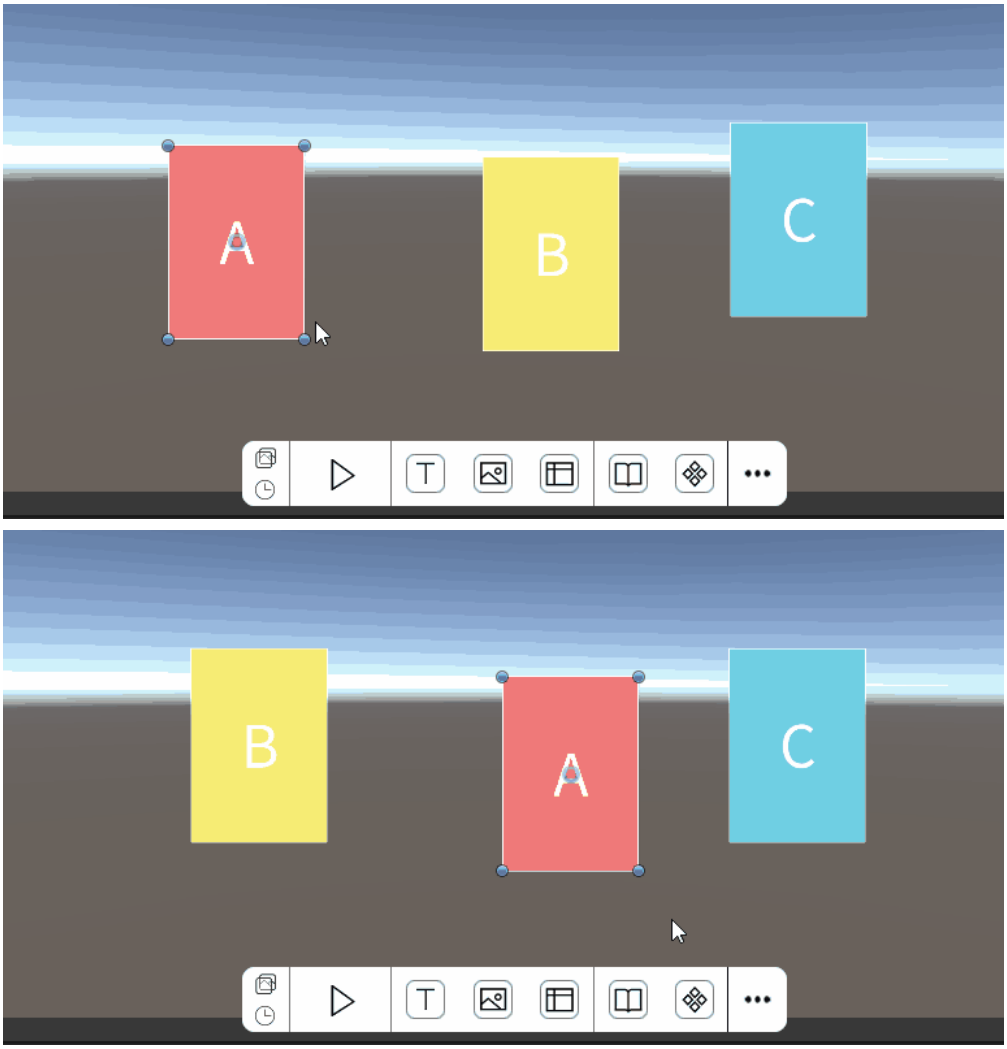
When editing multiple objects in a row in Scene, multiple automatic adsorption effects are generated between objects, including isometric, edge and center adsorption.

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 be automatically adsorbed to the exact position of the isometric

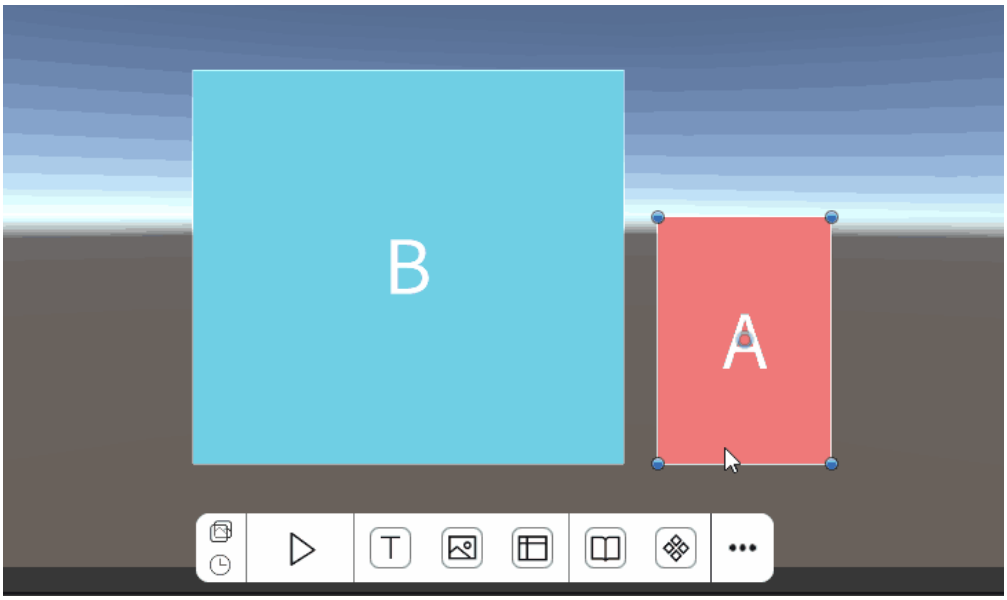
distance and display the numerical mark of this distance. (Here the distance refers to the distance between the edge lines of two objects relative to each other)

- 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 be automatically adsorbed 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 adsorb 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.

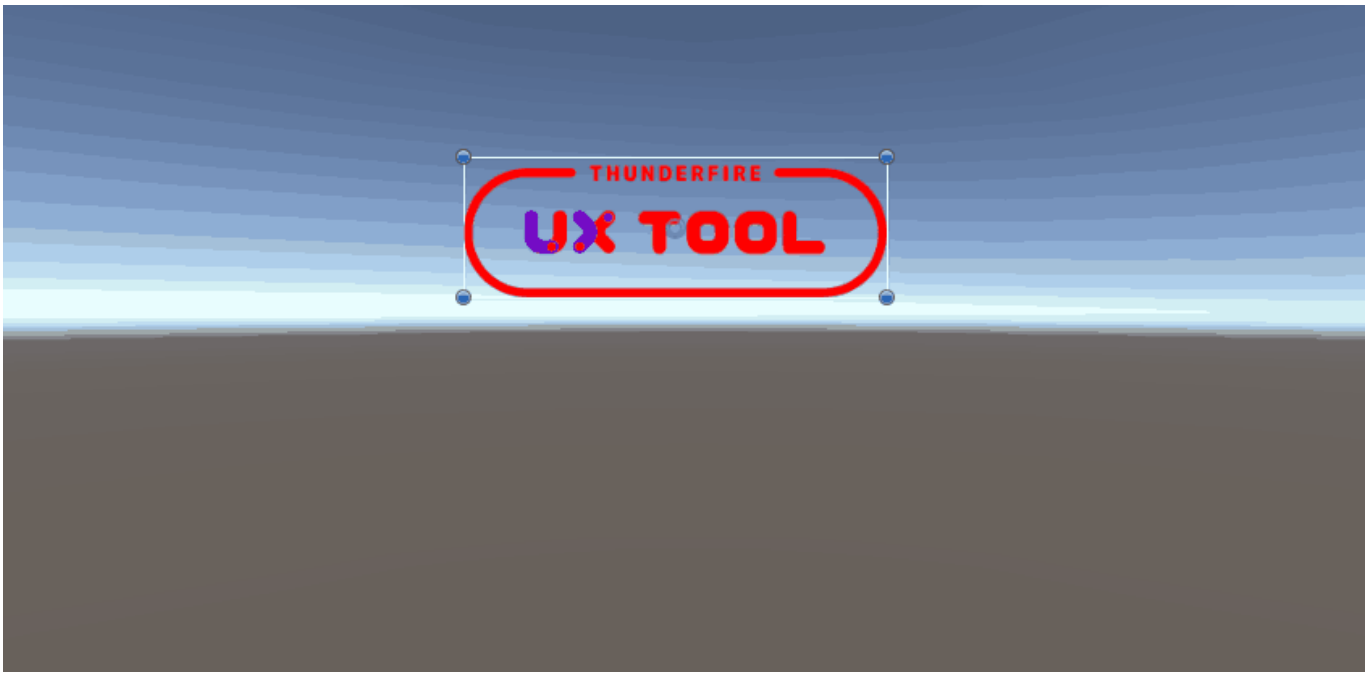
Position Micromotion

When one or more objects are selected in the 2D scene or Prefab, the up, down, left, and right arrow keys on the keyboard can be used to micromanage the selected objects.

Note: This feature overrides Unity's original lens shifting feature.

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.

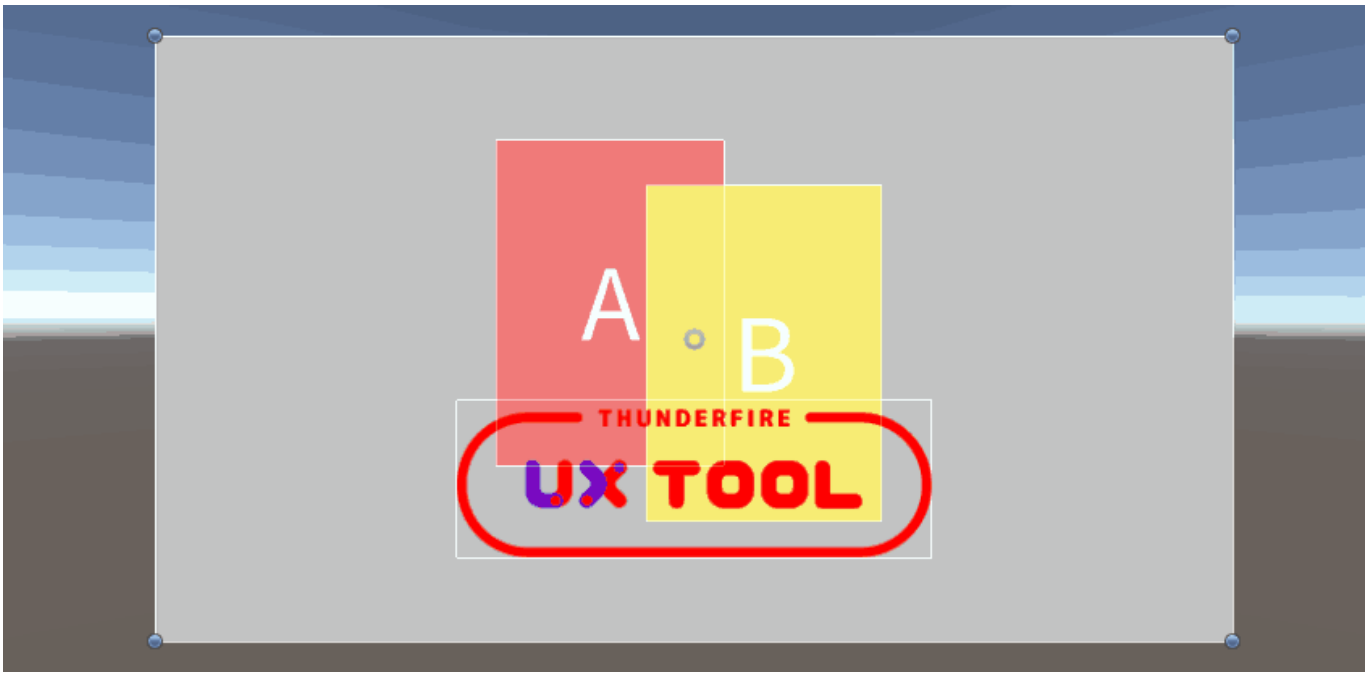
Delete : Delete the selected objects.

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.

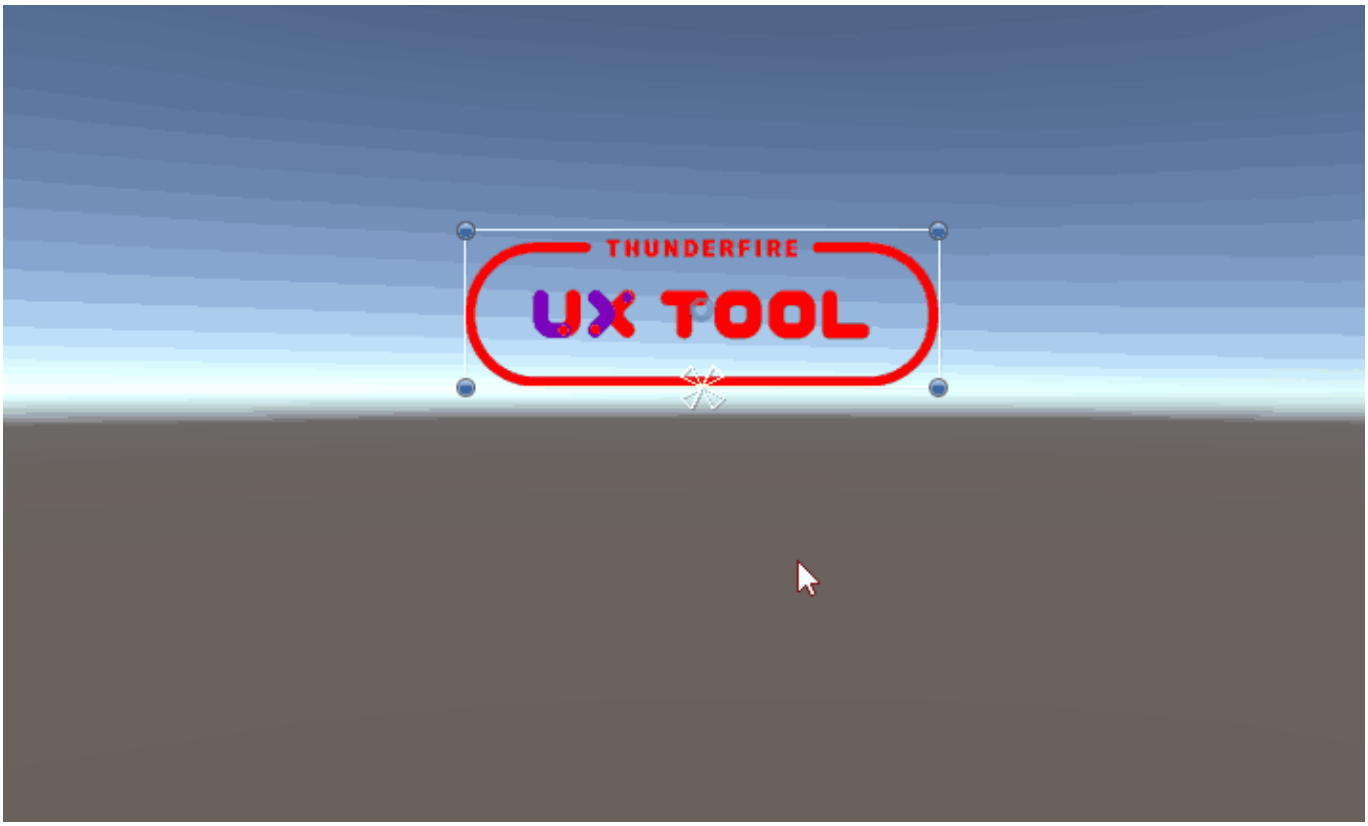


Quick Copy

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 become the copied object. After releasing the left mouse button, the copying is finished.

Note:

- If you release Alt without releasing the left mouse button during copying, the generated copied objects will disappear and the copying is interrupted.
- If the Layout tool function is enabled in the settings, the distance mark and automatic alignment between the copied object and other objects in the scene will also be displayed during the quick copy process.



Widget Repository

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

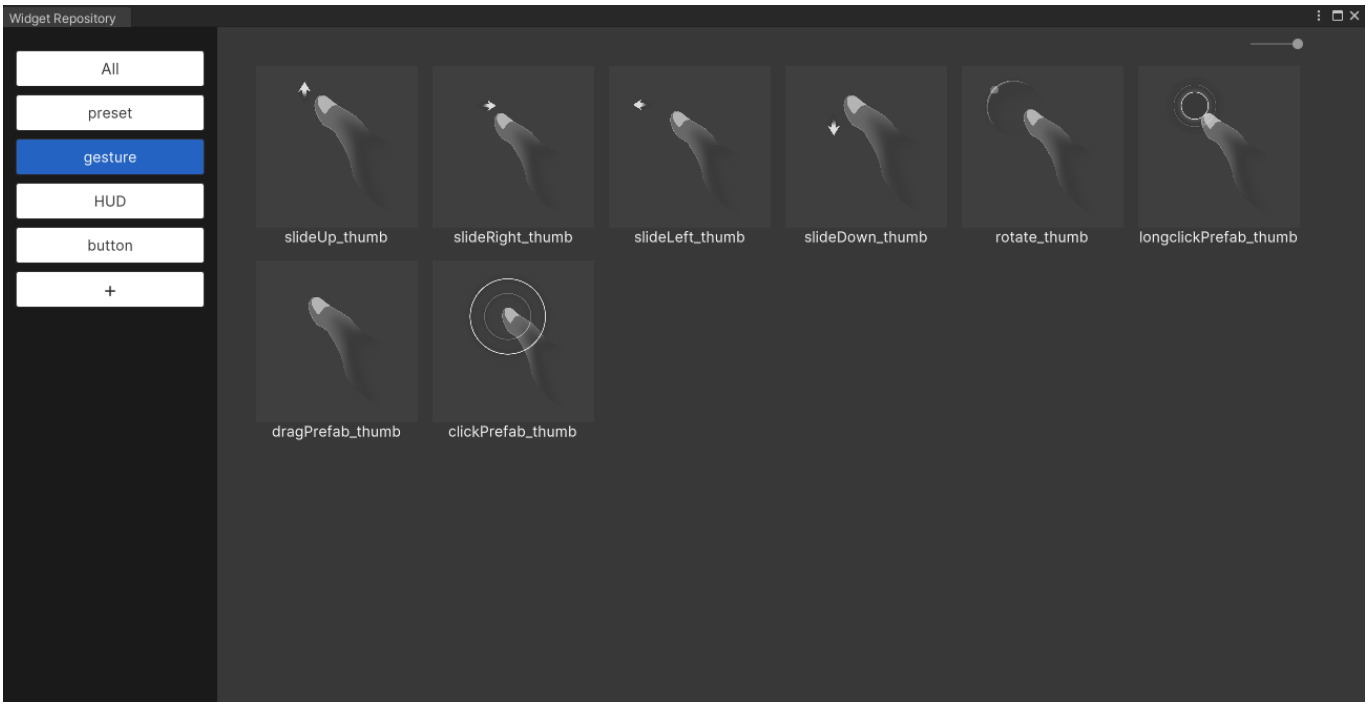
Terminology

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.

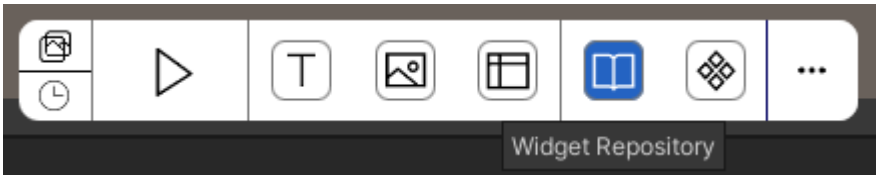
Widget Repository

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

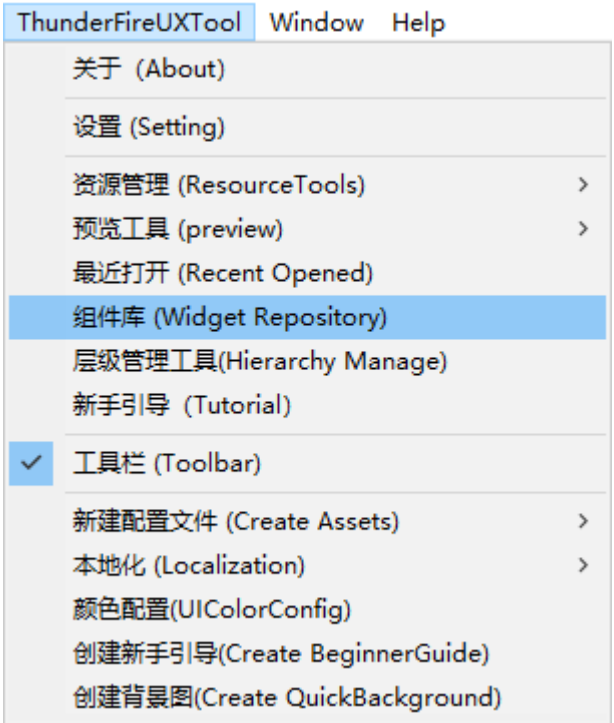


Use path

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



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

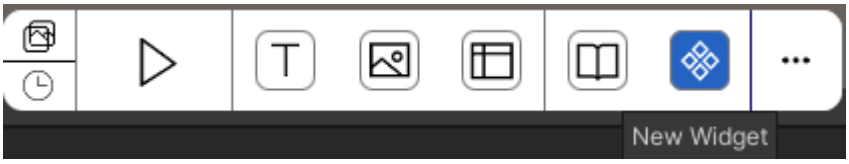


In the Widget Repository panel, the widget category tab is on the left side, click the corresponding tab to see all widgets under that category on the right side. Click the "+" sign to create a new widget category. Right-click on an existing category tab to modify the category name or delete the category.

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 Repository panel, directly drag and drop the prefab or node from Hierarchy or Project to the Widget Repository panel, a New Widget pop-up window will appear, fill in the information and you are done.

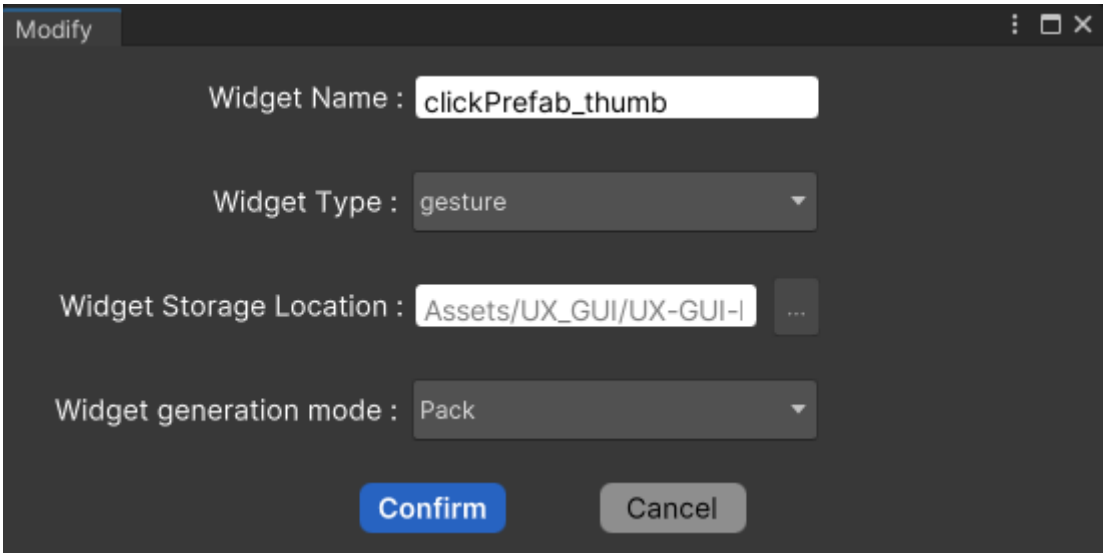
Create by right-click :Select a prefab in Project and right-click [设置为组件(Set As UXWidget)], a New Widget pop-up window will appear, fill in the information and it will be created.

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

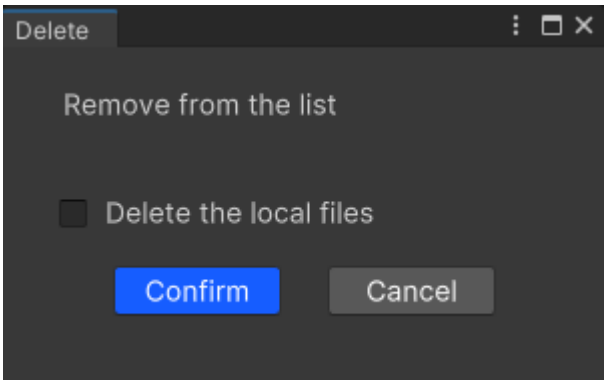
Right-click the widget in the Widget Repository to modify the widge properties, including widget name, widget type, widget storage location and widget generation mode.



- **widget name**: The name of the prefab corresponding to the widget. If you modify the widget name, the name of the prefab will also be changed.
- **widget Type**: The widget's classification in the Widget Repository. The default is "All", and the current classification will be automatically recognized when the widget is created by dragging and dropping. In addition to creating a new widget type in the Widget Repository panel, you can also create a new type in the widget information by clicking the "+" sign in the widget type drop-down list.
- **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

: Right-click the widget in the Widget Repository to delete the widget, you can choose to delete only in the Widget Repository list or delete the corresponding prefab local file at the same time.

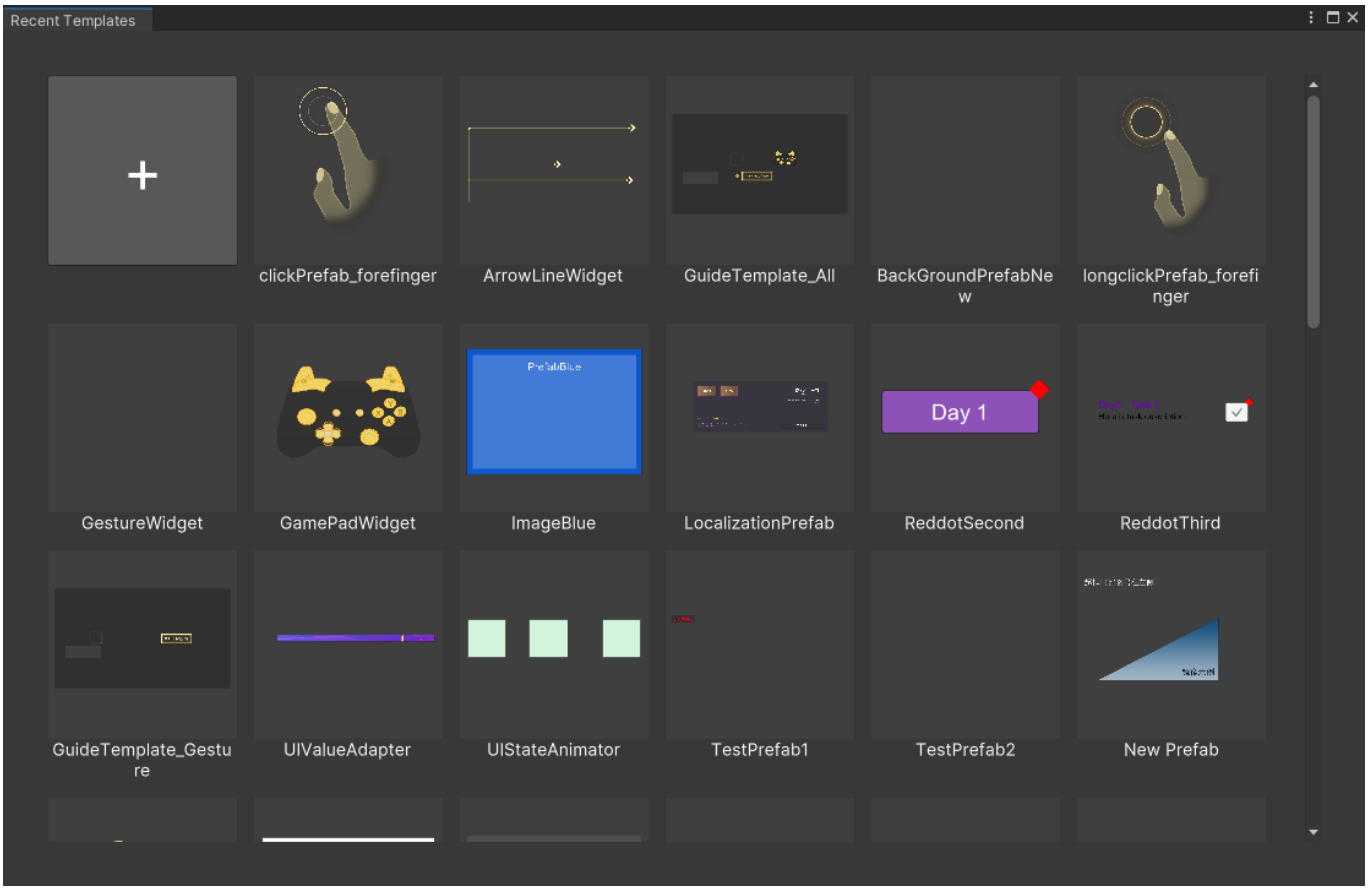


Using widgets

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 Prefab The Recently Opened Panel in ThunderFire UX Tool is designed to make it easy for users to quickly find their recently opened Prefabs and reduce the hassle of finding them in Project.

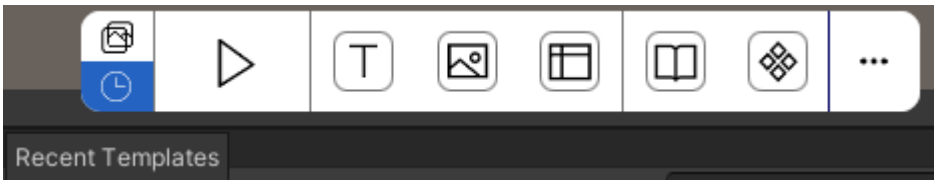
The Recently Opened Panel can be turned on or off in [ThunderFireUXTool - 设置 (Setting) - Function Toggle].



Use Path

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.

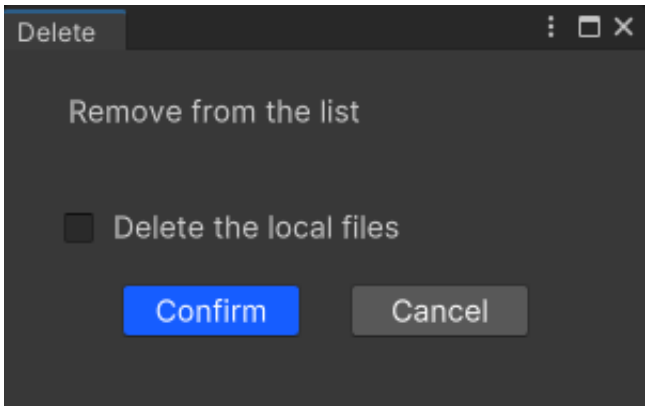
Panel Features

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.

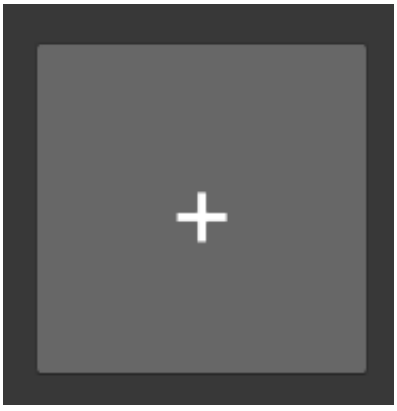
Deleting a prefab from the window

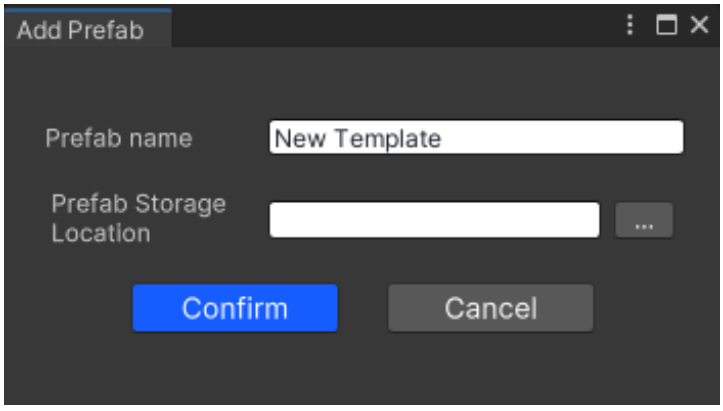
Select a prefab, right-click it and select "Delete", a pop-up window will appear, follow the prompts to delete it (you can choose to delete it only in the recently opened panel or delete the corresponding prefab local file at the same time).



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 Image

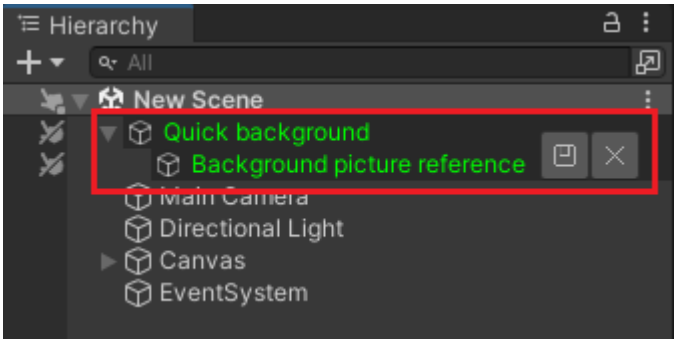
The Quick Background Map in ThunderFire UX Tool is designed to enable users to quickly create a base map for reference when stitching or reviewing an interface, while running or saving it without affecting the original hierarchical tree structure.

Use path

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

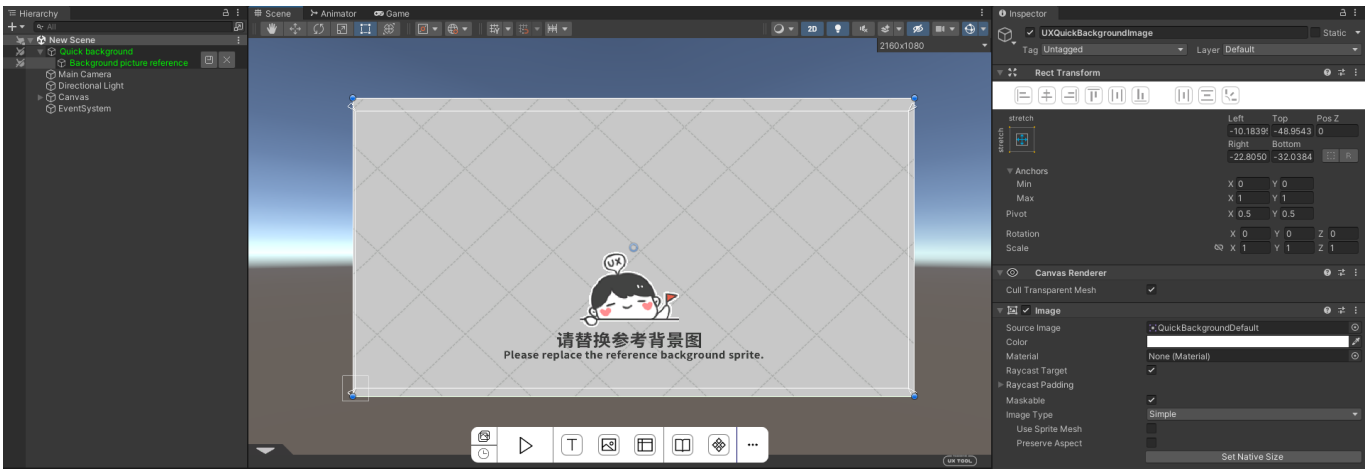
Open via ToolBar : Enable ToolBar via the Unity top drop-down menu [ThunderFireUXTool-工具栏 (ToolBar)], and click the [Quick background] button on the left to open the background image.

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



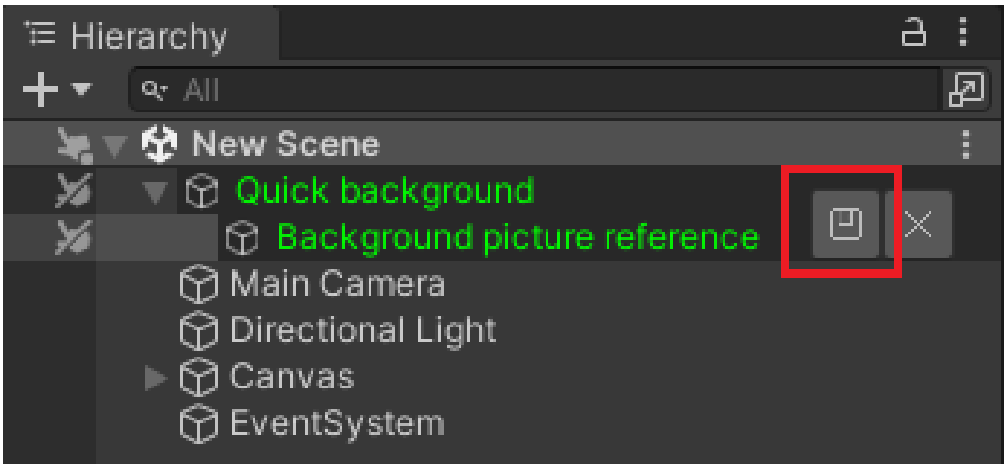
Quick background image in Scene

Settings You can select "Reference Background Image" in the Hierarchy panel, and adjust the size and position information of the image or change the image in the Scene or Project panel, etc.

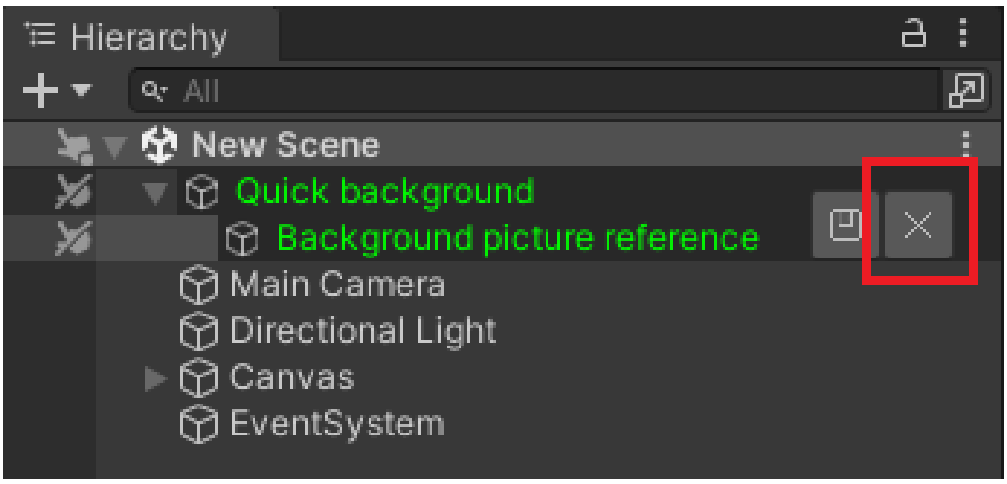


Note: Images cannot be selected directly in the Scene.

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.

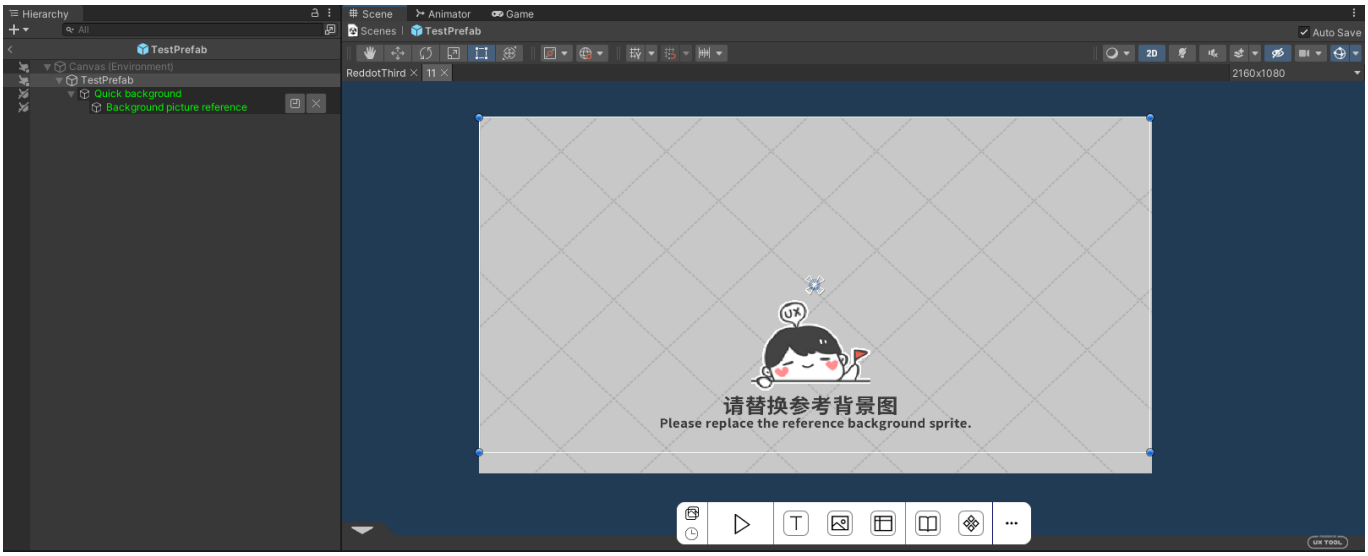


Note: When starting a run (Play mode), if the reference background is still present, the reference background will be destroyed thus not affecting the scene run; when exiting a run (Play mode), the reference background

will be turned back on until it is turned off.

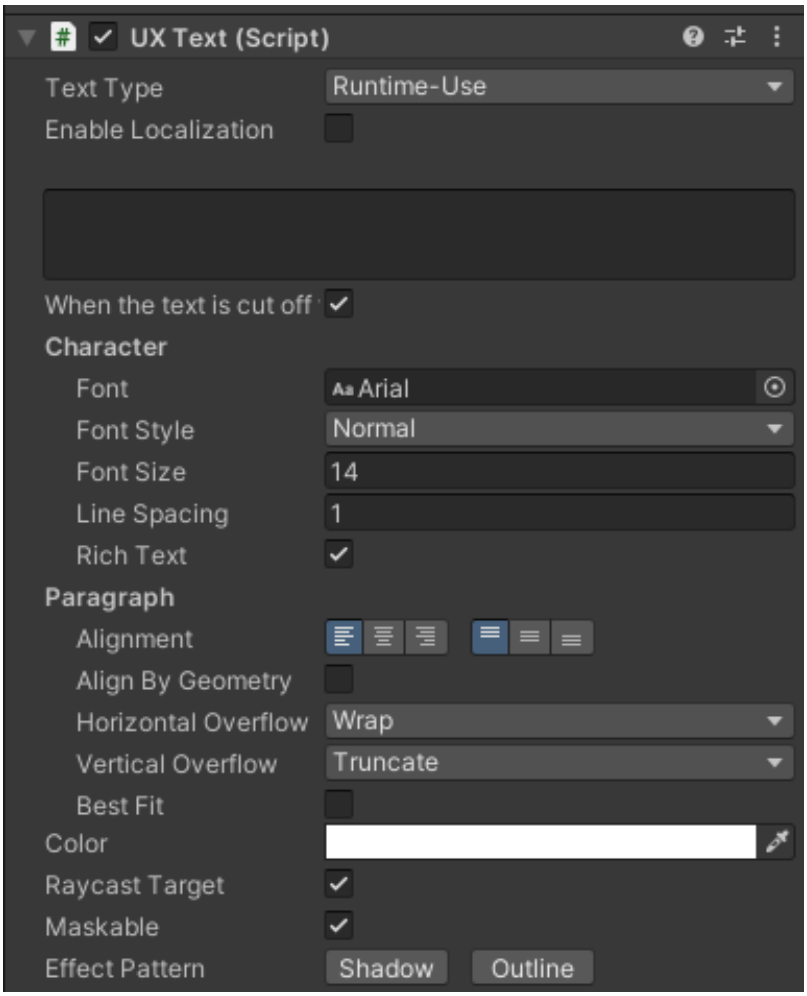
Quick background image in Prefab

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.



Use path

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

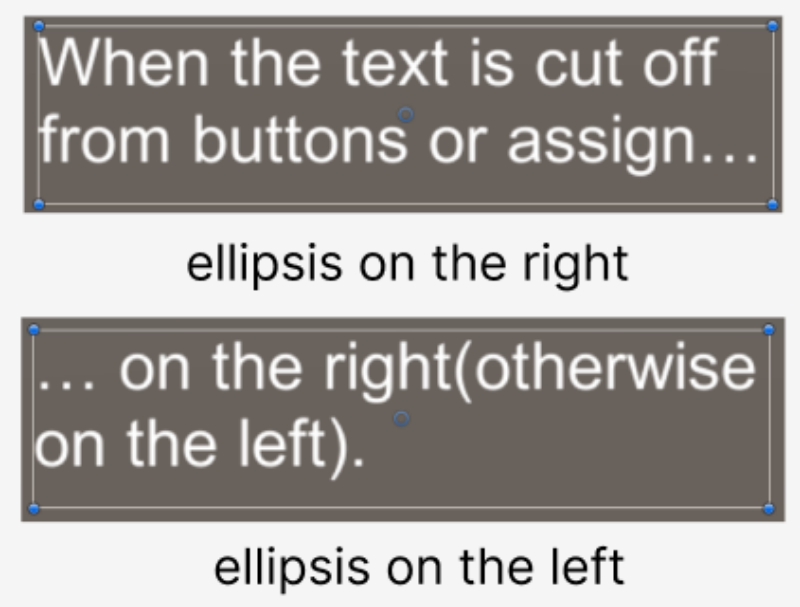
Menu bar Click [GameObject->UI->UXText] in the menu bar to create a UXText as well.

Text Localization

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

Hyperbox Omission

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.

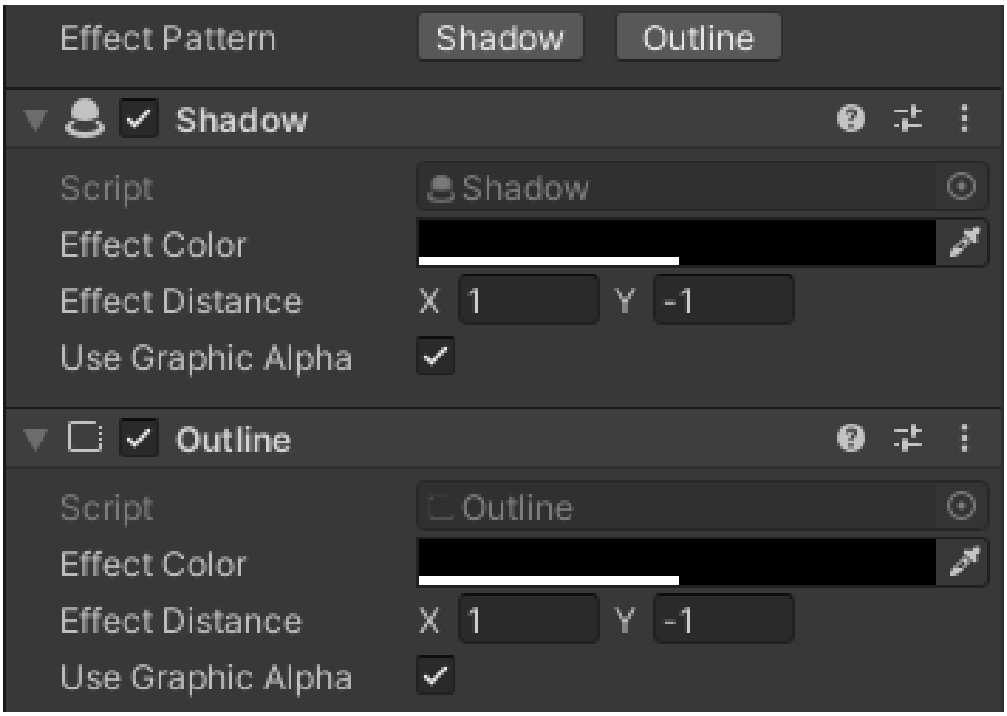


BestFit Adaptive Text Box

Checking Adaptive Text Box will automatically adapt the size of the text box so that it fills the entire text box. If Adaptive Text Box is not checked, it will take priority to fill the horizontal size until the text size reaches the minimum value.

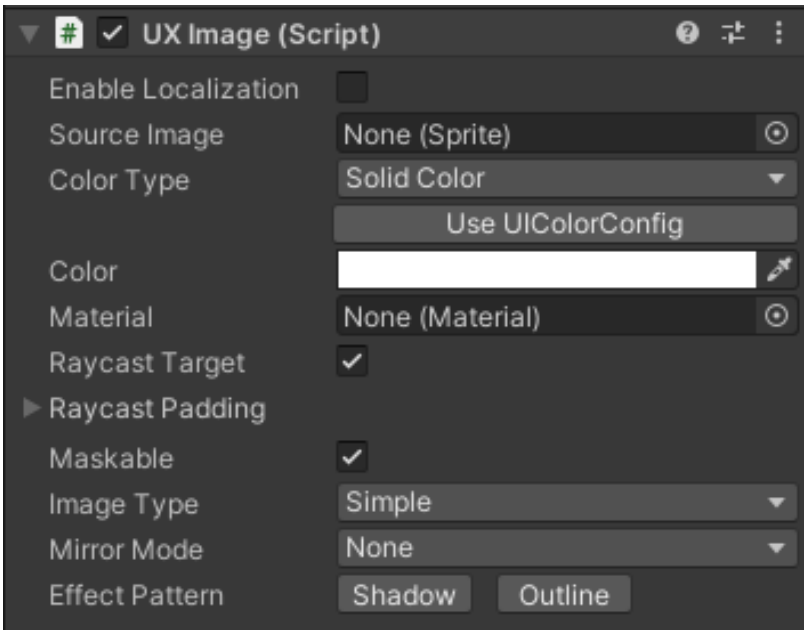
Effect Styles

Shadow and Outline components can be added quickly by clicking on the Shadow and Outline buttons to add styles to the text.



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.

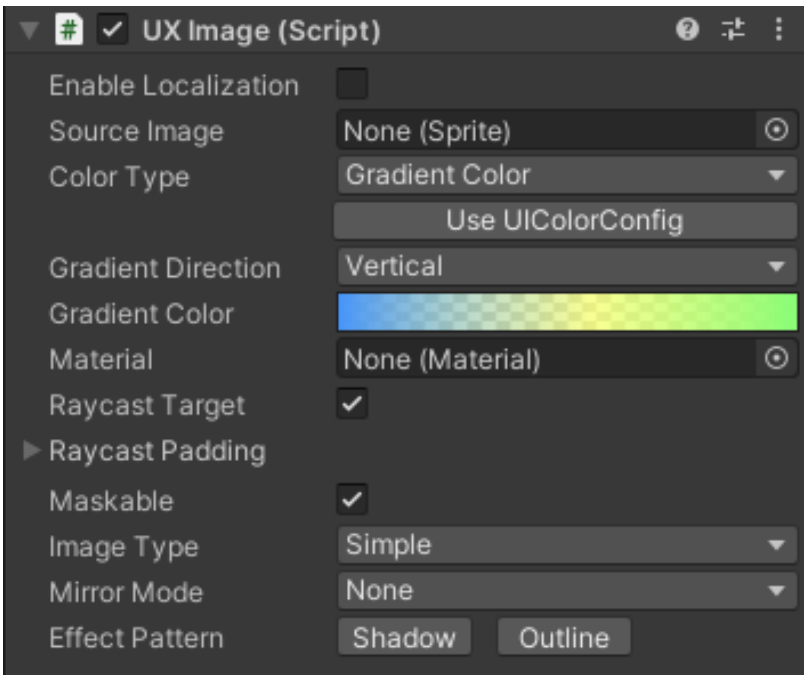
Menu bar Click [GameObject->UI->UXImage] in the menu bar, you can also create a UXImage.

Image Localization

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

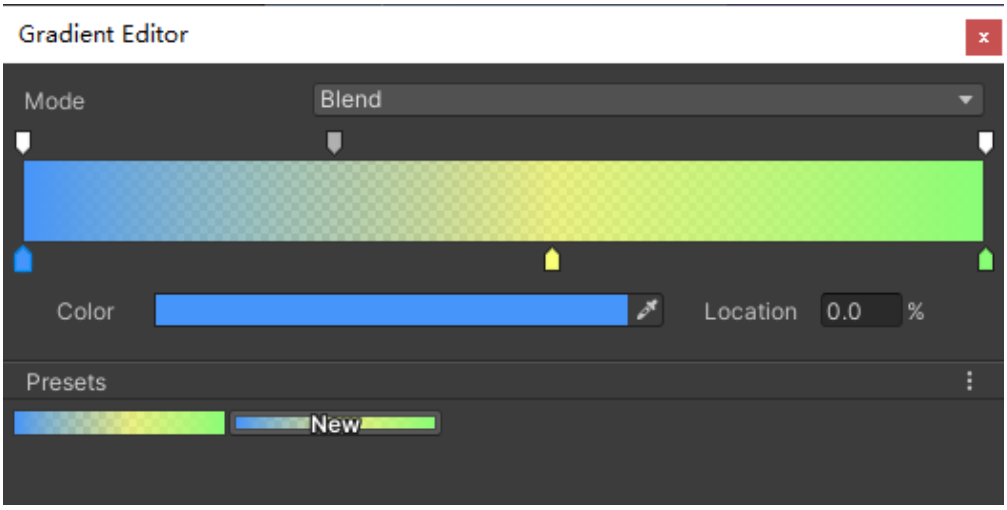
Gradient color

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.

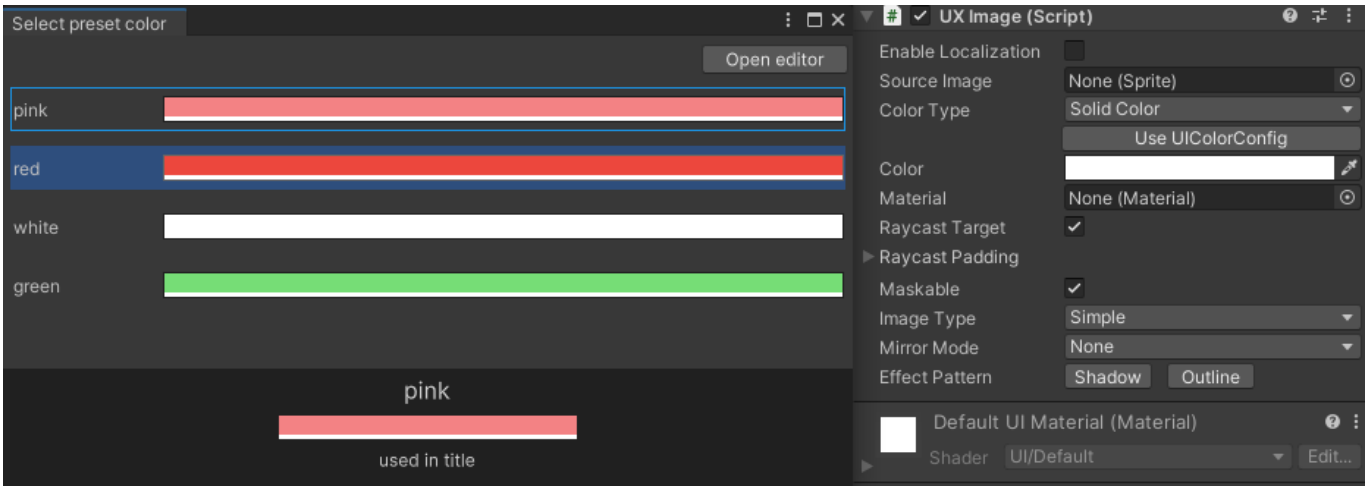
Color scale located at the top and bottom of the gradient color bar, used to indicate the key color used in the gradient, every two adjacent color scale between the smooth and uniform composition of the gradient. Click the upper and lower area of the color bar to add a new color marker, select and press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

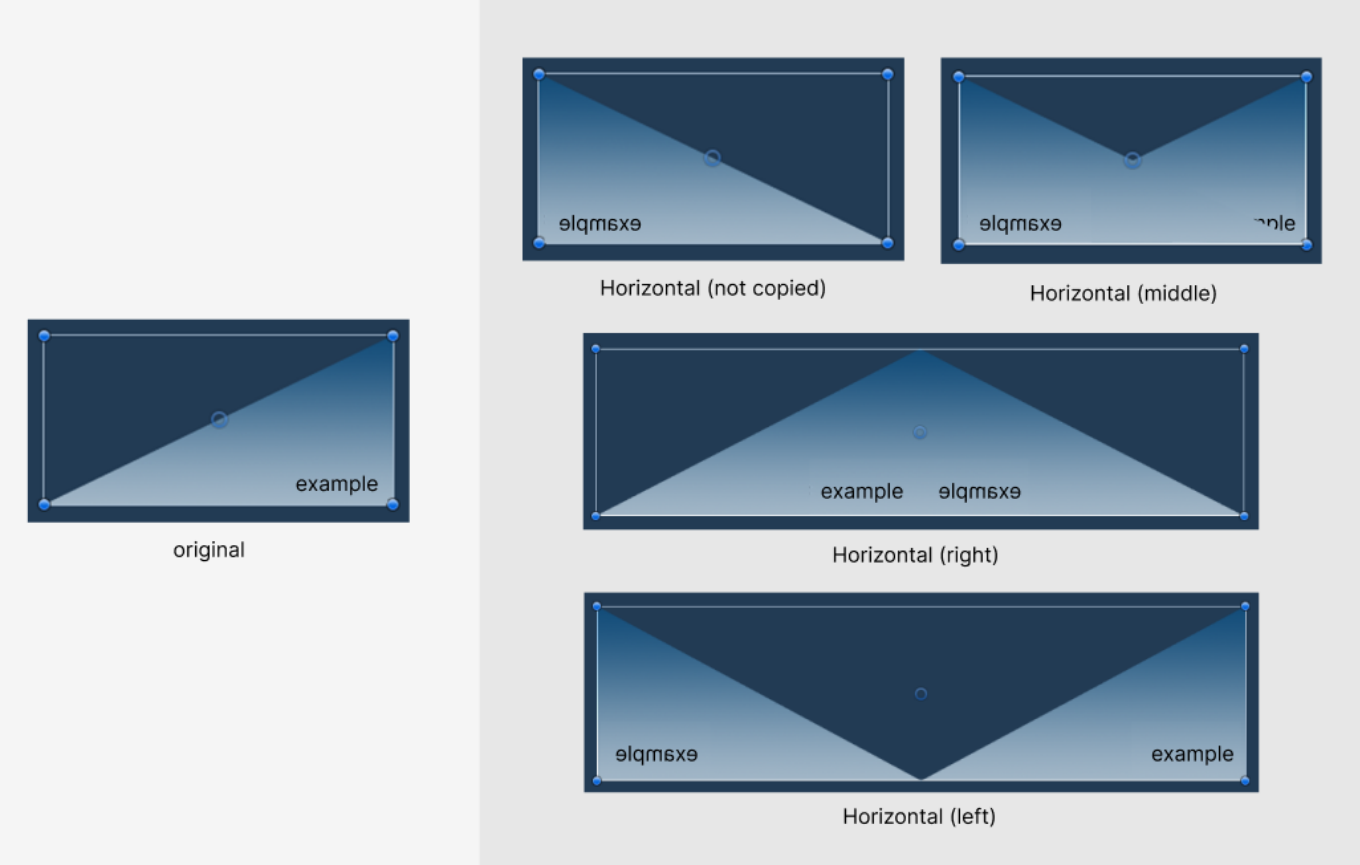


Mirroring

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

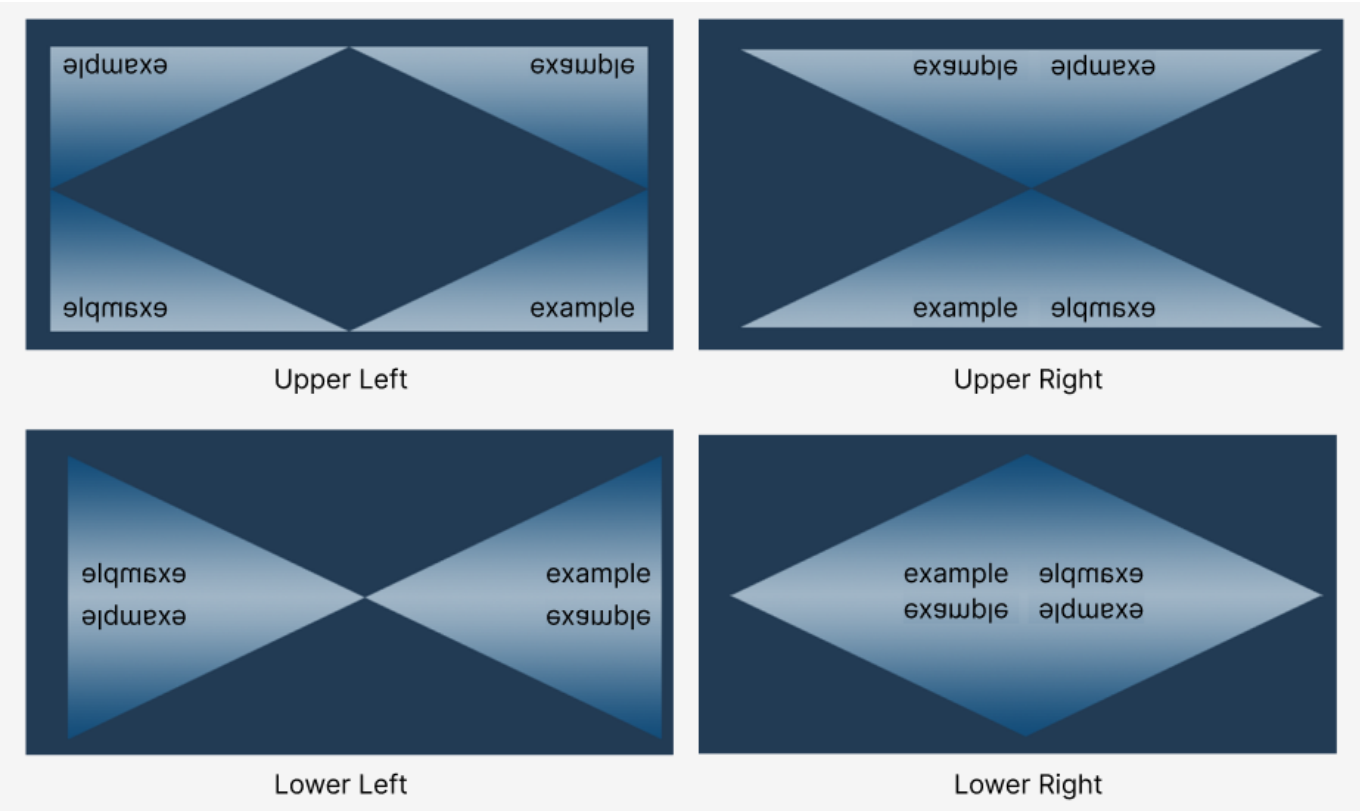
Horizontal Mirroring and Vertical Mirroring

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.



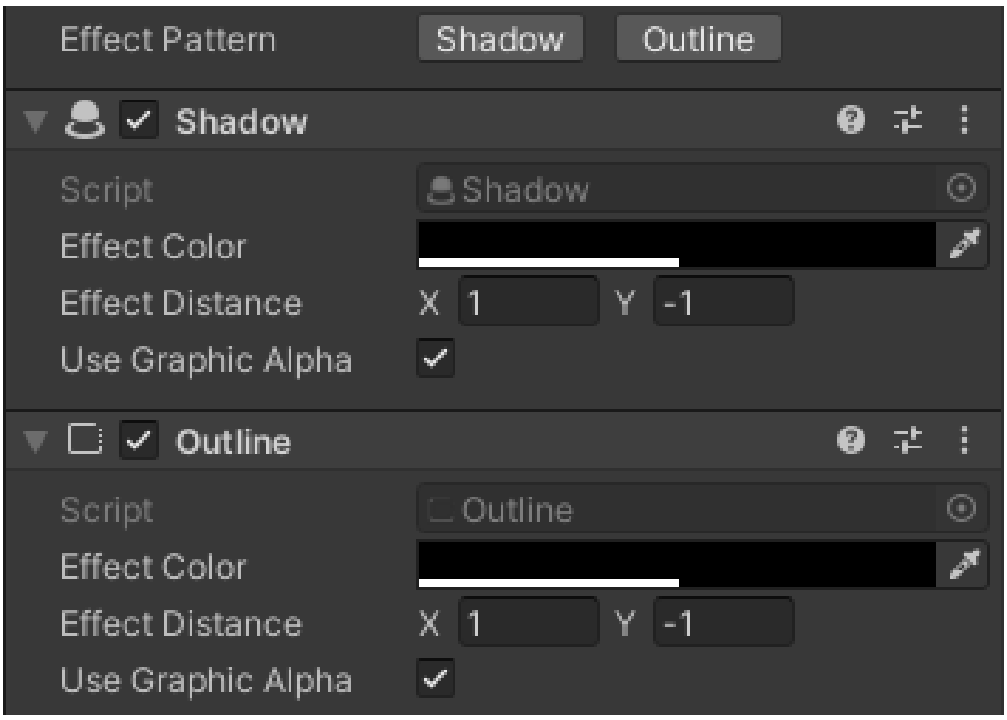
Surrounding Mirroring

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

Shadow and Outline components can be added quickly by clicking the Shadow and Outline buttons to add style to the image.



UXScrollView

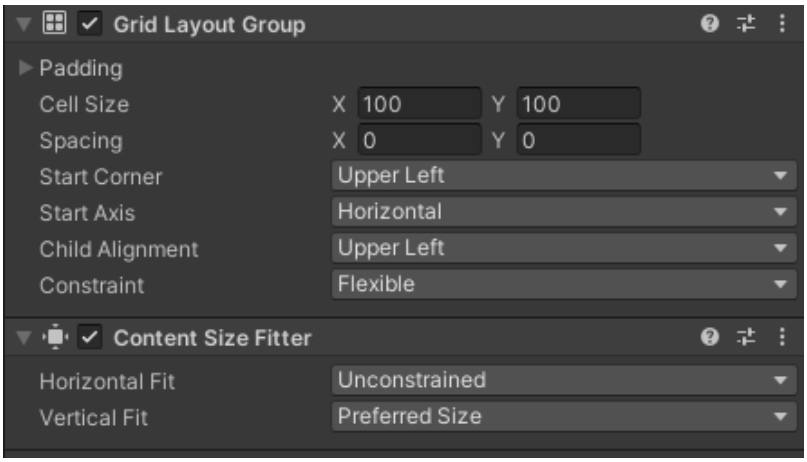
ThunderFire UX Tool extends Unity's original ScrollView in the UXScrollView. Compared to ScrollView, UXScrollView mainly provides **grid element preview** and **automatic component addition**.

Use path

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.

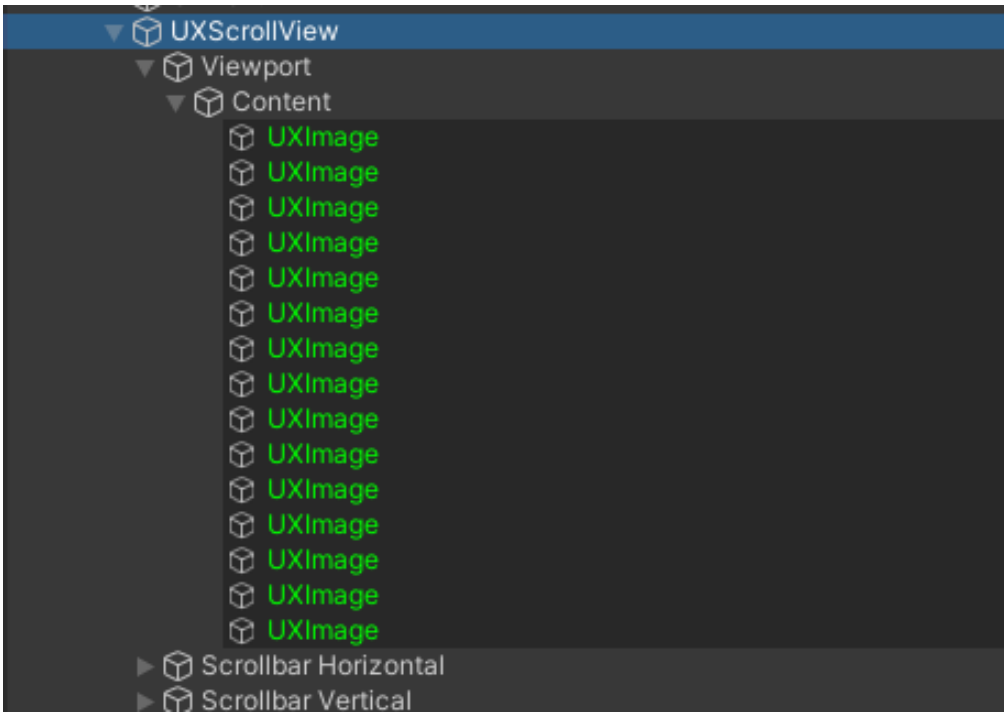
Grid count the number of grid elements to be previewed, default is 0, i.e. no preview is enabled.

Grid layout the arrangement of the grid elements in Viewpoint, there are grid layout, horizontal layout and vertical layout.

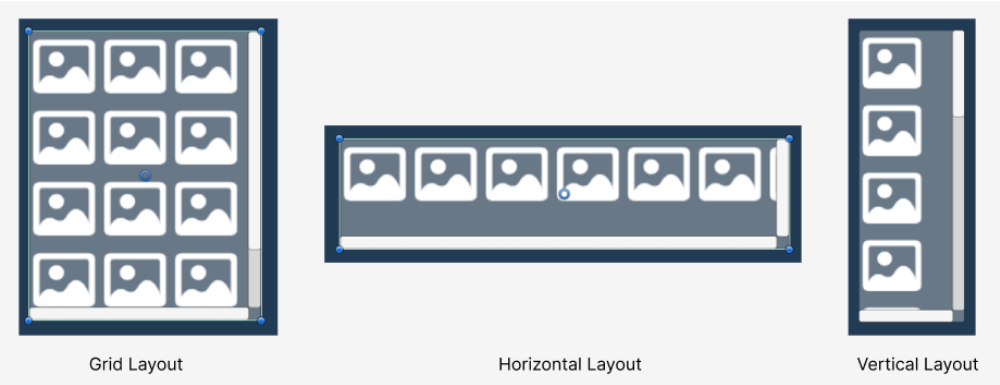
Preview effect

The preview effect can be seen in Hierarchy and Scene by modifying the grid properties.

In Hierarchy



In Scene



Color and Gradient Configuration

ThunderFire UX Tool provides a quick way to configure colors and gradients, using the configured color or gradient preset directly in the Inspector panel or in code.

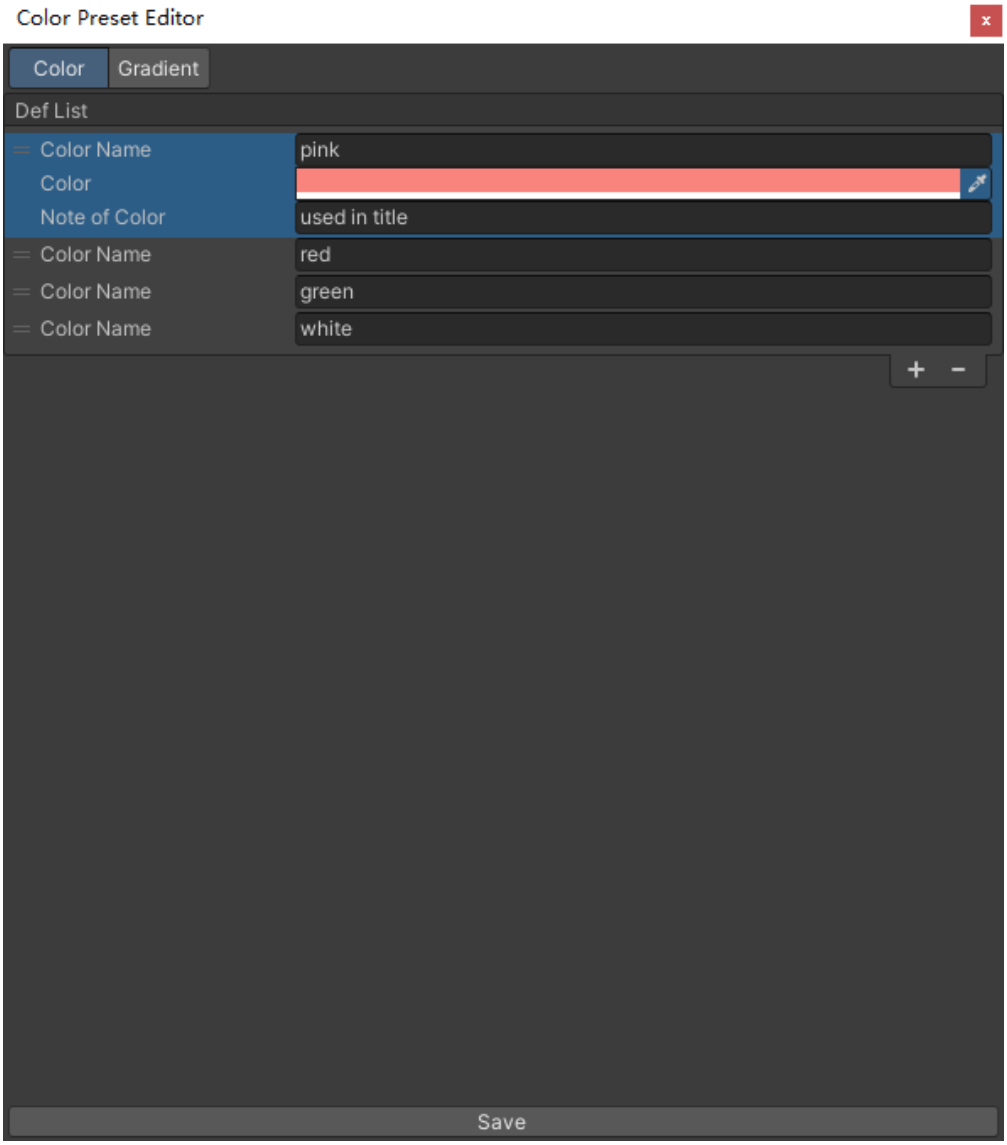
Use path

Menu bar Select the menu bar [ThunderFireUXTool->新建配置文件 (Create Assets)->UIColorAsset or UIGradientAsset], you can create a new color or gradient configuration file, and then click [ThunderFireUXTool->颜色与渐变配置 (UIColorConfig)], you can enter the following configuration interface (color configuration as an example, the same gradient).

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.

Click Save below to save the data to UIColorGenDef.cs, you can view the generated code file and call the generated color configuration in the code. The gradient configuration is then generated into the

UIGradientGenDef.cs file. (Refer to the **program interface** below for file format details)

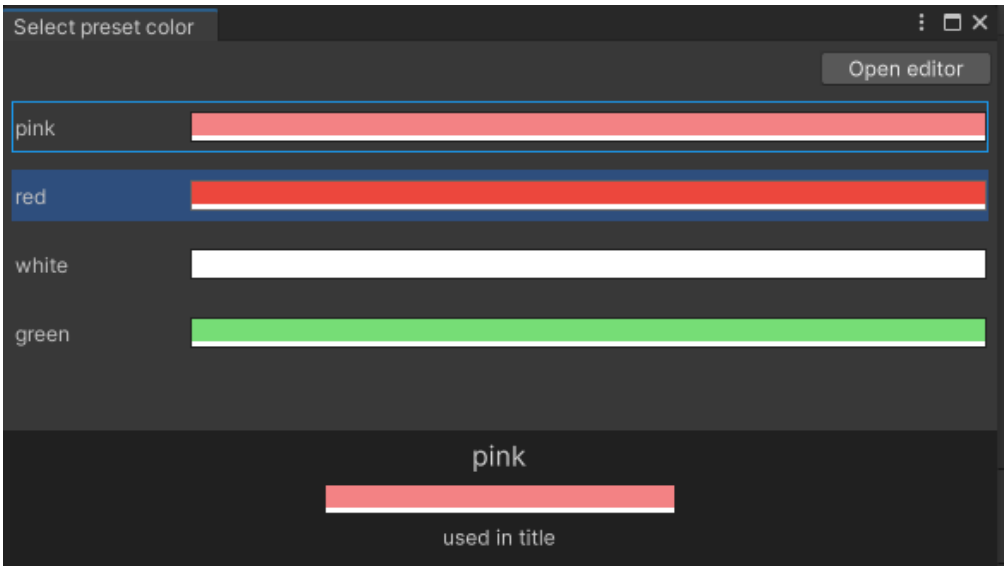


Reference configuration in UXImage component

Clicking on the Use UIColorConfig button in UXImage will bring you to the following screen.

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 Management Tools

The hierarchy management tool in ThunderFire UX Tool is designed to visually manage the hierarchy of Prefab in a project. Existing hierarchy management suffers from the following problems.

- Extremely complex interface hierarchies when the number of interfaces is too large.
- Difficulties in communication between front and back managers.

The tool hopes to solve the above problems through visual and direct representation, thus reducing management costs. Based on this expectation, the tool contains the following main features.

Use path

Open via menu Select [ThunderFireUXTool->层级管理工具 (Hierarchy Manage)] in the menu.

Hierarchical Panel Structure

As shown in the figure below, the hierarchical management tool is divided into three levels, namely **the first level** of channels represented by "HUD, interface, pop-up", **the second level** represented by numbers, and **the third level** of Prefab represented by gray squares, where the number of channels and the number of levels contained in each channel are controlled by the program code. (Refer to the **program interface** below for file format details)

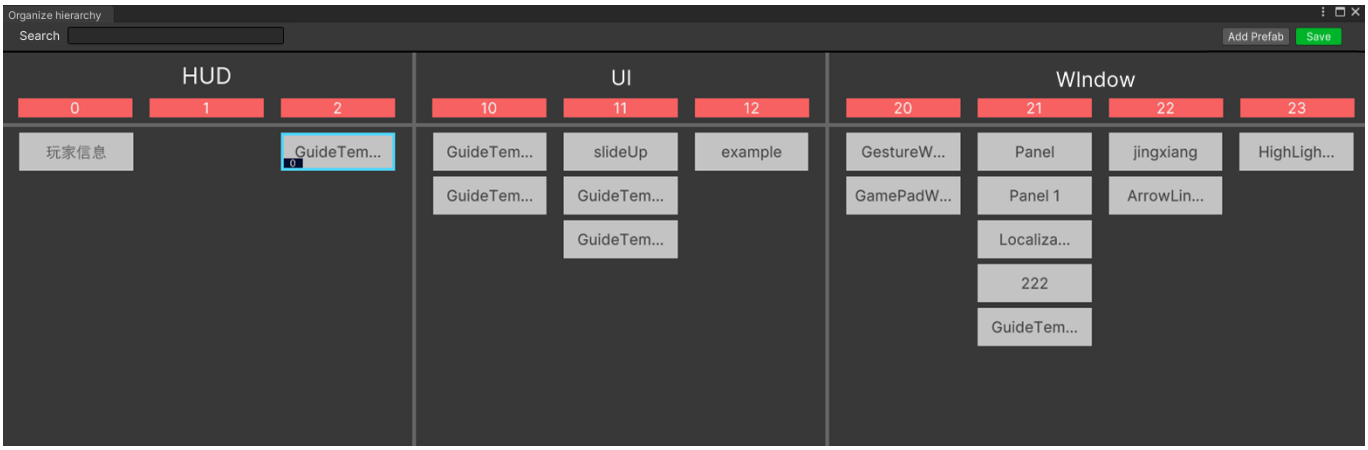
the first level : channels divide all levels into major categories, the number of levels in each category is set by the project, the default is 3, the names are "HUD", "interface", "pop-up", if there is If you have any adjustment requirements, you can contact your program development colleagues to modify them in the code.

the second level : hierarchies indicate the hierarchical rendering relationship at runtime, 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, rendered according to the level it is in, the later the Prefab in the same level, the closer to the top it is loaded.

prefab tab information The tabs in Prefab are intended to keep track of the associated Prefab for easy management later. The tabs, when added, generate fixed data about where the current Prefab ranks among all Prefabs containing this tab. In the window, **different tabs can have different colors**. The tab color is configured in the HierarchyManagementSetting.asset file and will default to black if it is added for the first time.

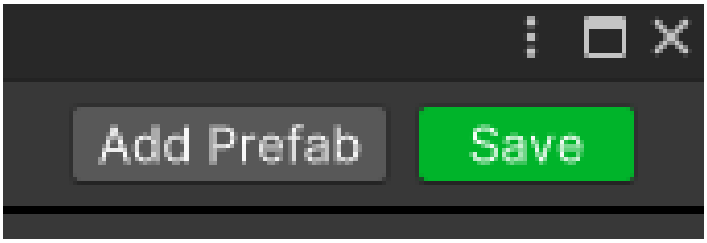
Note: This data **does not change with the Prefab hierarchy**, this is to remind the user that the order of the tabs is preset and cannot be changed at will, thus going to correspond to change the Prefab preceding or following the Prefab.

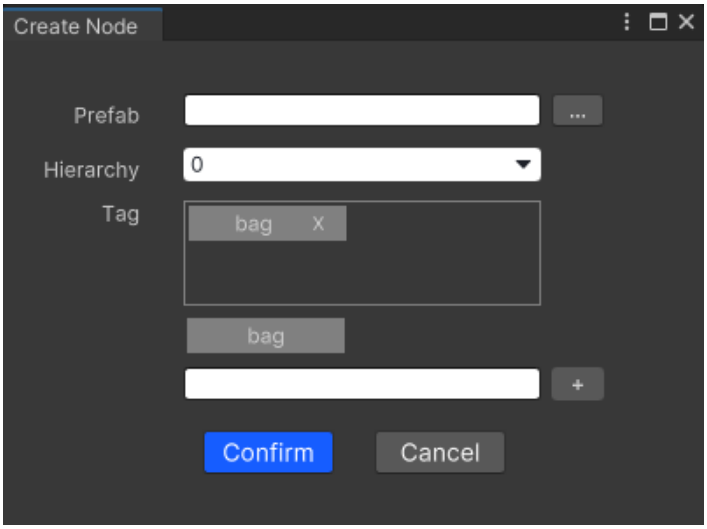


Interface functions

Add Prefab

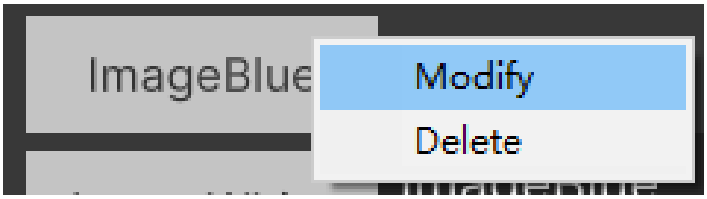
Click "Add Prefab" in the upper-right corner of the window to bring up the Add window. You can select a prefab by path and choose an existing layer for it.





Prefab information modification and deletion

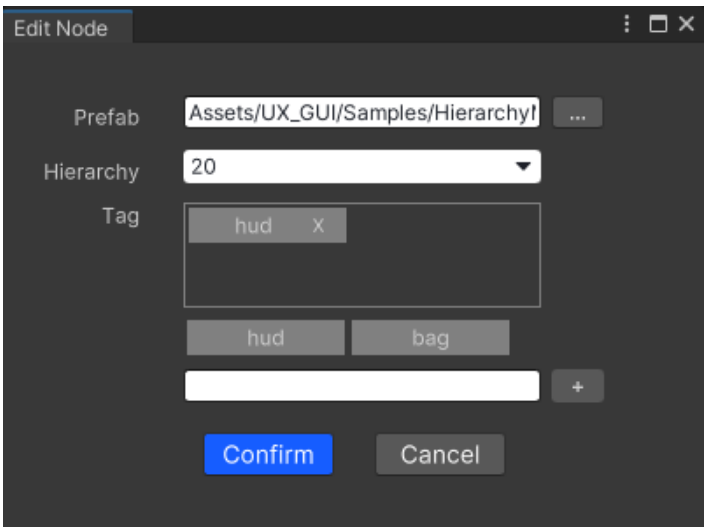
- Modify: Right-click on the Prefab and select Modify Information to bring up a modification window similar to Add.
- Delete: Right click on the Prefab and select Delete to delete the Prefab from the management window.



Tag addition and deletion

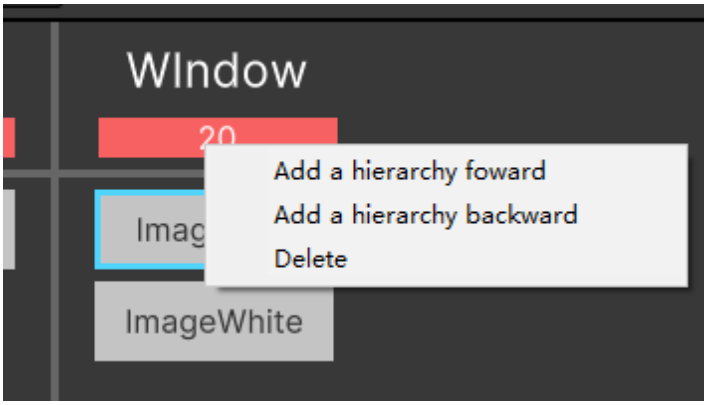
- Add: In the modify or add window, there is a tag editing area, you can click "+" after entering in the text box to add, it will be displayed in the upper box line area, and in the middle of the two will show the two recently added tags, you can directly add to the tag list after clicking.
- Delete: Click "×" in the box line area, you can delete.

Note: The above operation can only be saved after clicking OK.



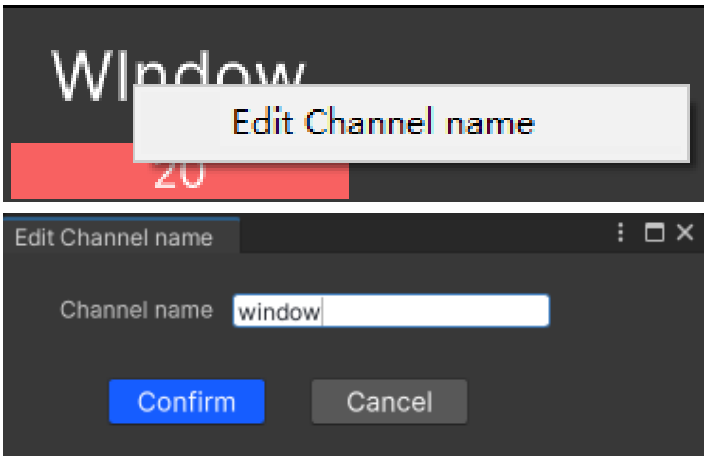
Add and delete hierarchies

Select a layer to [right-click] and choose "Add a hierarchy forward" or "Add a hierarchy backward", the added hierarchies can not exceed the maximum number of hierarchies contained in the channel, otherwise there will be a prompt. Click Delete, then you can delete **the hierarchies without Prefab**, when the channel contains only one hierarchy, it can not be deleted.



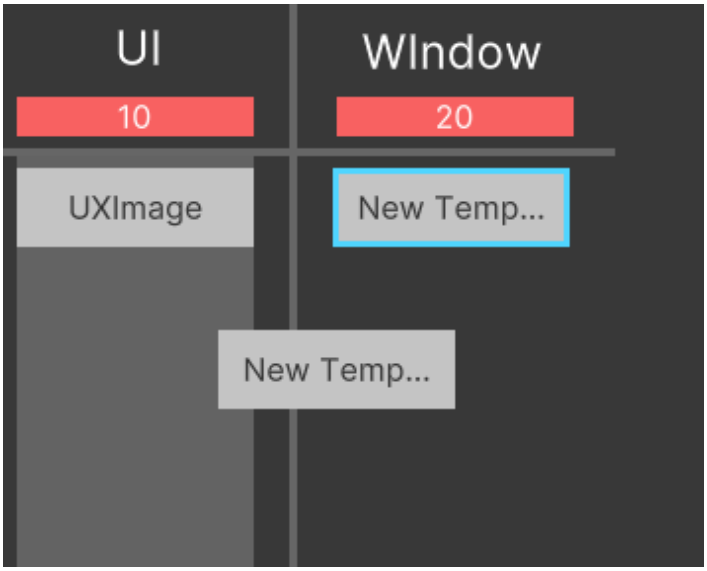
Channel Name Modification

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



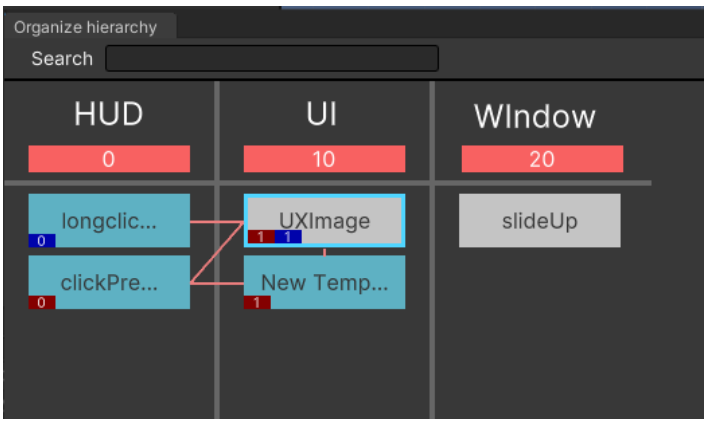
prefab drag and drop to move

After selecting the Prefab you can [drag] it, when dragging to other levels, the level will be highlighted, release the mouse to complete the level change.



prefab relationship indication

When you select a square, the square will have a blue border, and the Prefab associated with it will be highlighted and connected in order according to the label value relationship. (**Associated** is defined as having the same label as the selected Prefab, or having the same Prefab as the Prefab "with the same label as the selected Prefab", which can be interpreted as a grid spread)

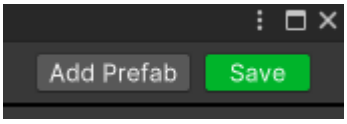


Quick search (fuzzy search)

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

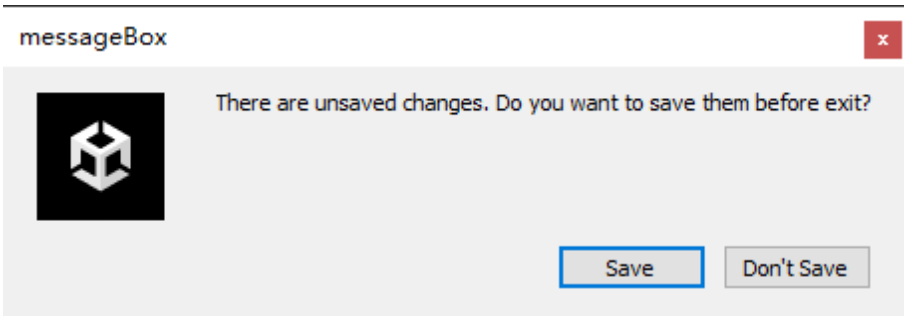
Save Information

Clicking Save in the upper right corner will store tab information, Prefab information, Channel name, number of levels, and other information that can be modified by the interface.



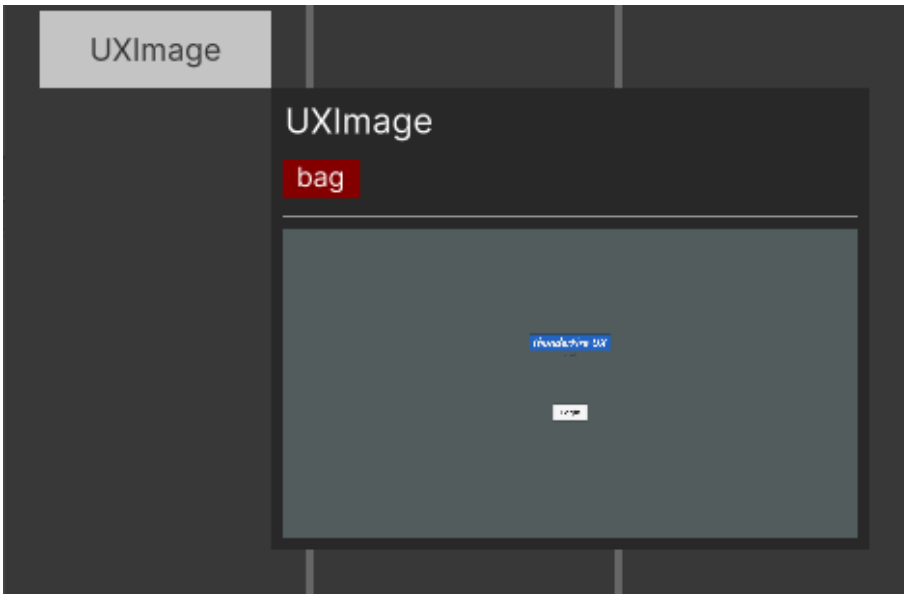
When the page is modified, if it is not saved before closing, a pop-up box will appear.

Note: If you directly click the "×" in the upper right corner of the pop-up window, no saving will be done and the administration tool will be closed.



prefab preview

When the mouse moves over a Prefab block, a hover window displays the name, tabs, and thumbnails 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's Guide

ThunderFire UX Tool provides a complete set of bootstrapping features, including bootstrapping editing tools and runtime functionality, to help interaction designers quickly style and adjust the bootstrapping interface in the Unity editor and allow engineers to quickly access the bootstrapping in their projects.

Terminology

GuideWidget A guide control that represents only one kind of guide effect, such as skeleton highlighting, gestures, guide text, etc.

GuideWidgetData Record various data in GuideWidget. It is used to save the data of the guide interface edited in the editing tool.

UIBeginnerGuide A guide interface template with a variety of different GuideWidgets on the interface.

UIBeginnerGuideData record the various data in UIBeginnerGuide. Including guide ID, guide type, guide duration, guide template, and GuideWidgetData contained in UIBeginnerGuide (refer to the **parameter**

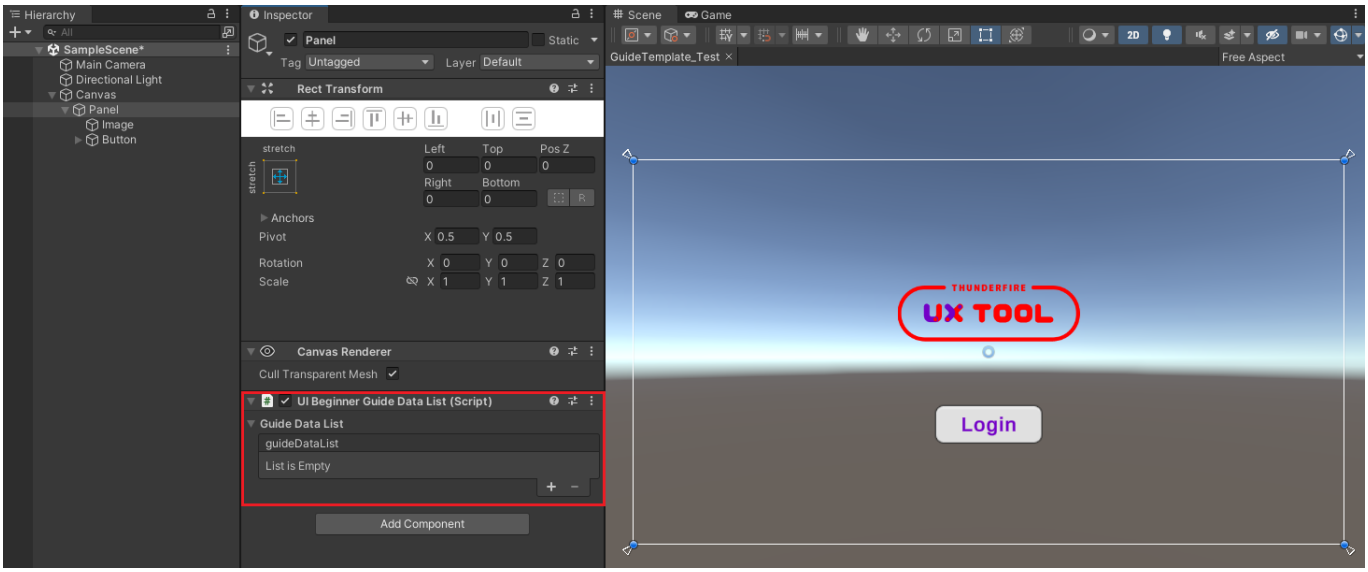
details below for details).

UIBeginnerGuideDataList A guide list containing a set of UIBeginnerGuideData. When a guide in the list is completed, the next guide in the list will be opened automatically until the whole guide list is completed.

How to use

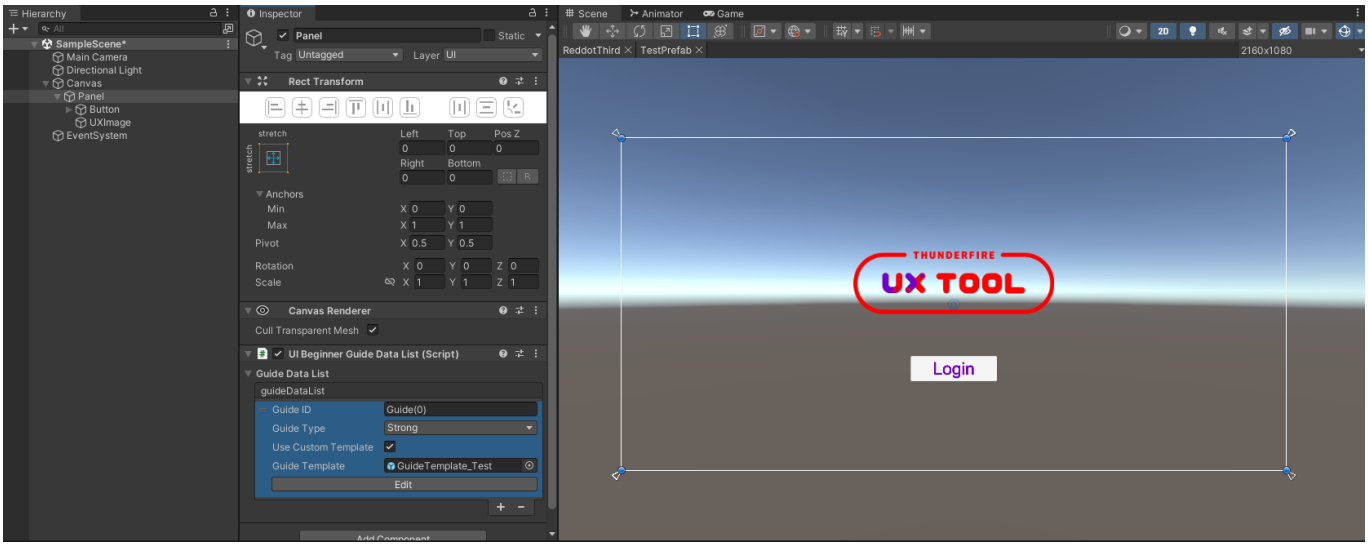
1.Create the boot sequence.

Select a node in the scene or Prefab (usually choose the resident node), click [ThunderFireUXTool-创建新手引导 (Create BeginnerGuide)] at the top, a UIBeginnerGuideDataList component will be added to the node, i.e. a guide sequence (you can also add this component manually in the Inspector panel manually).



2.Create a new boot in the sequence.

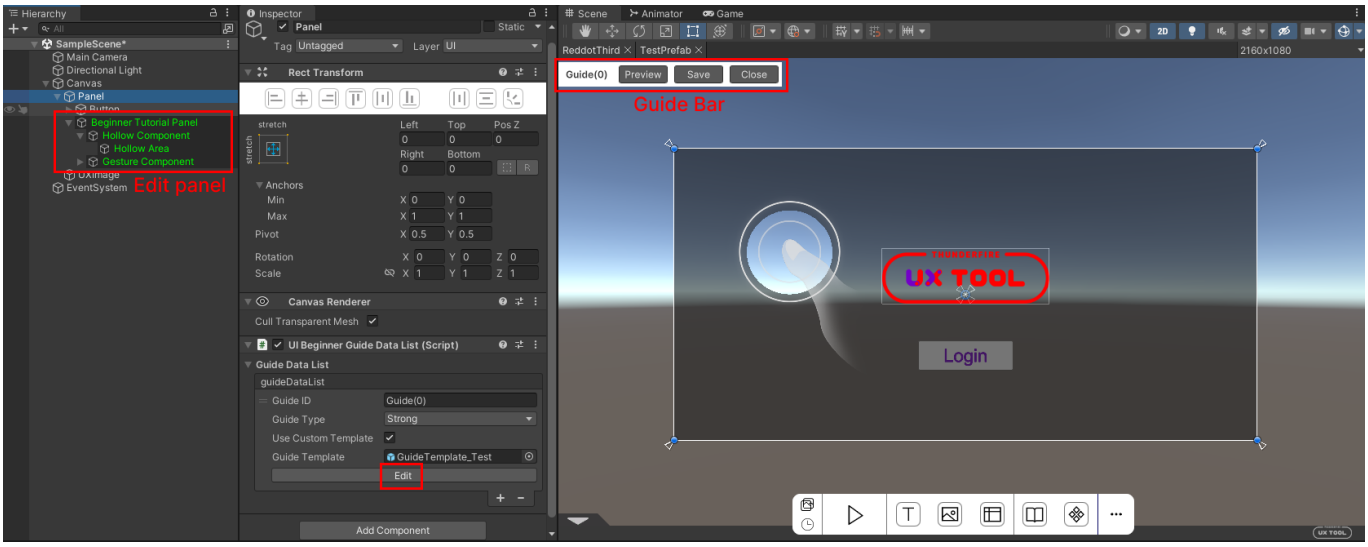
The default sequence is an empty list, click the "+" sign to create a new guide. Configure a boot template for this boot, either a pre-defined template or a custom template, here a custom template is used. (See **Creating a custom boot template** below)



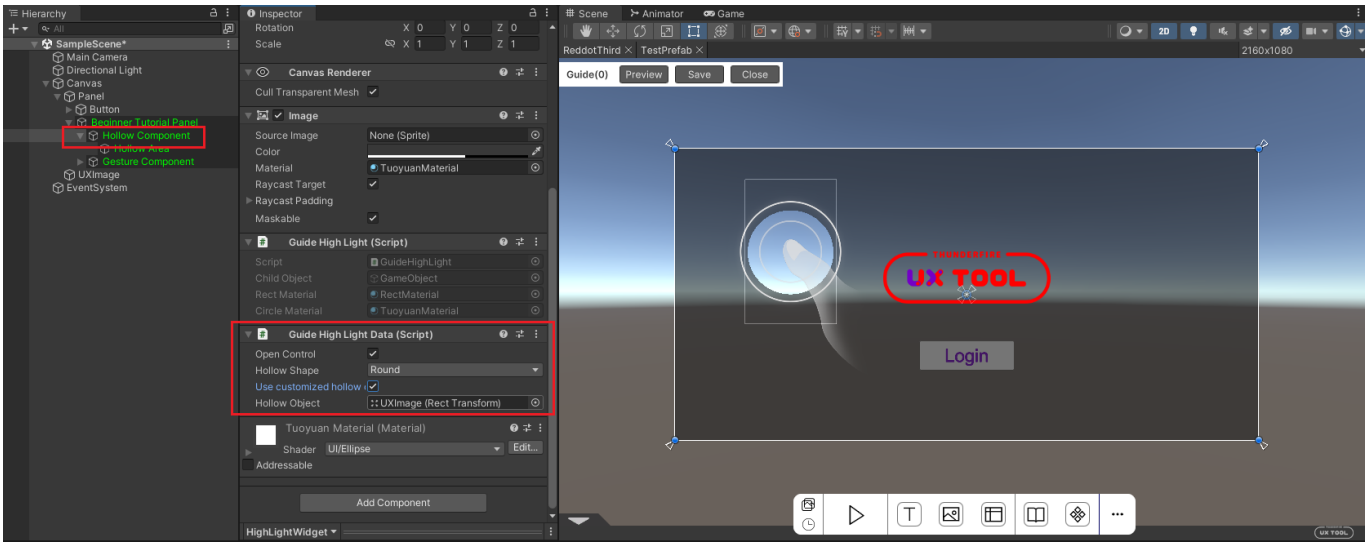
3.Editorial guidance.

Click the [Edit] button, a bootstrap editing panel will be created in the Hierarchy window, and the editable objects in the panel are displayed with green names. At the same time, the bootstrap toolbar will be displayed in the top left of the Scene window, showing the bootstrap ID, Preview, Save, and Close buttons in order.

The bootstrap editing panel is a temporary object, which is an instantiation of an object in the "bootstrap template" and is only intended to provide visual bootstrap editing functionality. It is deleted after switching scenes, starting a run or clicking the close button.

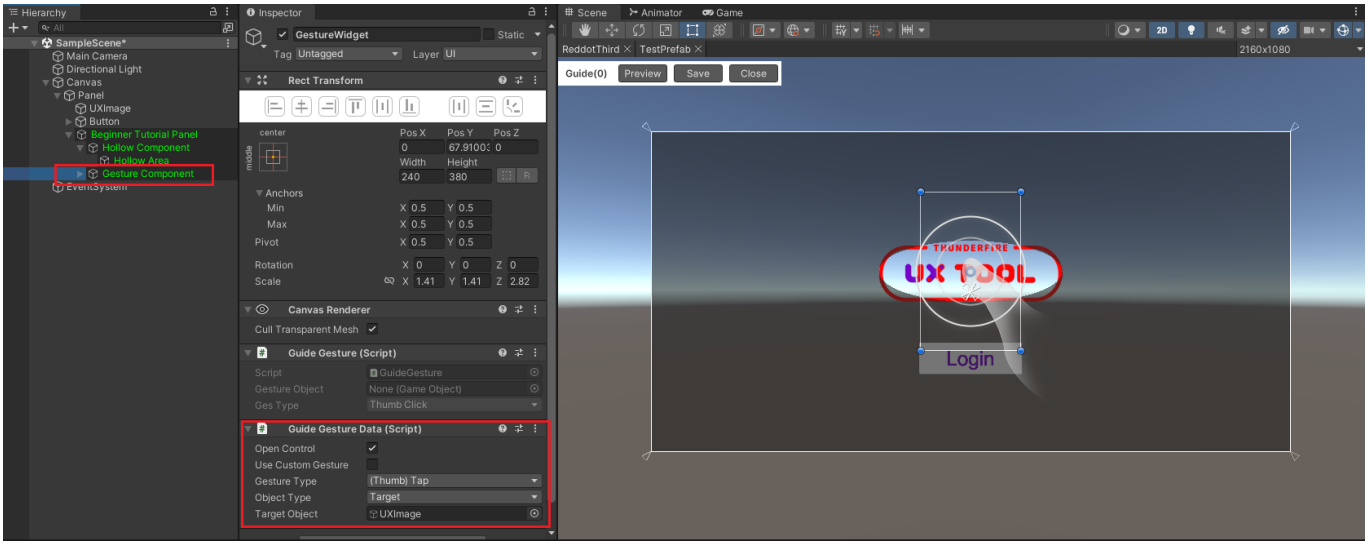


Edit the skeleton highlighting object. Click on the "Skeleton Component" object in the Hierarchy window and configure it in the GuideHighLightData component as shown.



The above operation is to select "Circular Skeleton" and the skeleton object is "UXImage" (i.e. the UX Tool image in the figure). After finishing editing, click the [Save] button in the toolbar to save.

Edit the gesture component. Open the guide editing screen, select the "Gesture component" object and configure it in the GuideGestureData component as shown in the figure.

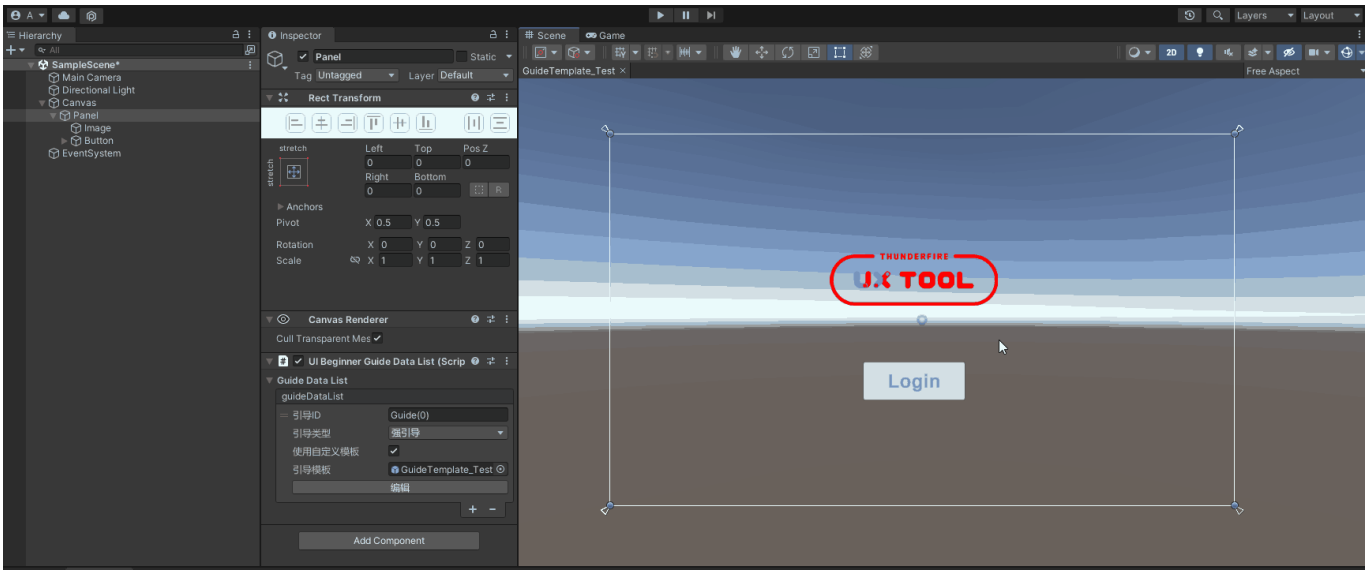


That is, add a gesture click animation effect to the UXImage (i.e. the UX Tool image in the figure) as well. After editing, click the [Save] button in the bootstrap toolbar.

4.Preview guide.

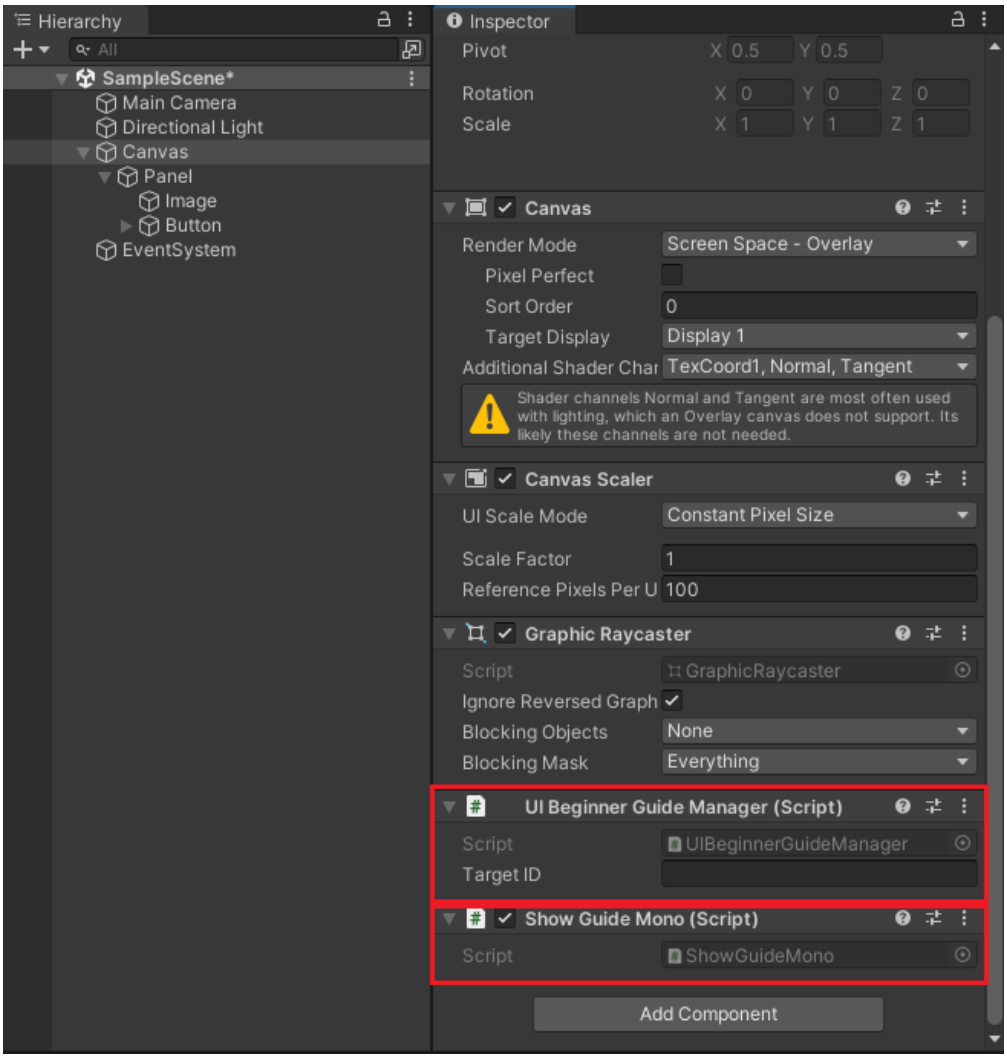
Click the Preivew button in the Toolbar to see the skeleton highlighting and gesture animation effect you just edited.

Note: Because the skeleton 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.



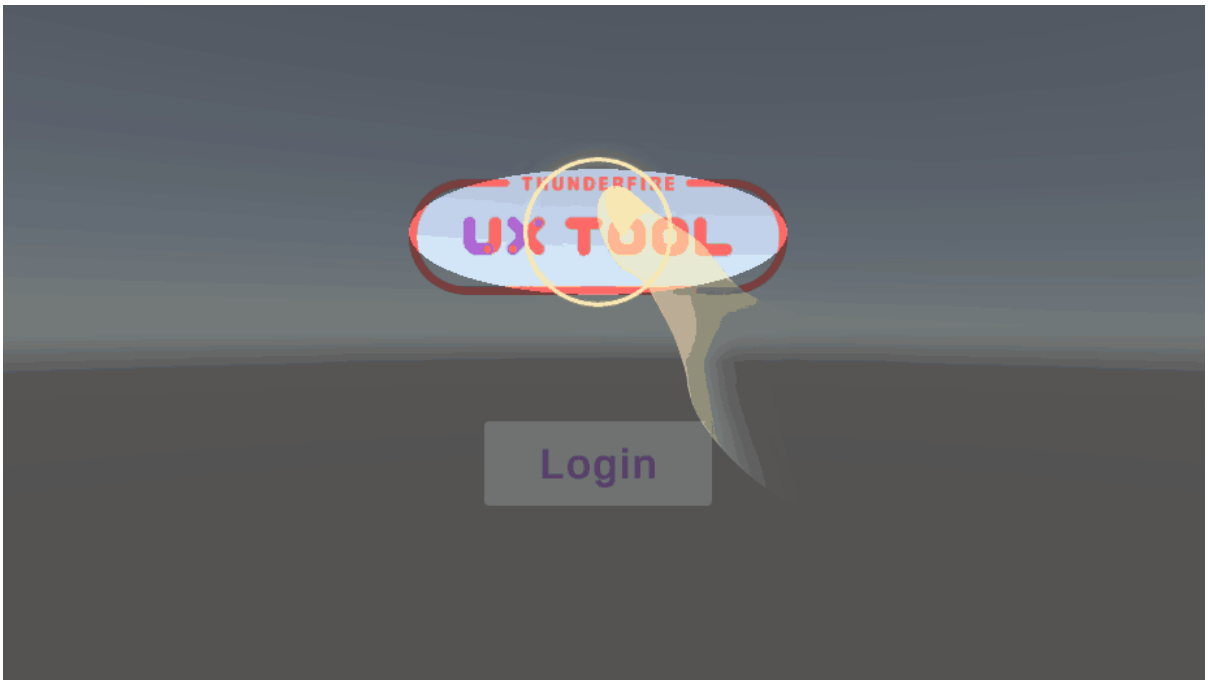
5.Running guide.

The prerequisite for proper display of guide content at runtime is the presence of the UIBeginnerGuideManager component in the scene. The UIBeginnerGuideManager is **a single instance object**, there should be only one globally.



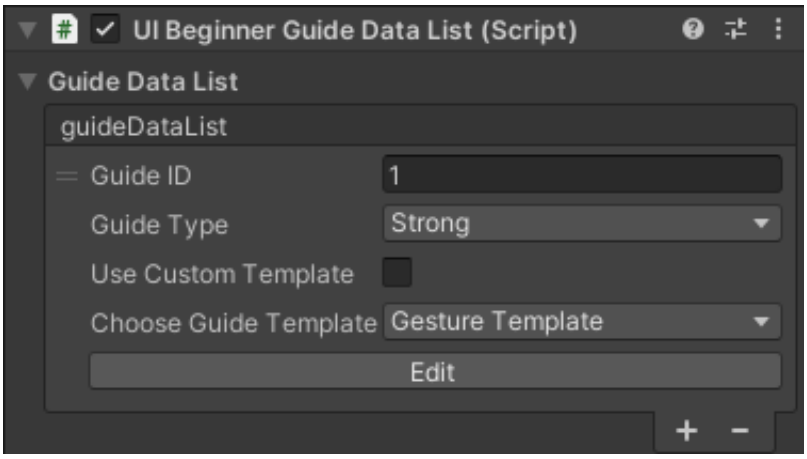
In addition to this, there is a program script that controls the timing of the playback of the newbie guide (the specific interface can be found in the program interface below). The script example ShowGuideMono in Sample performs two actions: adding the UIBeginnerGuideList to the Manager, and then playing the guide in the Manager.

The boot created in the UIbeginnerDataList component determines the boot end condition depending on the boot type and automatically opens the next boot in the group until there is no boot in that boot group.



Parameter details

Bootstrap parameters



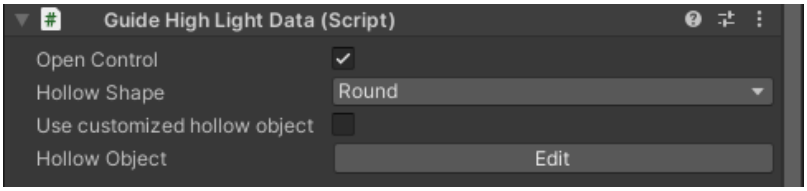
- **Boot ID**: The name used to distinguish different boot, generated by default, can be modified according to the boot content.
- **Lead type**: The type of lead ending, divided into strong, medium and weak leads.
    - Strong guide: must be used with the skeleton component, and only clicking on the skeleton area will end the guide.
    - Middle guide: Click anywhere on the screen to end the guide.
    - Weak boot: Automatically ends boot after a period of time.
- **Boot duration**: Set the duration of boot when the boot type is weak boot. Other cases do not take effect.
- **Use Custom Template**: Whether to use a custom bootstrap template. When checked, the "Guide Template" property is displayed, and when unchecked, the "Select Guide Template" drop-down box

appears.

- **Bootstrap template**: Specify a custom bootstrap template.
- **Select guide template**: Select the guide template preset provided by the tool.

Preset guide controls

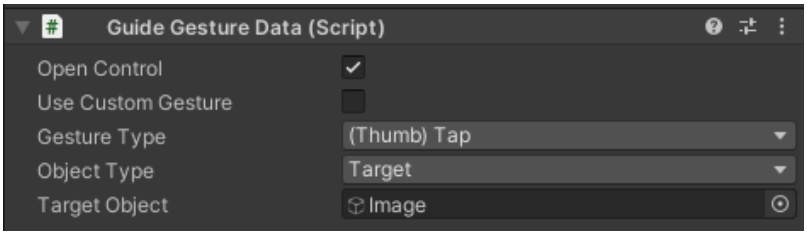
Skeleton controls



Used to add a tinted mask to a scene or interface and skeletonize the guided content for display.

- **Turn on the control**: control switch, checked by default.
- **Skeleton shape**: You can choose square or circle as the shape of the skeleton area.
- **Use custom skeleton objects**: Skeleton objects can be used by default or customized.
- **Skeleton area**: determined by whether Use custom skeleton object is checked above.
    - If not checked, use the skeleton area node that comes with the control. Click the [Edit] button of the skeleton object to edit the size and location of the skeleton area directly in the scene.
    - If checked, you can drag and drop other UI objects in the scene, such as buttons, pictures, etc., as skeleton objects, at which time the skeleton area will automatically identify the size and location of the object and be visible at runtime.

Gesture Controls

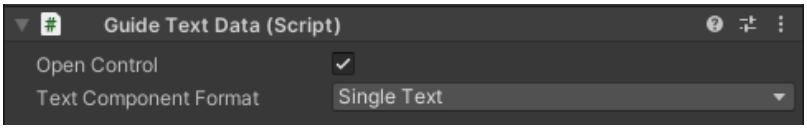


Provides commonly used gesture-guided presets with drawing and customization options.

- **Turn on the control**：control switch, checked by default.
- **Use custom gestures**: Gestures can be used by default or customized.
- **Gesture type**: determined by whether Use custom gestures is checked above.
    - If unchecked, the gesture presets provided by default are used, and the desired gesture type can be used directly by selecting it in the drop-down list.
    - If checked, you can 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, drag start marker and end marker will appear in the scene to set the start point of the drag gesture. Also adjust the drag curve to adjust the drag animation.
- **Object type**: The objects that can be guided by gestures are custom or specified. Custom can be guided by dragging the gesture to the guide position in the scene, and specified can select other objects in the scene as the guided objects.
- **Specified object**: Show this parameter when the object type is specified. You can drag and drop other UI objects in the scene as the gesture guide object, when the gesture will always follow the position of that object.

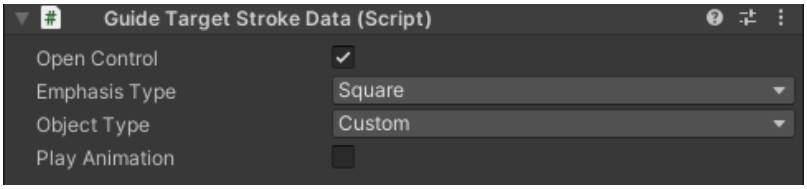
Text Control



Provide basic text objects to be used as guidance tips.

- **Turn on the control**: control switch, checked by default.
- **Text control style**: optional text control with or without caption, with two options: single text and text with caption.

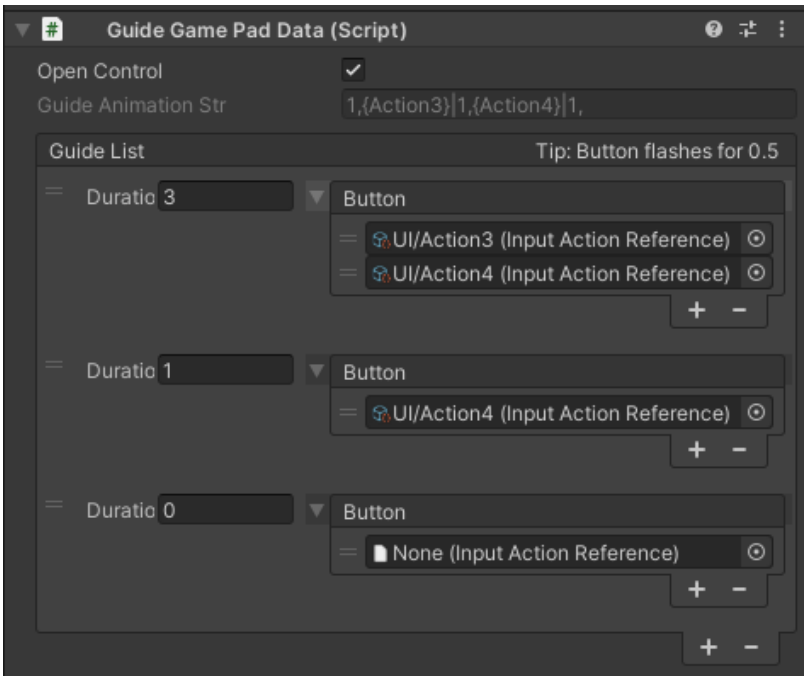
Emphasis on controls



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

- **Turn on the contro**: control switch, checked by default.
- **Emphasis type**: You can choose square or circle as the shape of the emphasis box.
- **Object type**: The object to be emphasized can be customized or specified. Customized can be done by dragging the emphasized box to the desired position in the scene, while specified can select other objects in the scene as emphasized objects.
- **Specified object**: Show this parameter when the object type is specified. You can drag and drop other UI objects in the scene as emphasis objects, when the emphasis box will always follow the position of the object.
- **Play animation**: if or not play the fade animation of the emphasis box.

Handle controls



Provides joystick key guidance.

- **Turn on the control**: control switch, checked by default.
- **Guide list**: the sequence of the handle key animation guide, click the "+" sign to add a guide, select the existing guide and click the "-" sign to delete.
    - Duration: that is, the display duration of this guide, 0.5 seconds for each flash of the key..5秒。
    - Key: the key corresponding to the guide, you can set more than one flashing key at the same time, and the default is not flashing when no key is set.

Preset guide templates

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

Gesture templates

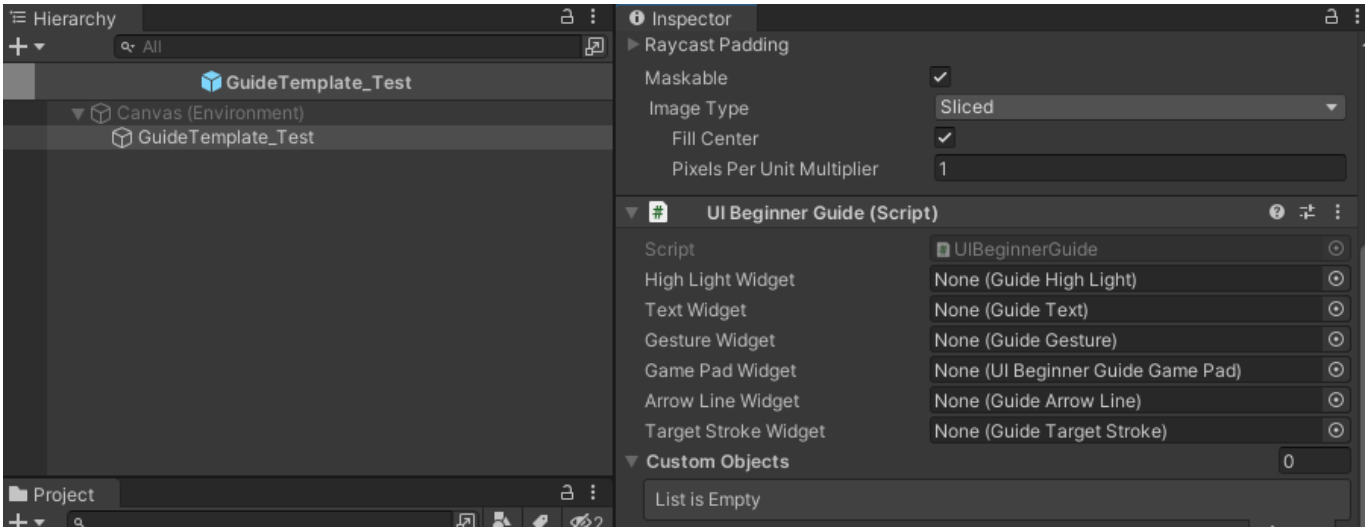
The gesture template contains skeleton controls, gesture controls, emphasis controls and text controls. When using the template, you can turn on or off various controls and adjust their parameters as needed, and save them to take effect.

Handle template

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

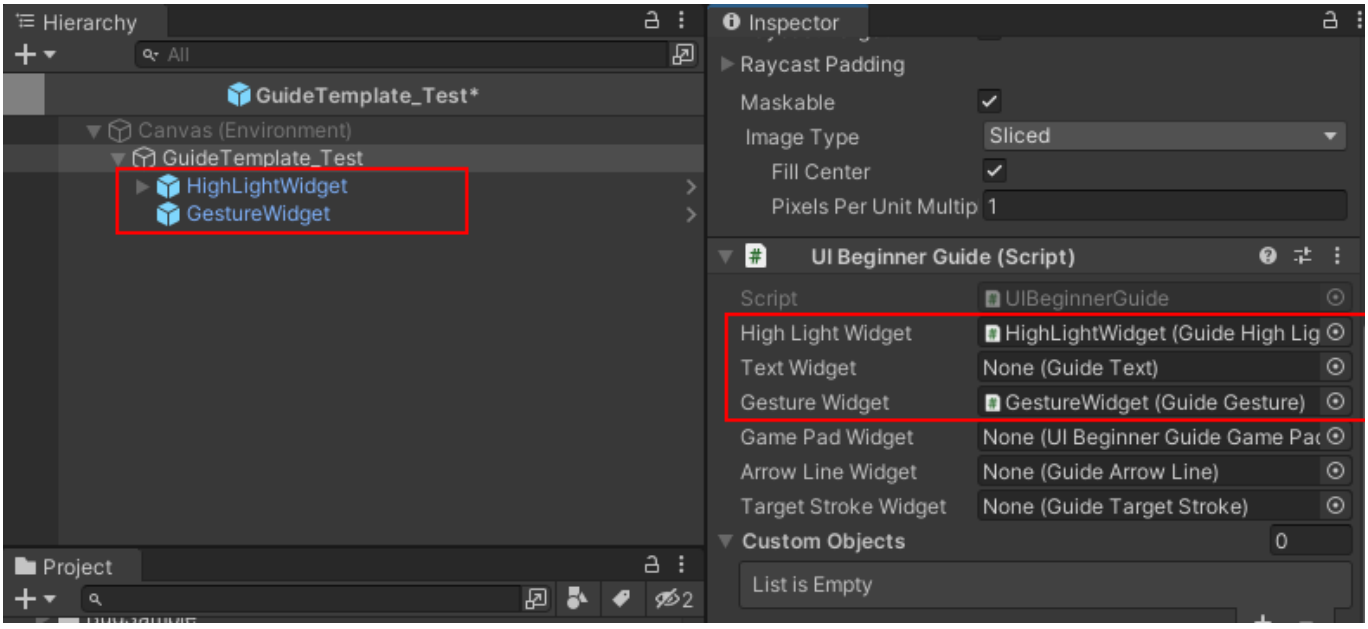
Create custom boot templates

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



Step 2 In the folder of preset controls, add the guide control as a child node of Prefab according to the requirement and bind the corresponding widget in the UIBeginnerGuide component. Add the common skeleton highlighting and gesture two kinds of guides here.

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



Step 3 Save the Prefab so that a bootstrap template is created and the rest of the usage is the same as the definition 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 Features

ThunderFire UX Tool's localization features support two categories: image localization and text localization.

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

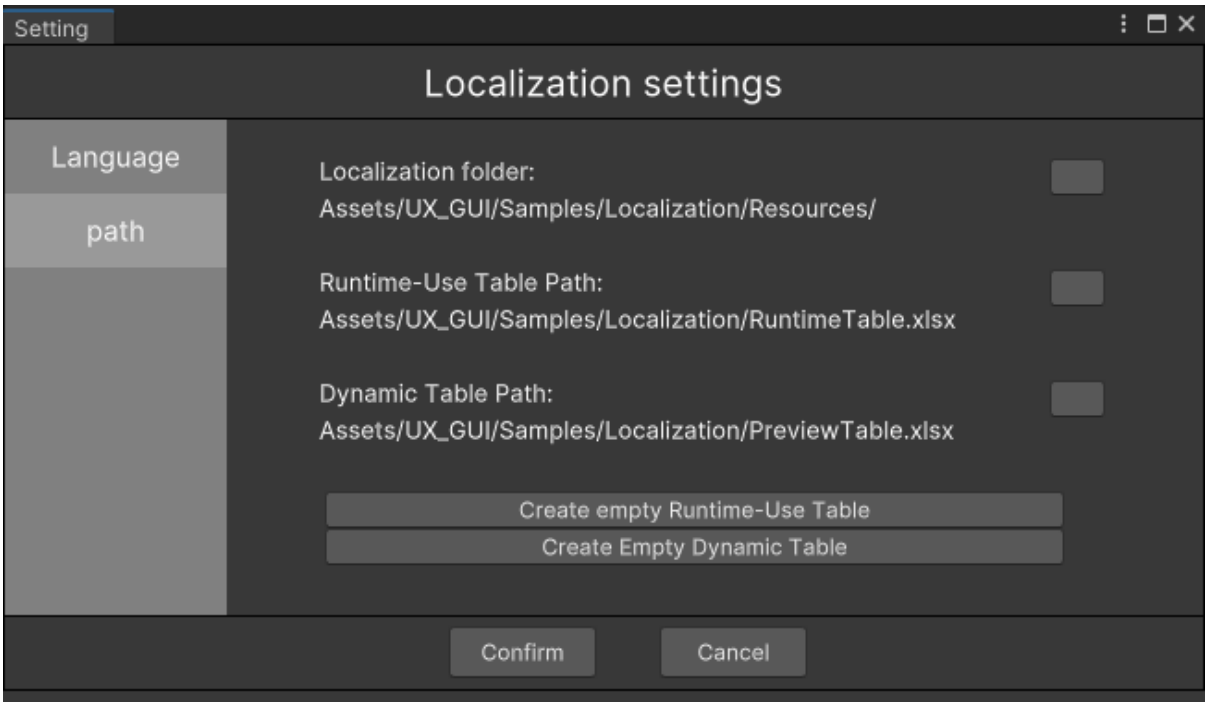
Text localization includes two text types: static text types and dynamic text types, which are marked in the subheadings in this manual.

Localization Overview

Configure localization language and path

Click the menu [ThunderFireUXTool->本地化 (Localization)->设置 (Setting)], a pop-up window will appear as follows.

In the "Language" section you can check all the languages supported by the game, there are 14 languages in total.



In the "Path" column, you can set the localization folder, static text form path and dynamic text form path.**The localization folder must be located in the path where resources can be loaded at runtime (e.g. under Resources folder).** All three paths must be in the "Assets /" directory.

The localization folder holds all localized image resources and the JSON file after converting the text form to JSON, named TextLocalization.json

- □ Static text form path is the form path used for static text localization, with the suffix ".xlsx"
- □ Dynamic Text Form Path is a form path for dynamic text localization with the suffix ".xlsx"

If there is no static text form, click the [Generate Empty Static Text Form] button, select a path in the pop-up window, and click [Save] to generate an empty static text form. It contains key, path, original text and translations for various language types.

If there is no dynamic text form, click the [Generate Empty Dynamic Text Form] button, select a path in the pop-up window, and click [Save] to generate an empty dynamic text form. It contains the key as well as translations for various language types.

Click [OK] after setting the above content to take effect.

Runtime Preview Language Switching

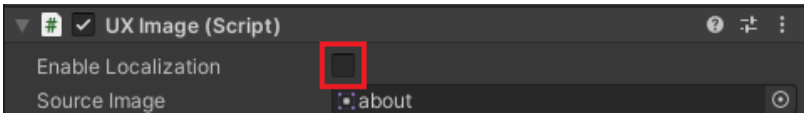
When running, you can set the preview language in the top right corner of Game (it will not change the in-game settings). The preview language will switch all UXImage, UXText and UXTextMeshPro with localization enabled to the corresponding language.

Click "In-game language" means not to use the preview language, use the game internal settings, the default in-game language is "not set", at this time click "In-game language" program does not do The user needs to use the code to set the in-game language (refer to **program interface**).

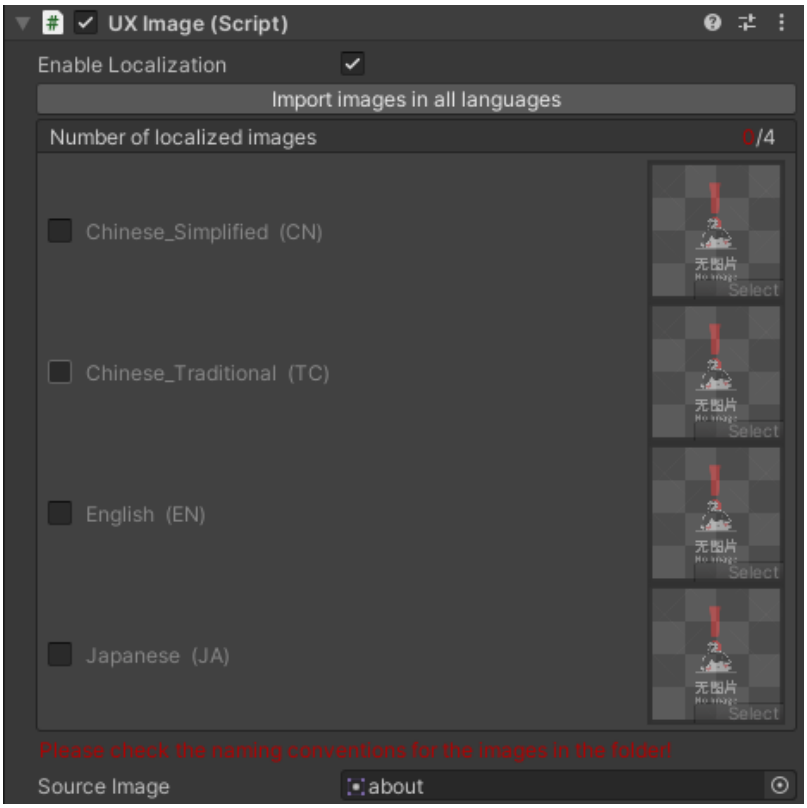
Image Localization

Turn on localization

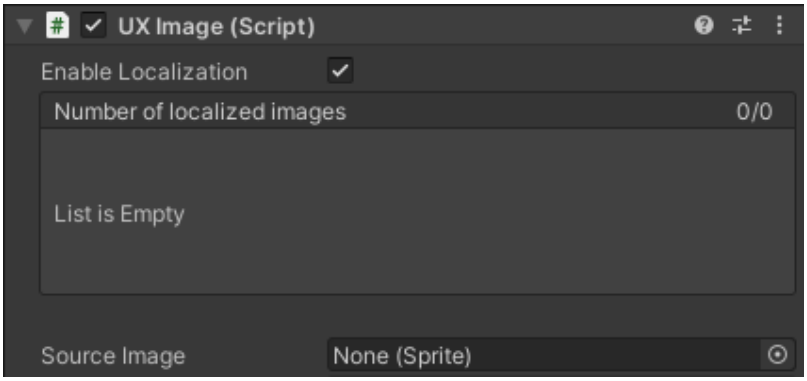
Select the UXImage that needs to be localized and turn on localization in Inspector.



When checked:



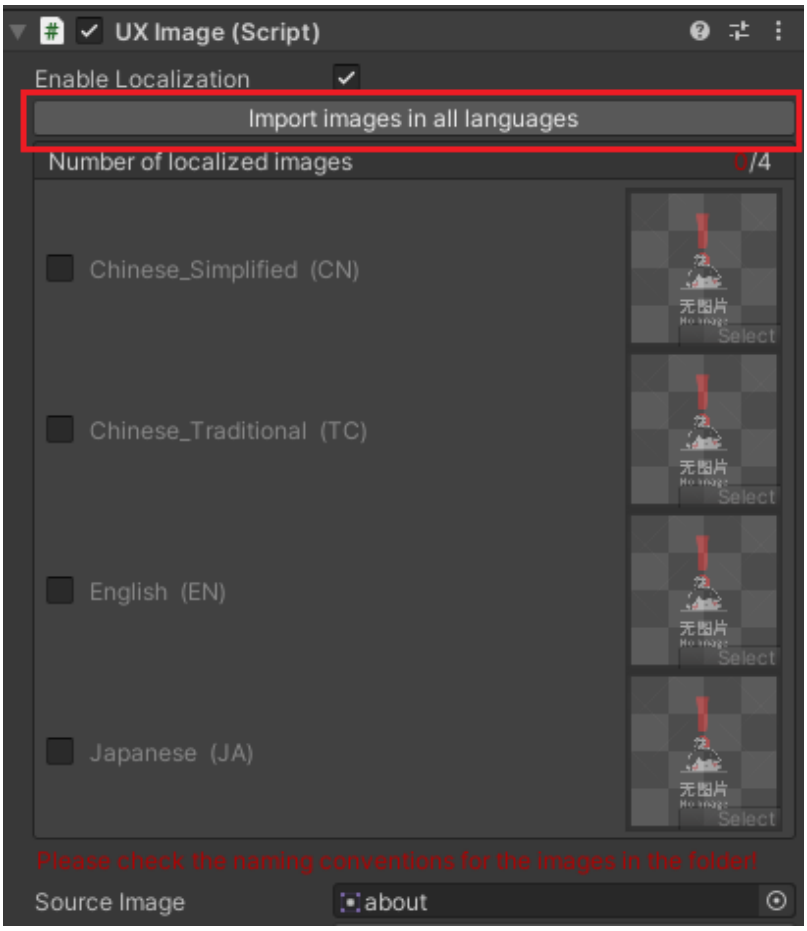
In particular, if the Source Image is empty, the "Import images in all languages" button is not displayed and the list of localized images is empty.



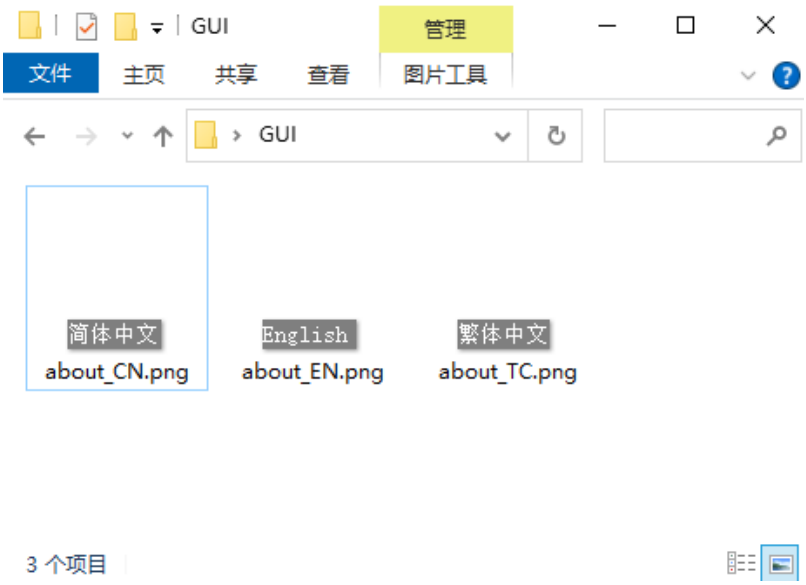
Import images in all languages

Click on the "Import images in all languages" button and select the folder where the localized images of the current Source Image are located to import the localized images and display them in the Inspector's localized pixel list, where:

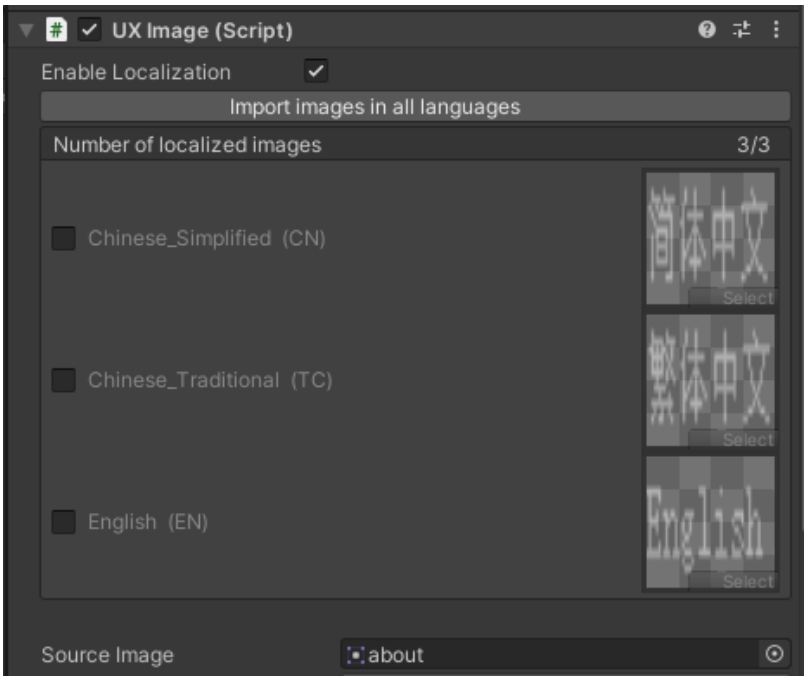
- The folder where the localized images are located does not have to be in the "Assets/" directory.
- The program recursively looks for all files in the folder.
- The image is named "image_name_language_suffix" with the extension ".png".
- The path to import is "localization folder/language suffix/image full name", if there is already an image with the same name in the path, it will be overwritten.
- If "Automatically convert Texture to Sprite" is checked in the "Function Switch" setting of the UX tool, the imported image will be converted to Sprite type automatically.



Selected folder:



After importing:



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

If a picture is missing for a certain language, a placeholder picture will be substituted.

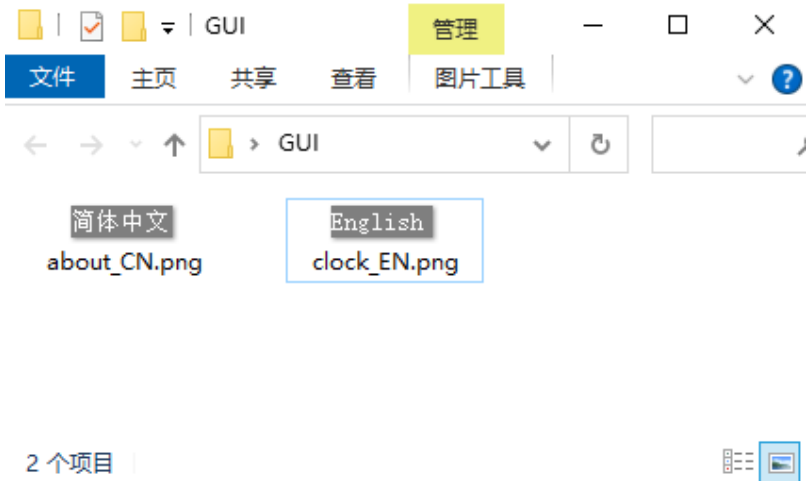
Batch import images

After clicking the menu [ThunderFireUXTool->本地化 (Localization)->导入图片包 (Import Localization Images)] and selecting the localization images folder, you can import images in bulk, where:

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

- The program recursively looks for all files in the folder.
- The image is named "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 the path, it will be overwritten.
- If "Automatically convert Texture to Sprite" is checked in the "Function Switch" setting of the UX tool, the imported image will be converted to Sprite type automatically.

For example, import the following folders.



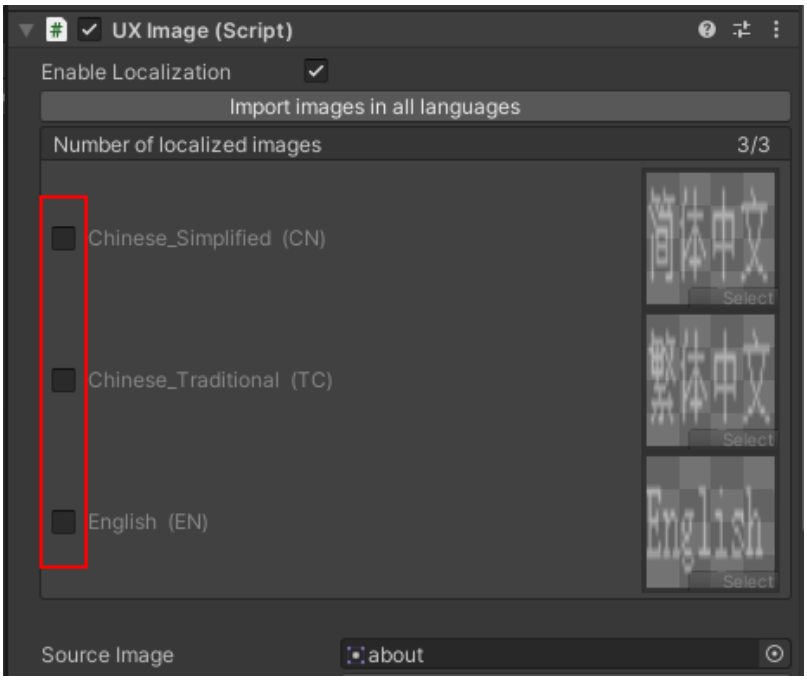
The program will import "about_CN.png" to the path "localization folder/CN/about_CN.png" and "clock_EN.png" to the path "localization folder/EN/clock_EN.png". " into the "localization folder/EN/clock_EN.png" path.

Automatic recognition after original image change

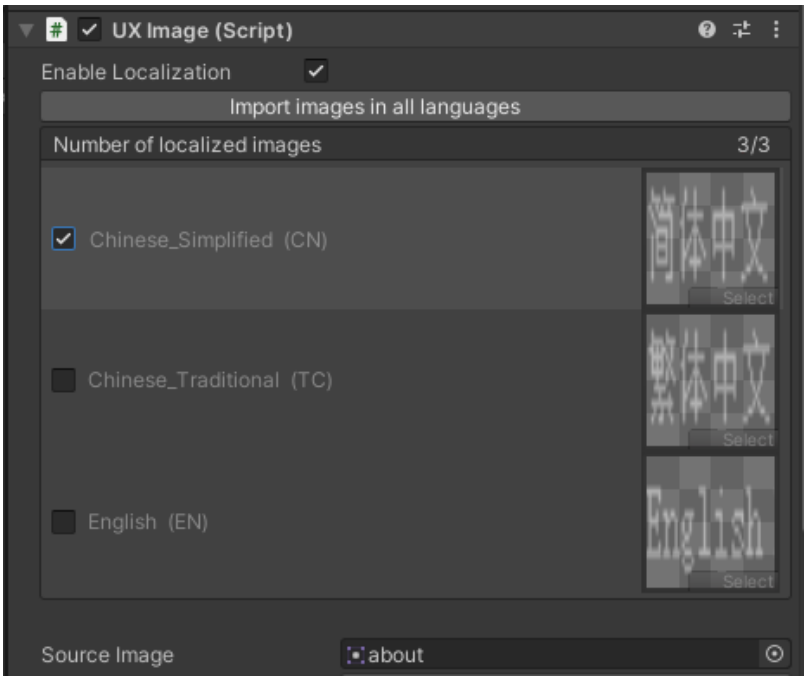
When the Source Image is changed in the Inspector, the localized pixel list is automatically updated.

Preview images in different languages

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



When checked:



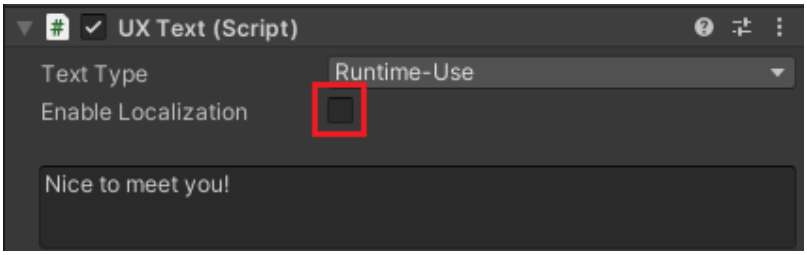
In the scenario:

Note: This function will automatically set the current object as invisible in Hierarchy, if there is an overlapping of images, please check if  is open in Scene.

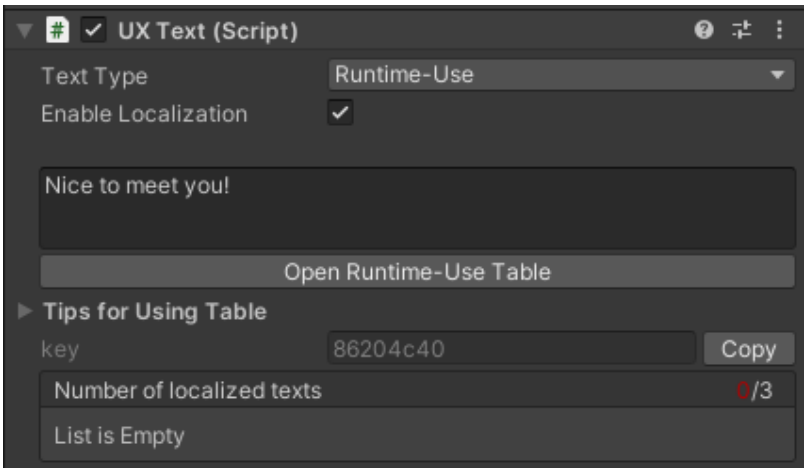
Text localization

Turn on localization

Select the UXText or UXTextMeshPro that needs to be localized and enable localization in Inspector.



When checked:

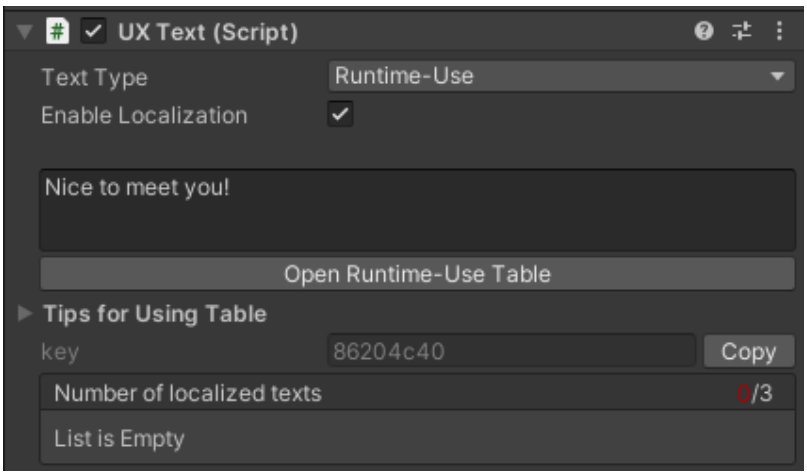


Static text

Static text is text created and edited directly in the scene or Prefab and can be previewed directly at runtime. Change the text type to "Static Text" and configure the text content.

Fill in the translation data

Clicking on "Open Static Text Form" will record the localized UXText and UXTextMeshPro information in the current Prefab into the form and open the form, if it does not exist a new form window will pop up.



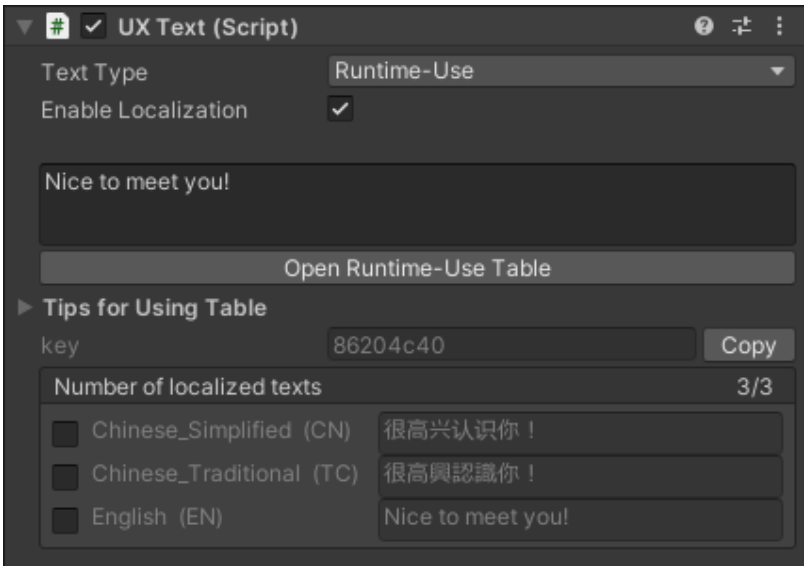
When opened:

	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!			

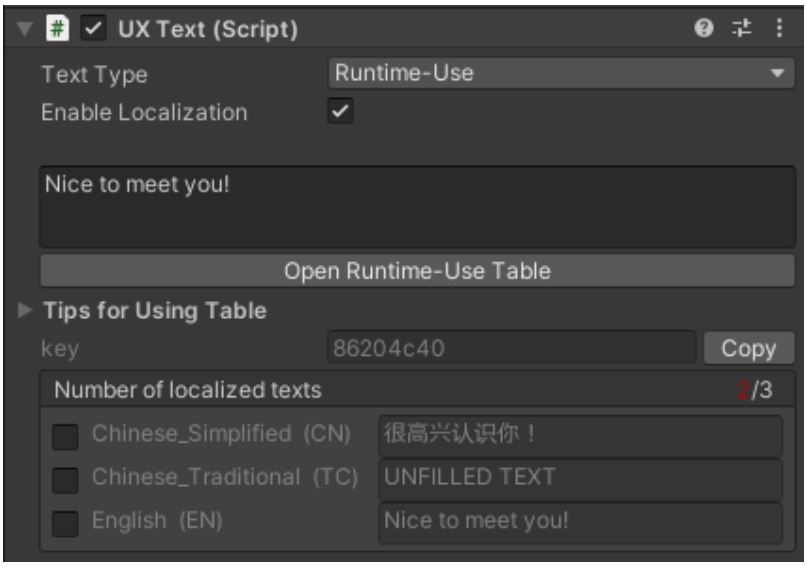
Fill in the translation data (multiple rows of data in one cell are supported).

	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, click the menu [ThunderFireUXTool->本地化 (Localization)->将文本表格转为JSON文件 (Convert Text Table to JSON)], and you can see the effect in the localized text list in Inspector.



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

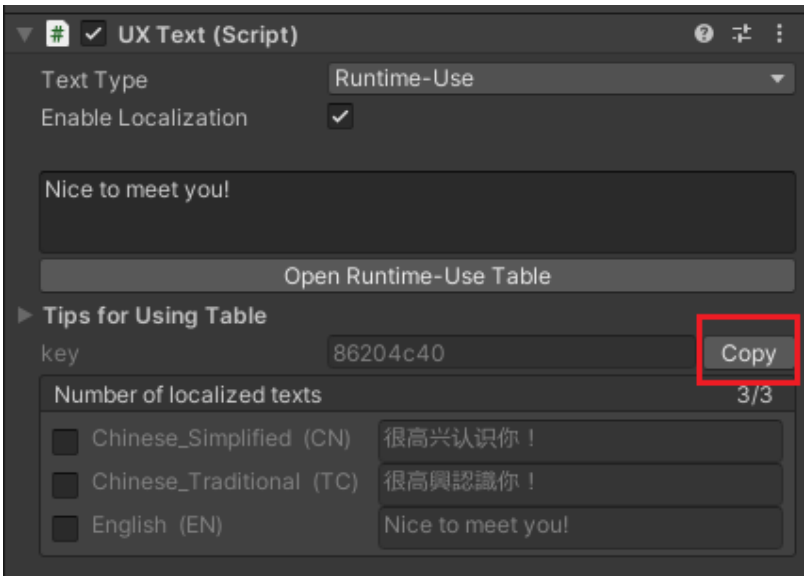


key value generation

The key value for the static text type is automatically generated, and a unique key value is generated for UXText or UXTextMeshPro when it is first checked with "Enable Localization", and also when the original text is changed.

key value copy

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, consisting of the file name + level name. When the same object has multiple instances in different Prefab, the path information will record all these paths, separated by "&&".

Form Synchronization

Click the menu [ThunderFireUXTool->本地化 (Localization)->刷新静态文本表格 (Refresh Runtime-Use Text Table)] to manually synchronize the static text table. The manual sync will use all UXText and UXTextMeshPro information in Prefab that has localization enabled and the text type is static text to sync.

Also, when the Prefab is saved, it is automatically synchronized with the UXText and UXTextMeshPro information in the current Prefab that has localization enabled and the text type is static text.

User Data Deletion Prevention

When synchronizing forms, translation data already filled in by the user is automatically retained, even if the object corresponding to that data no longer exists (the path and original text are empty at this point).

Exception Information

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

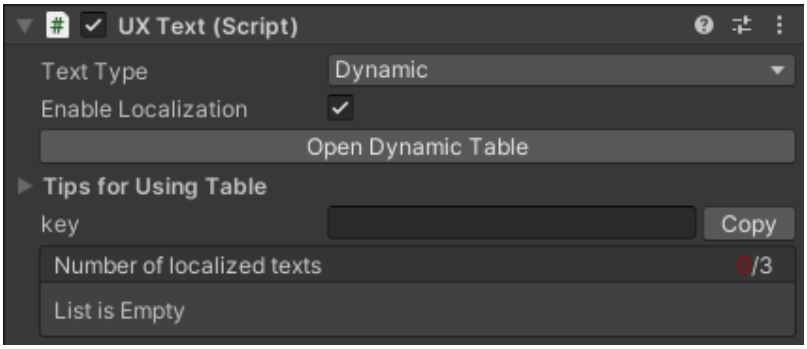
- An error is reported when the program tries to write to a form but the form is already open.
- When two objects have the same key value, but the original text is different, a warning will be reported (usually not encountered).

Dynamic Text

Dynamic text is text that is generated in real time during runtime 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 Text Form" will open the form according to the "Dynamic Text Form Path" in the localization settings, and if it does not exist, the Create New Form window will pop up.



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, then you need to click the menu [ThunderFireUXTool->本地化 (Localization)->将文本表格转为JSON文件 (Convert Text Table to JSON)].

key value copy

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.

Converting text tables to JSON

Click the menu [ThunderFireUXTool->本地化 (Localization)->将文本表格转为JSON文件 (Convert Text Table to JSON)] to convert static text table and dynamic text 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, check a language to preview the text in that language in Scene, where the preview does not modify the original file.

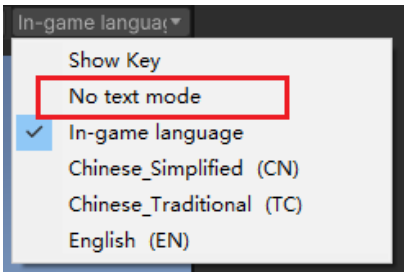
When checked:

In the scenario:

Note: This function will automatically set the current object as invisible in Hierarchy, if there is an overlapping of images, please check if  is open in Scene.

No text mode

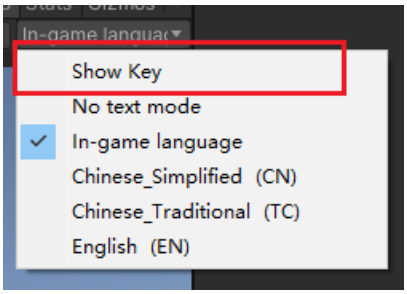
The textless mode can be set in the upper right corner of the game at runtime, which will switch all UXText and UXTextMeshPro to box placeholders.



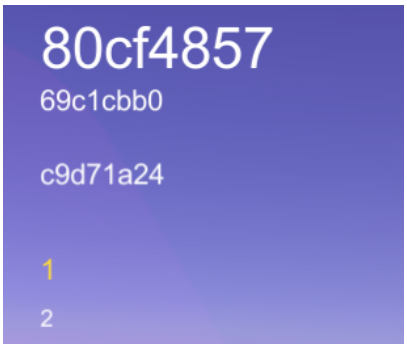
After the switch:

Show key

When running, 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 the switch:



Program Interface

Function

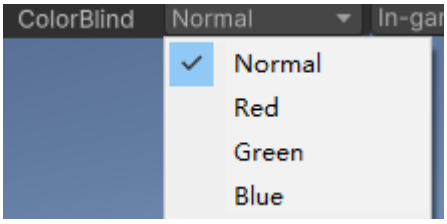
```
// Description: You can switch the in-game language
// Class：LocalizationHelper
// Parameters:
//     type: the language type after switching
// Return value: None
public static void SetLanguage(LocalizationHelper.LanguageType type)
```

Color Blind 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 people with color blindness.

How to use

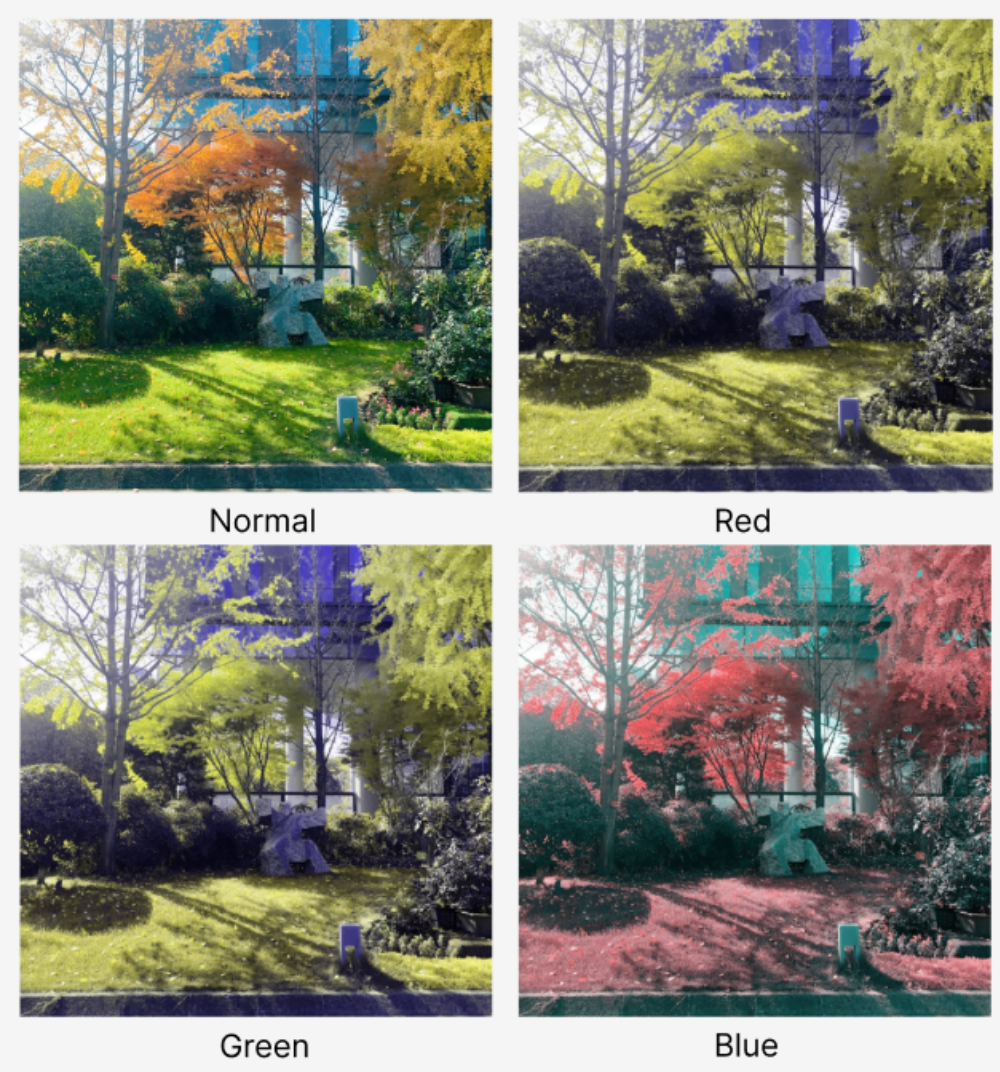
You can set the color blind mode in the upper right corner of the Game window. The default colorblind mode is normal, which is the interface effect in non-colorblind state. The drop-down list can be switched to red, green and blue colorblind modes.



Scope of action

The color blind mode works on the full screen, i.e. including 3D objects, 2D objects, UI, etc.

Example



Note: The color blindness mode of this tool only supports the built-in rendering pipeline, which creates a script called Color Blindness Effect on the MainCamera.

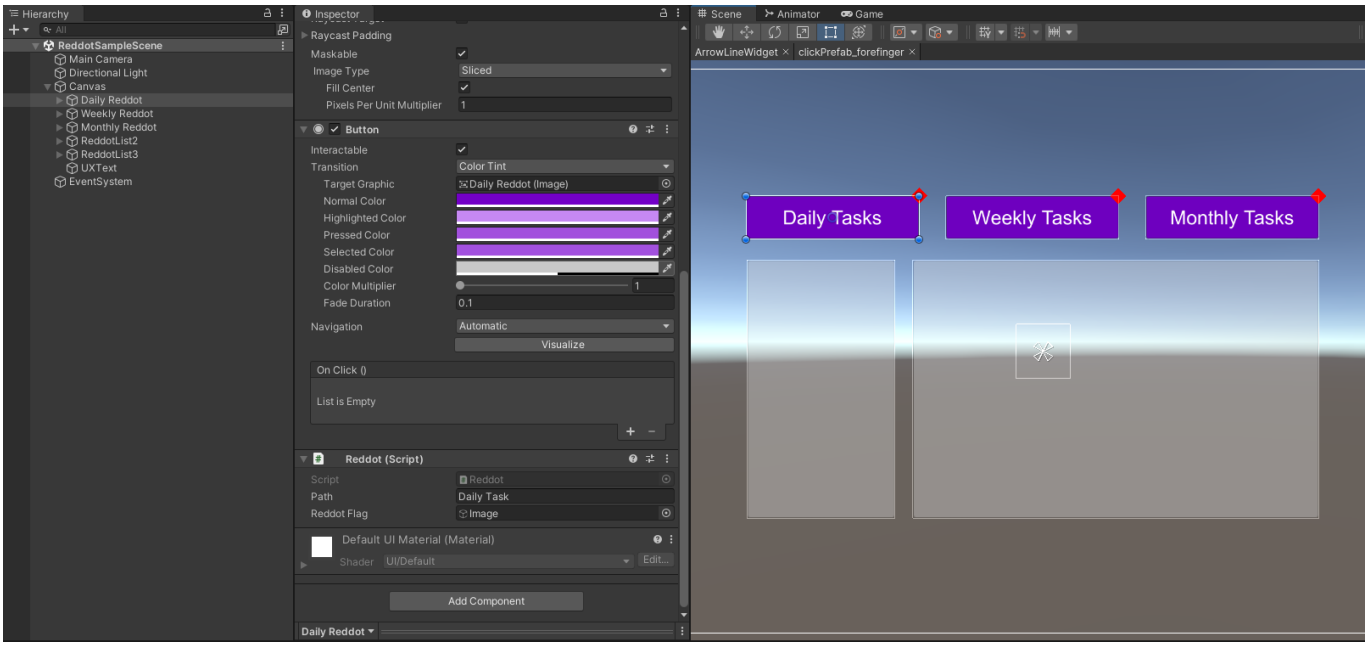
Red Dot Function

ThunderFire UX Tool implements a built-in red dot management system, by adding Path to each red dot object, to achieve unified processing of red dot hierarchy.

How to use

1.Mount script

Add a Reddot script to the object to be displayed with red dots.



2.Red dot marker configuration

The Reddot Flag object is the object that is revealed when the red dot state changes. It is recommended to set it as a child node of the Reddot object.

3.Red dot path configuration

Path means the associated path of the red dot data, set the format as "stringA/stringB/stringC/stringD".

Note: When setting the Path in the second level of the red dot, you should set the full path, it should be "stringA/stringB", not just "stringB".

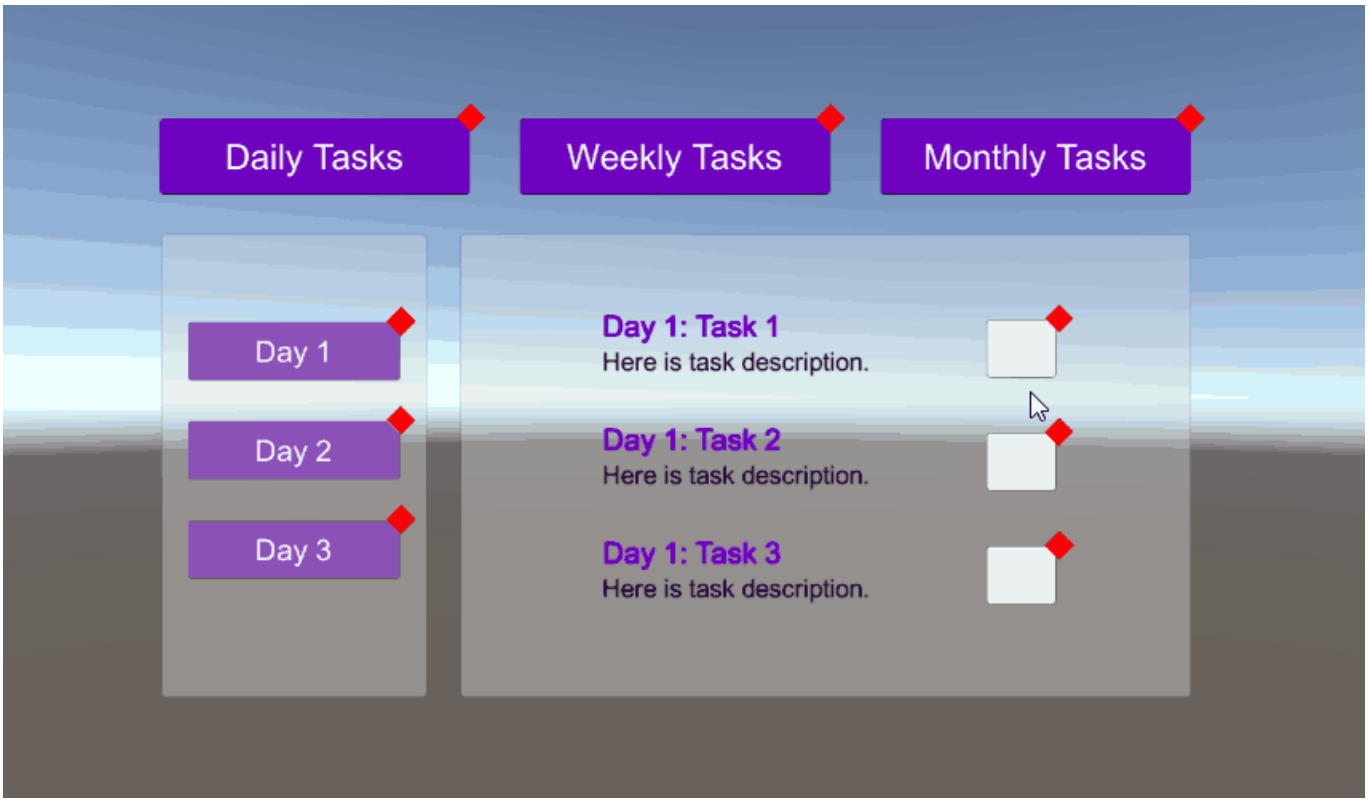
- Static configuration For the identified red dot objects (such as the function entry buttons on the main screen), you can set them directly in the editor.
- Dynamic Settings For red dot objects that are generated dynamically or cannot be determined at the editing stage (e.g. red dots in a list), they can be set by code. (Refer to the **program interface** below for the code)

4.Red dot status setting

The red dot status can be set by code during game operation. (Refer to the **program interface** below for the code)

After setting the red dot status, the red dot tool will update the display status of all red dots on the path according to the path. The update rule is that **the leaf nodes in the red dot tree use their own display status, and the non-leaf nodes decide their own display status according to the display status of their children**.

Example



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

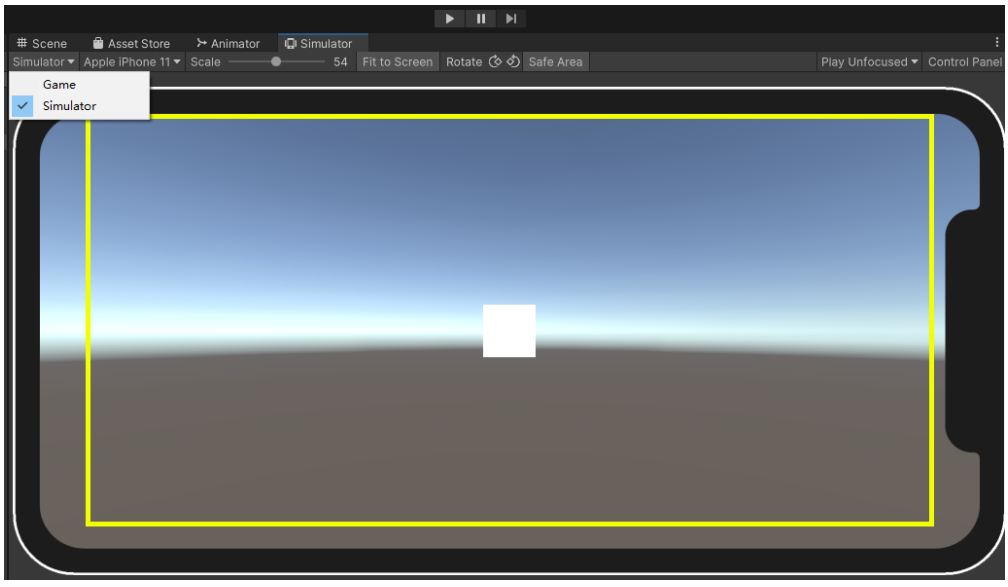
```
// Case 1
// Dynamically add red dot paths to components in the code
Reddot reddot = go.GetComponent<Reddot>();
reddot.Path = "Reddot1/firstChild";

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

Fringe Screen Adaptation

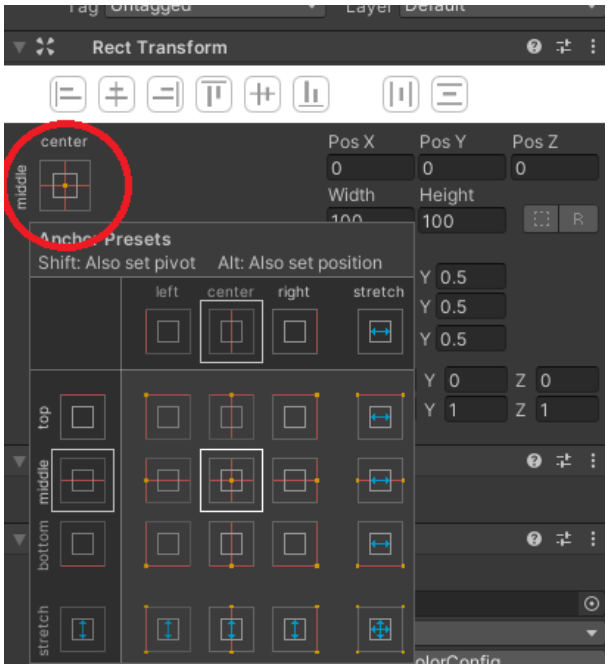
The ThunderFire UX Tool enables fast adaptation of the UI to different models with different operable areas.

How to use

- Switch Unity Game scene to Simulator mode.

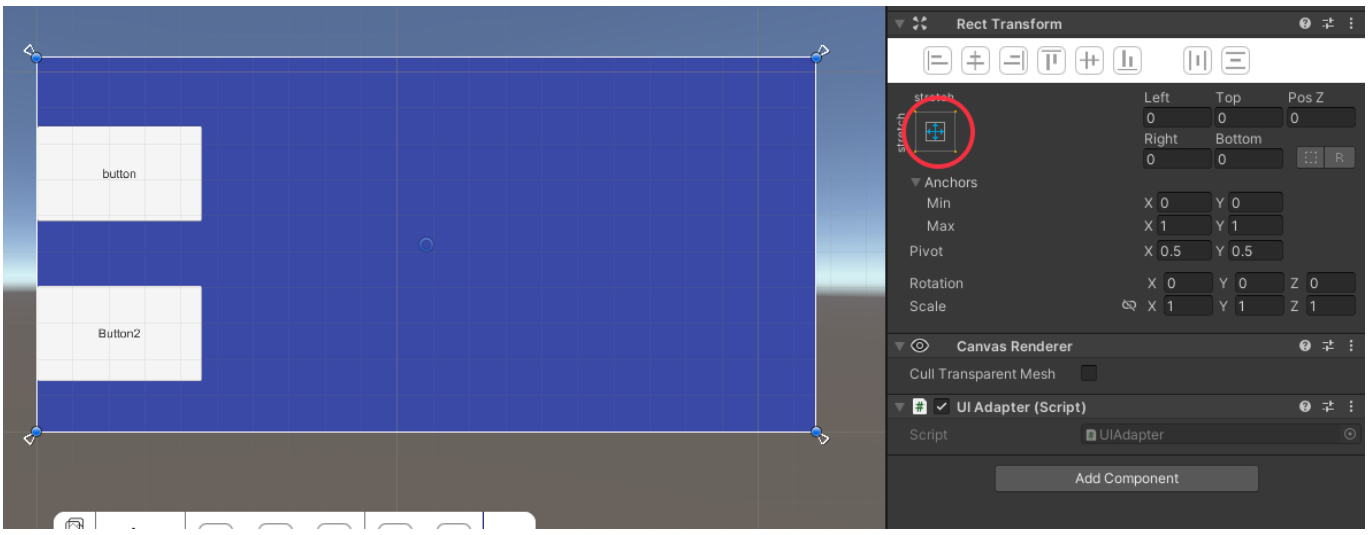


- Create a root node under Canvas that includes all other UI components, and set Anchor Preset to Stretch and fill up the Canvas space.



- The root node mounts the UIAdapter script and sets the Anchored Preset of the component on the edge of the UI that needs to be adapted to the bangscreen to a type other than Middle Center.
- Hooking up IgnoreUIAdapter scripts to UI components based on edge-determined positions can make them ignore the fringe screen adaptation (mainly for background images, etc.)

UIAdapter script usage



To enable the UI root node to adapt to the Canvas size, generally set the Anchor Preset of the root object to Stretch, and then make UI adjustment based on this, as shown in the red box in the figure. Hang the UIAdapter script on this root object, and the script will adjust the Anchor of this root object to the appropriate value according to the size of SafeArea, so that the bangs screen adaptation function is completed. The steps are as follows.

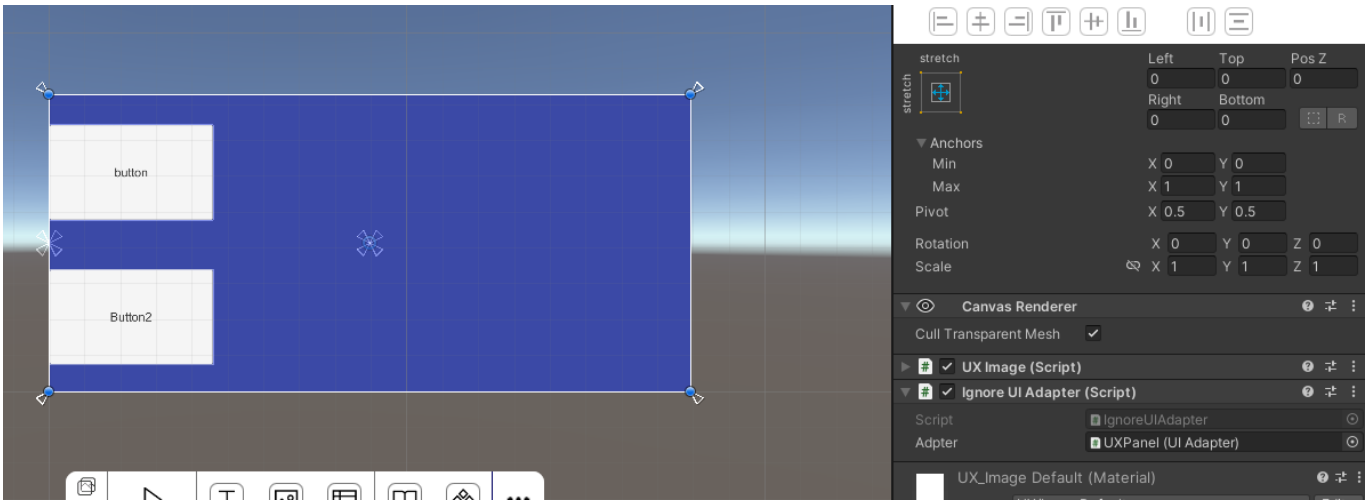
1. Set the root Anchor Preset to Stretch.
2. Resize the root node to the appropriate size.

3. Mounts the UIAdapter script.

4. Open Simulator to select the model and run it to see the result.

Please note: If a child object in UI needs to be adapted for bangscreen, please set its Anchored Preset to a type other than Middle Center.

IgnoreUIAdapter script usage



For some UI that doesn't want to adapt to the bangs screen (mainly refers to the background in the game), hook it up with IgnoreUIAdapter script, set the UIAdapter of its parent node in its Inspector panel, we will calculate the offset of UIAdapter and add the original offset to this object to achieve the effect of ignoring the Adapter in the parent node . In general, the Anchor Preset of this type of background should also be of Stretch type. The steps are as follows.

1. Set the background Anchor Preset to Stretch.

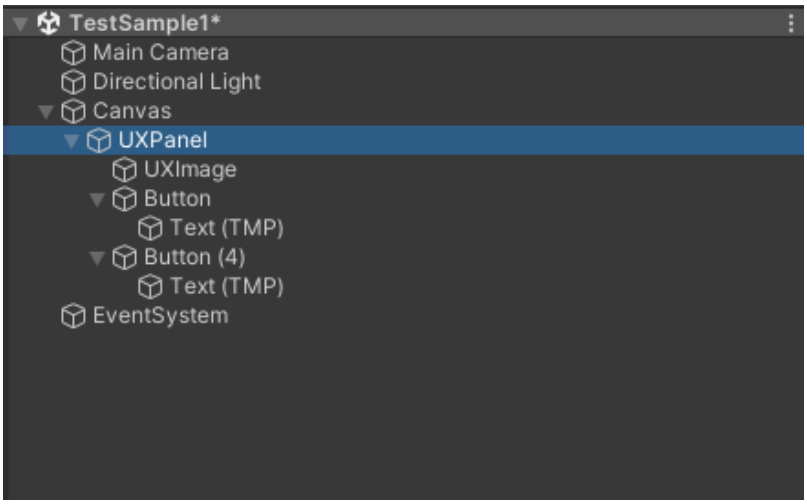
2. Resize.

3. Mounts the IgnoreUIAdapter script.

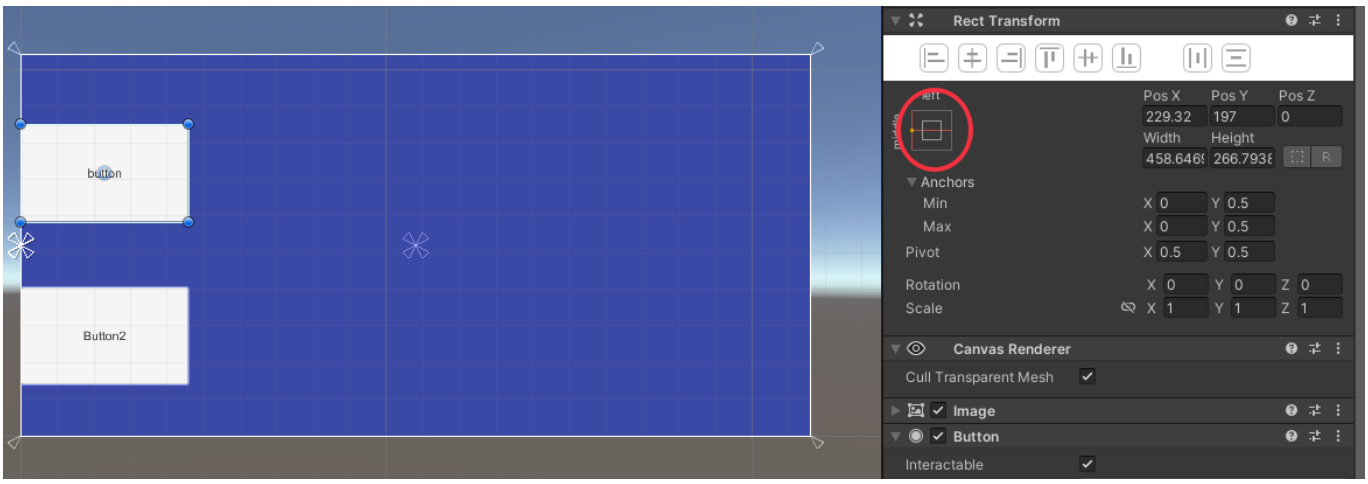
Please note: If the Anchor Preset for this object is Middle Center, please do not hook up this script.

Adaptation examples

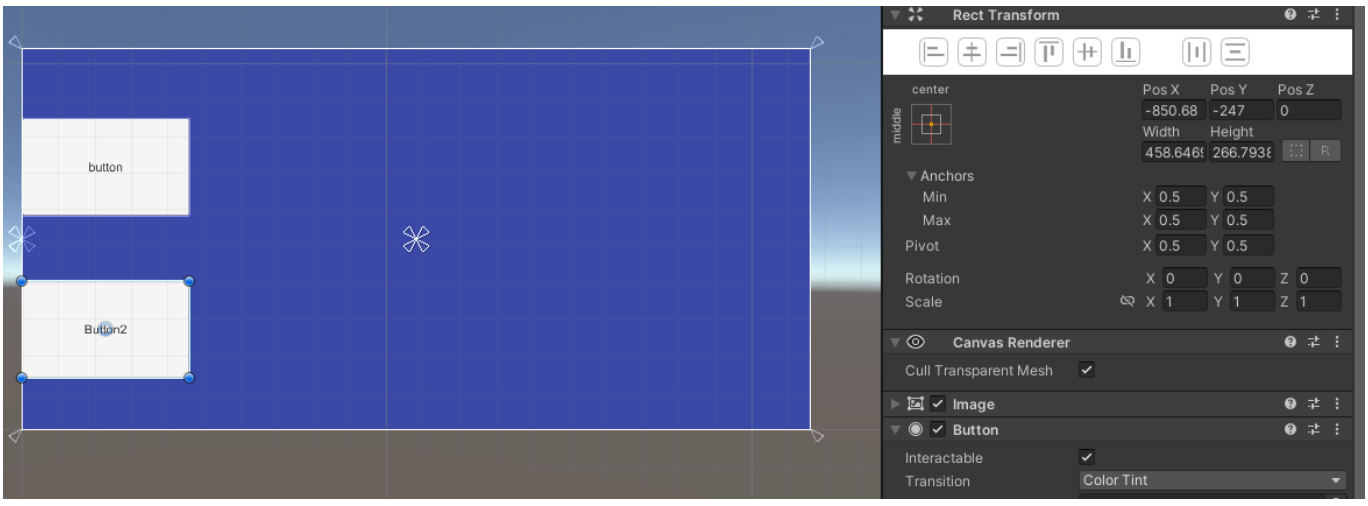
Example scene directory structure: UXPanel is the root node to be adapted, according to the previous requirement to mount UIAdapter script, and should occupy the full Canvas space. uxImage is the background image, according to the previous method to do ignore processing, it is best to occupy the full space of the root node.



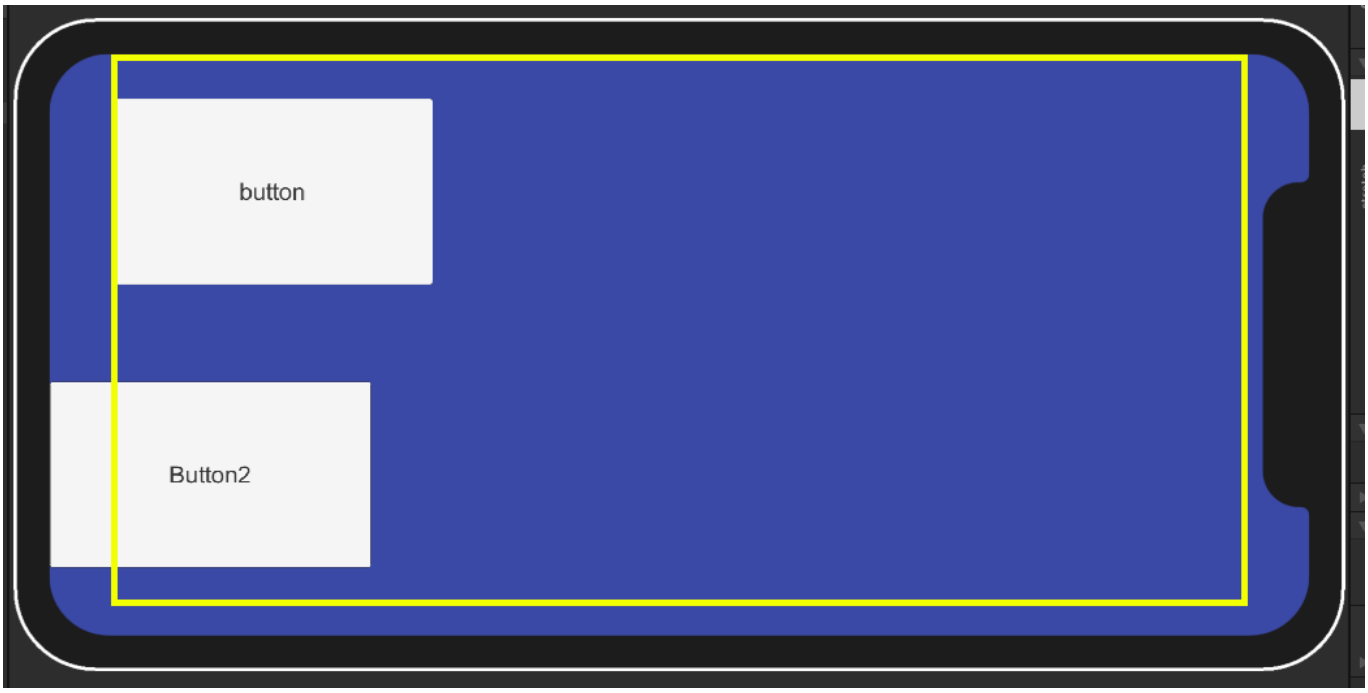
Example of the first Button: Anchored Preset set to a type other than Middle Center.



Example of the second Button: Anchor Preset is Middle Center, so the adaptation does not guarantee that it is necessarily within the safe range.



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


button
Button2