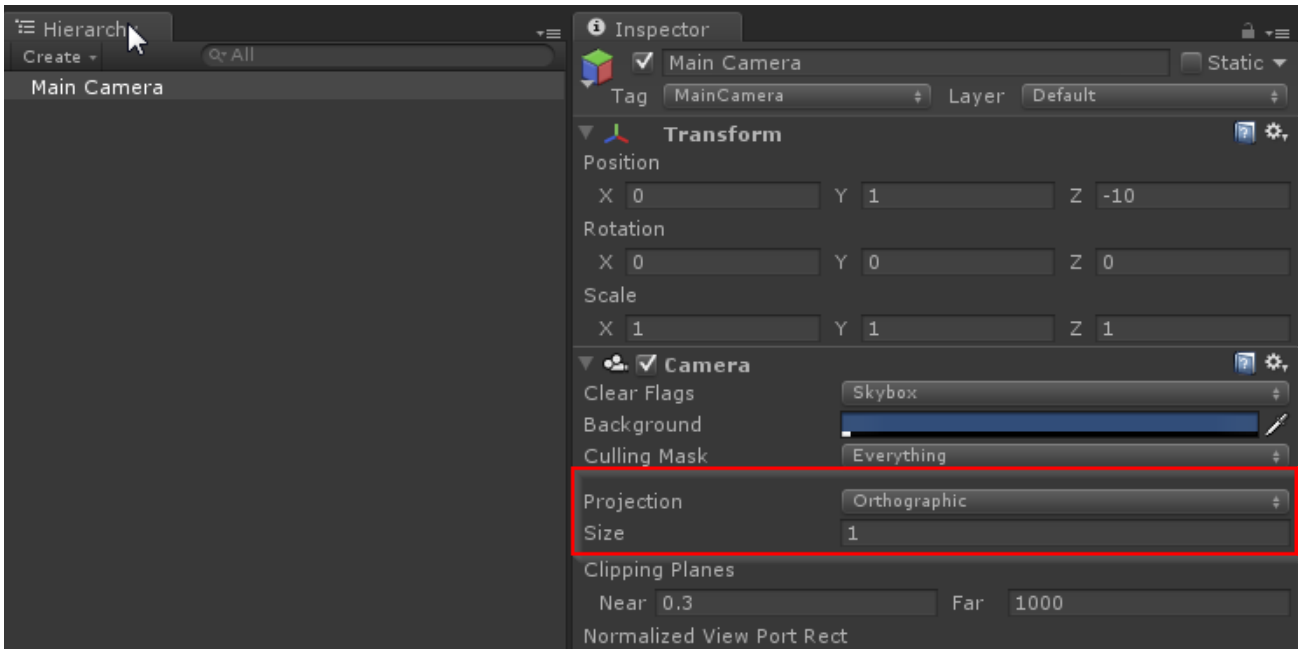
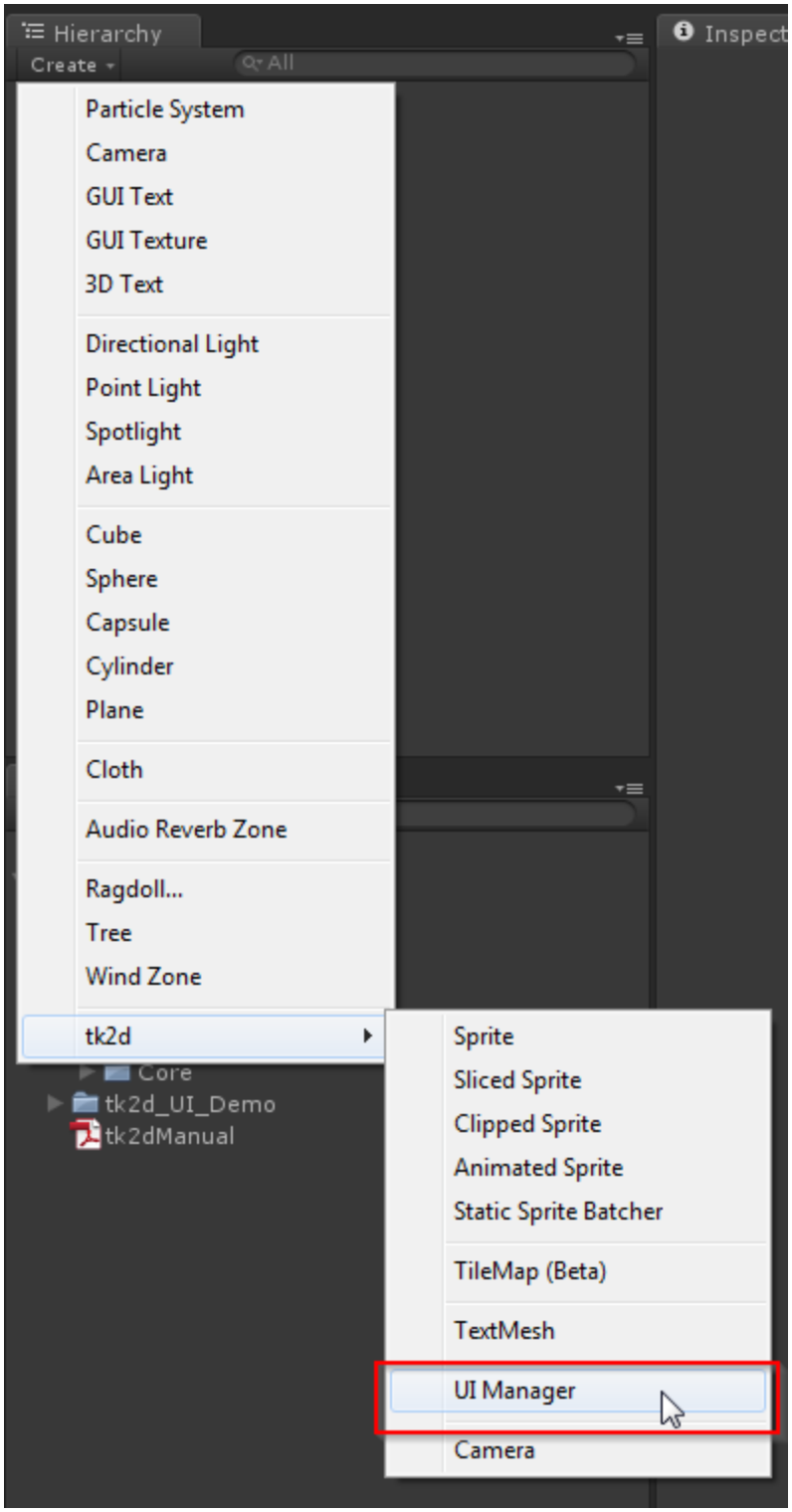
Setting up UI

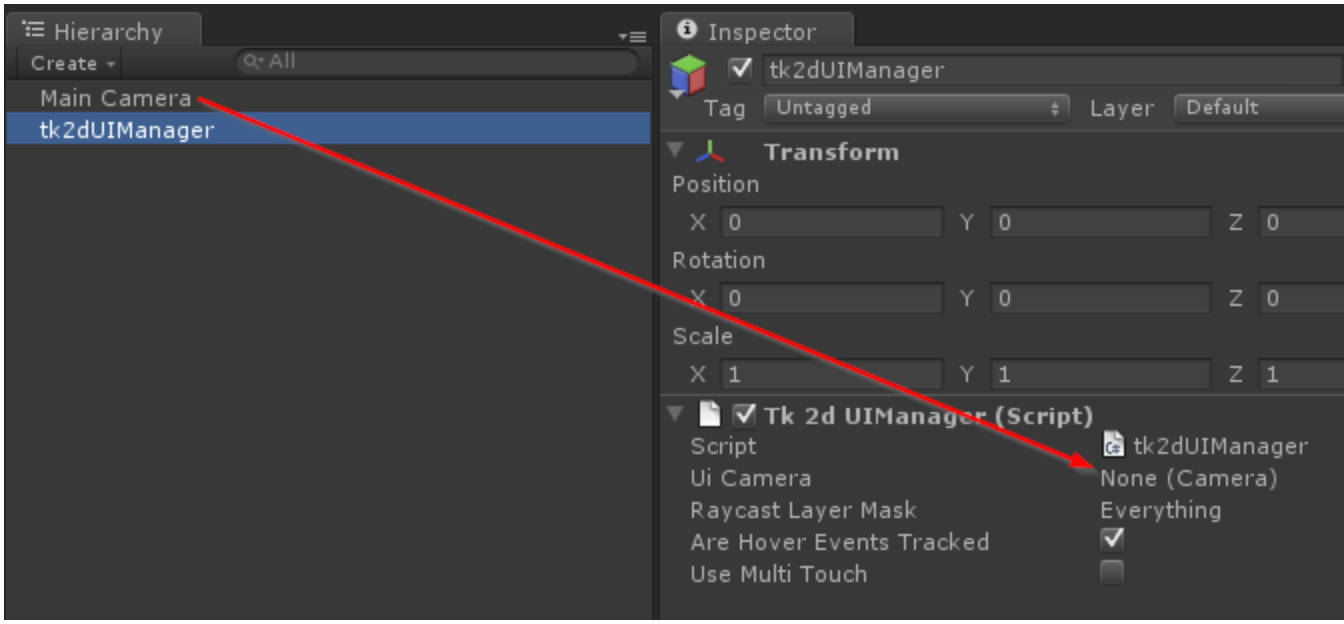
1. Open a new scene
2. Click on Main Camera object in the Hierarchy Window, and the Camera editor inspector appears
3. In Main Camera Inspector set Projection = Orthographic and Size to 1.



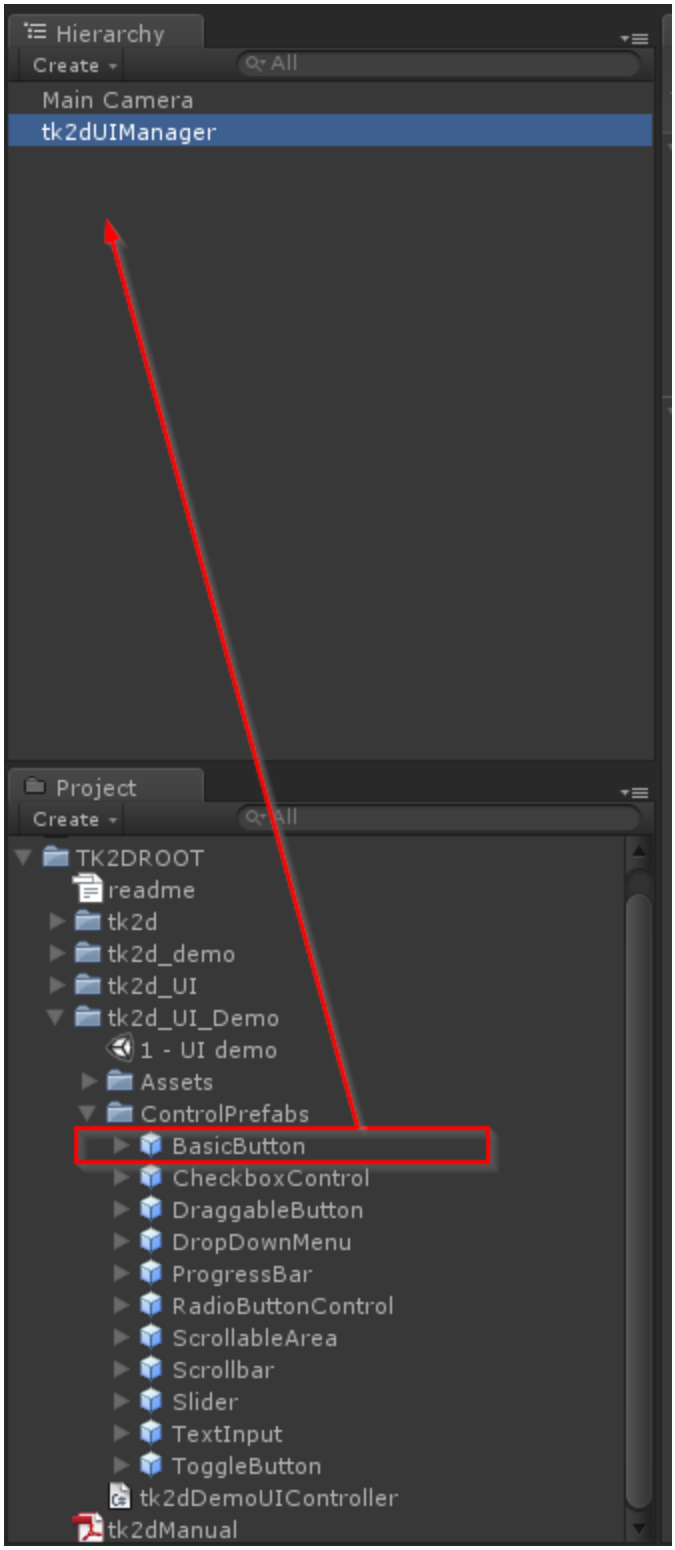
4. Create UIManager by clicking on "Create > tk2d > UI Manager" in the Hierarchy Window.



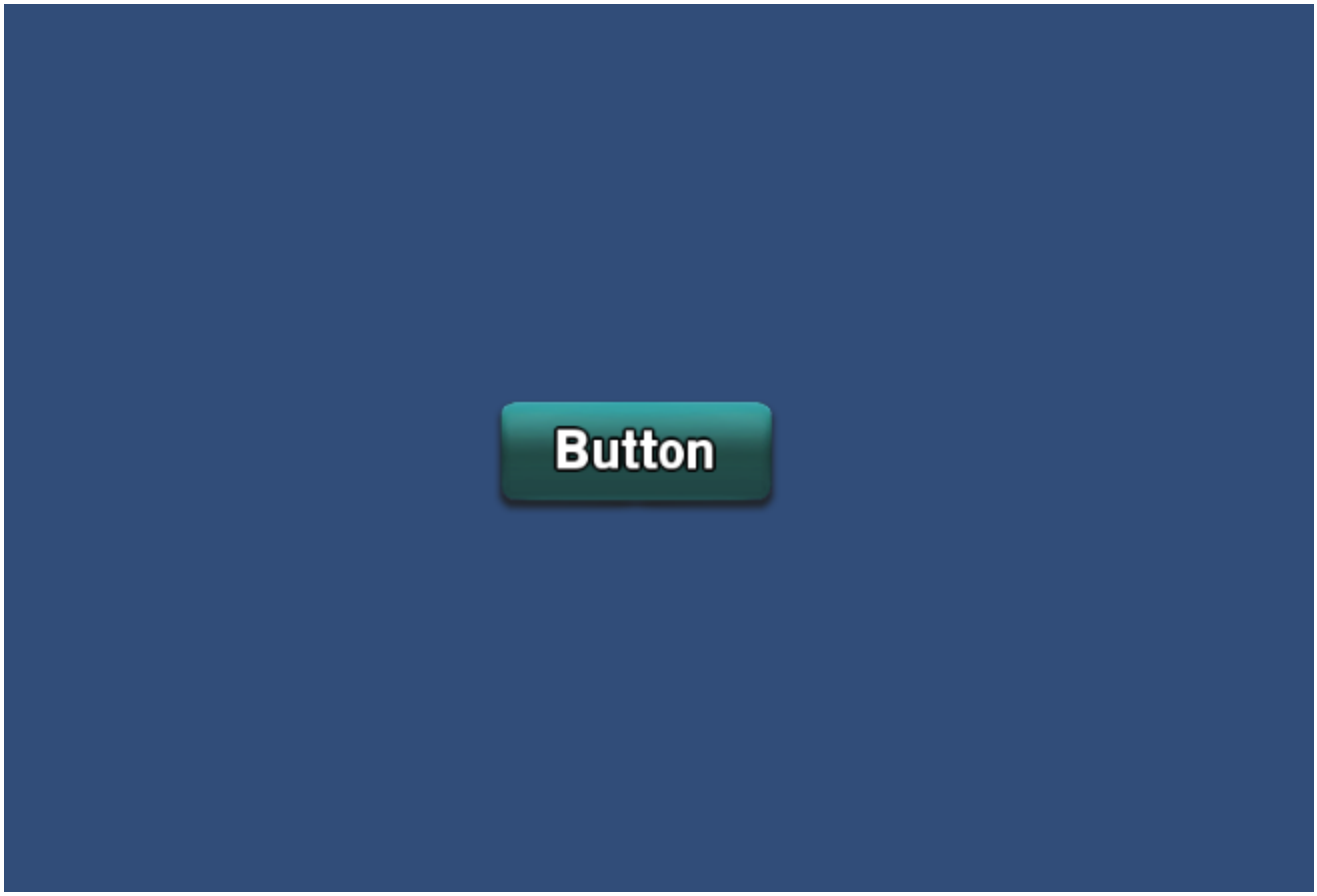
5. Click on the UIManager object in the Hierarchy Window, and the UIManager editor inspector appears
6. Drag the camera used to show the UI in the scene into the ‘UI Camera’ field in the inspector



7. In Project View, navigate to TK2DROOT/tk2d_UI_Demo/ControlPrefabs
8. Drag BasicButton prefab into the scene

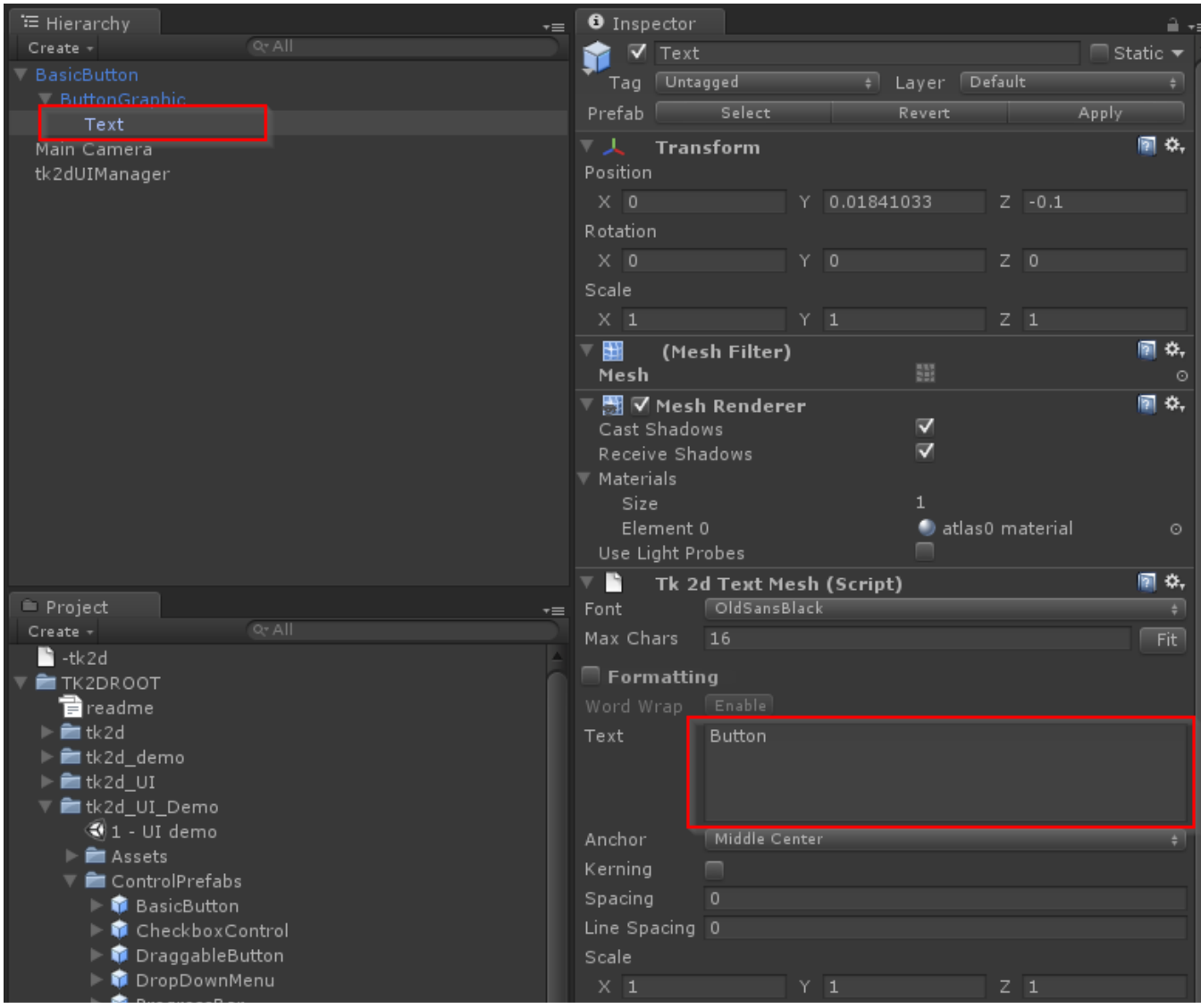


9. Hit Play and you will be able to click this newly created button

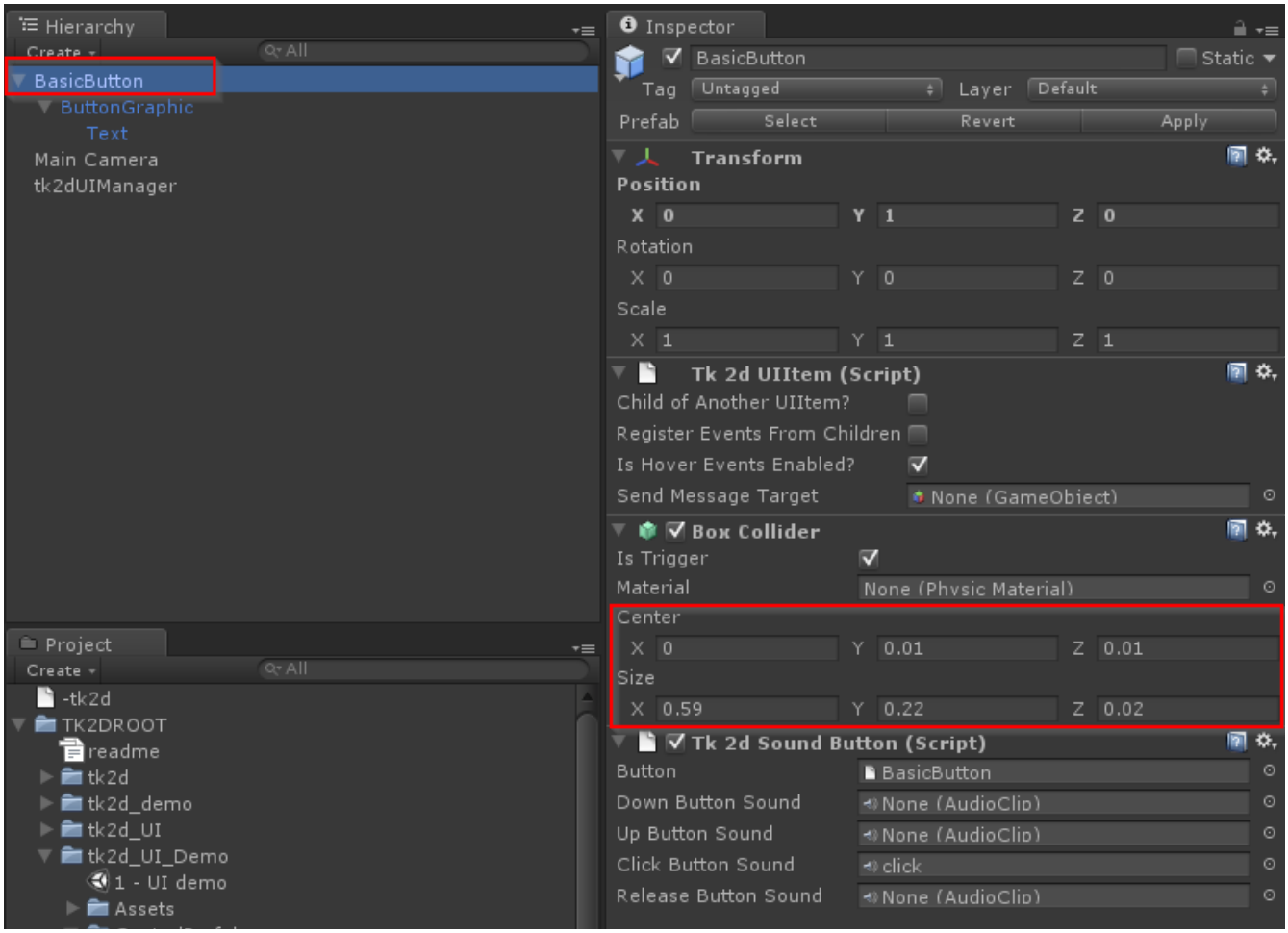


Customizing Basic Button

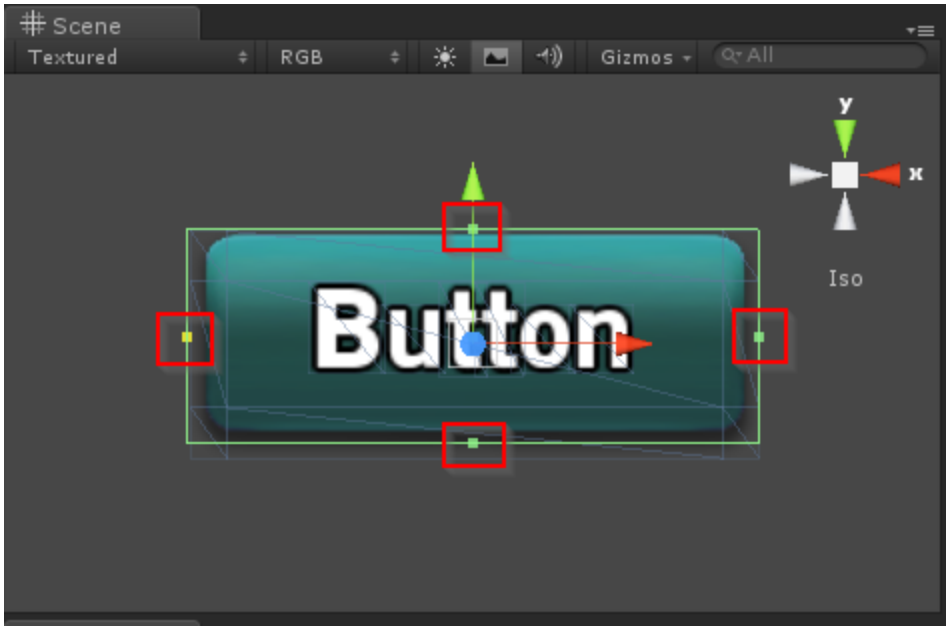
1. Follow instructions in ‘Setting up UI’ tutorial
2. Click on BasicButton object in the Hierarchy Window
3. To change text, in the Hierarchy drill down under BasicButton, then under ButtonGraphic to find Text.
4. Select Text GameObject in the Hierarchy. Edit the ‘Text’ field in the ‘Tk 2d Text Mesh’ in the inspector.



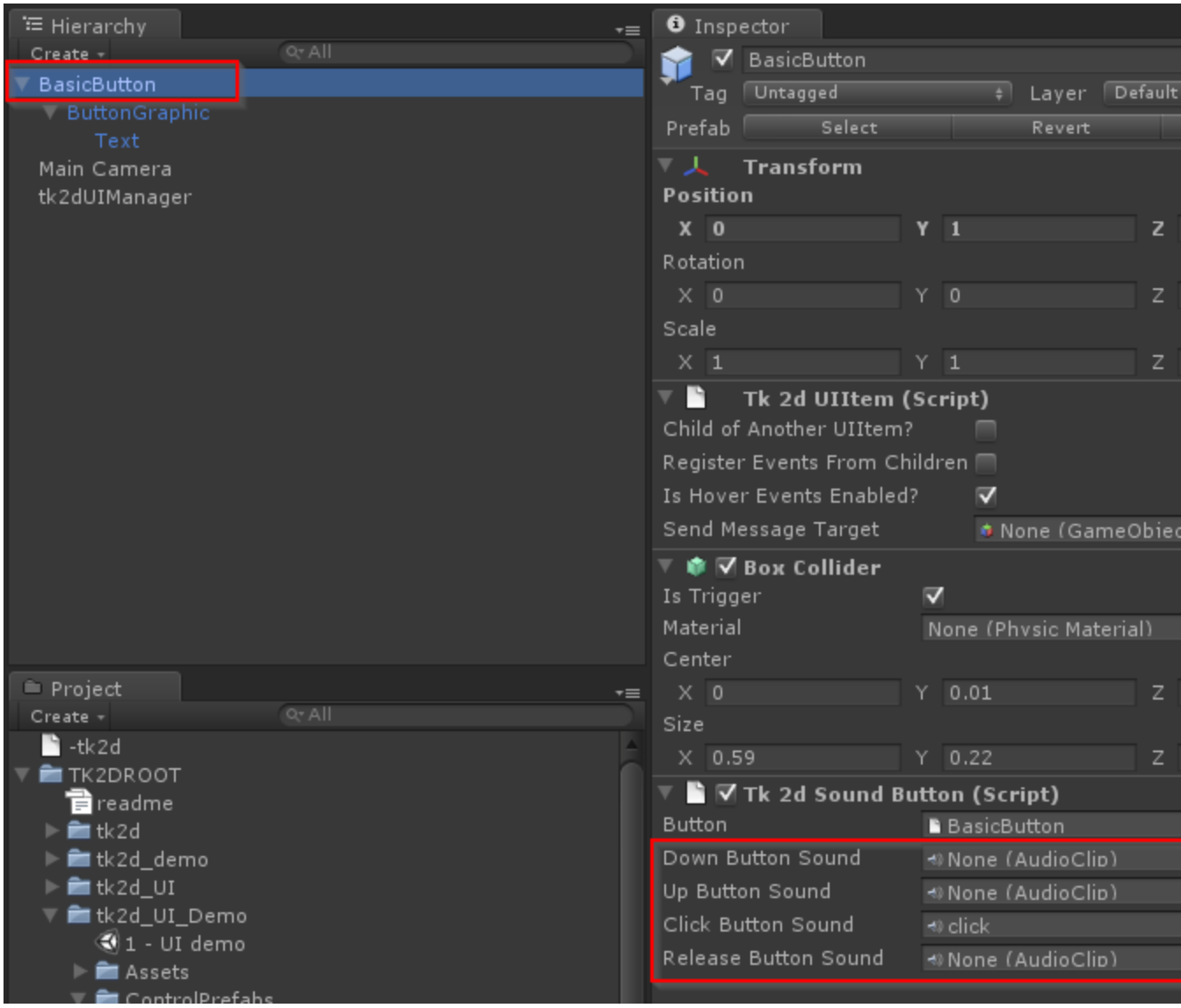
5. To change button size or graphics. Select ButtonGraphic in the Hierarchy (child of BasicButton). Edit the sprite in ‘Tk 2d Sliced Sprite’ and/or change the Dimensions (Pixel Units) to change the width and height. Then click on ‘BasicButton’ in the Hierarchy and modify the Box Collider width and height to match.



Tip: If you hold shift while a GameObject with a BoxCollider is selected small handles will appear in the Scene View which can be used to resize the collider.



6. To change the sound click on ‘BasicButton’ in the Hierarchy and in the inspector there is a ‘Tk 2d Sound Button’. You can added/remove/change audio clips here.



Fixing up colliders.

When you have switched sprite collections and resized your sprites to fit a new configuration, you should select your UI root gameobjects, and run the “2D Toolkit > UI > Fix Selected Item Bounds”. This will automatically try to best-guess the most appropriate collider sizes for all selected objects and children. Refer to the “Advanced tk2dUIItem” section for more details on how this is performed.

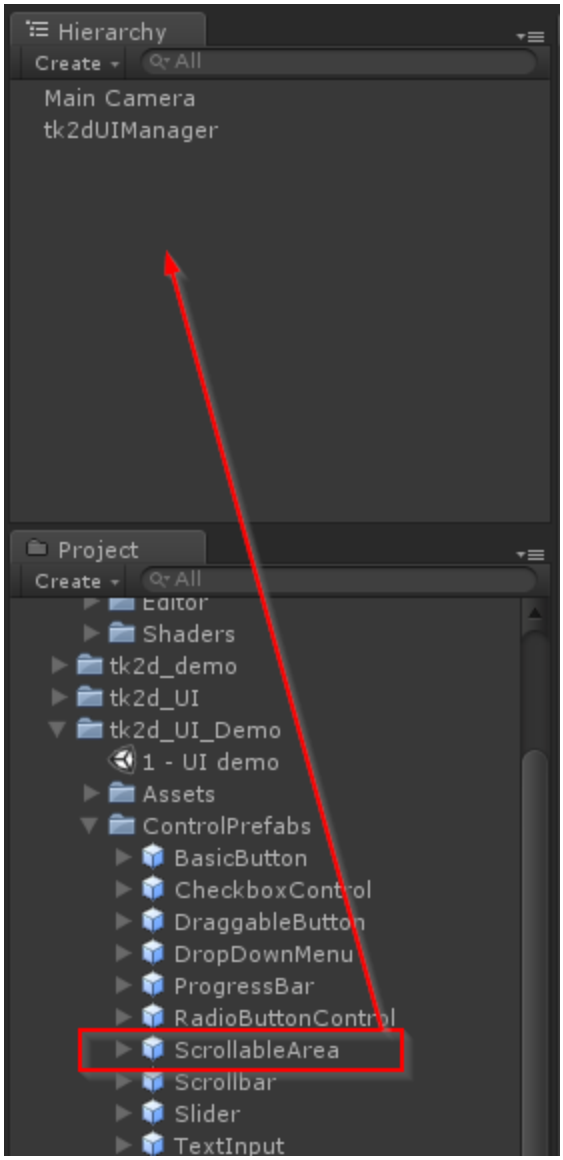
Built-in Component Examples

- Basic Button - Default button that scales when clicked and plays click audio
- CheckboxCheckboxControl - Contains a checkbox and a description
- DraggableButton - When held down you will be able to drag it around the scene
- DropDownMenu - Dropdown Menu List Control. Change items in list by editing - ‘Starting Item List’ in the tk2dDropDownMenu inspector
- ProgressBar - Will move a progress bar based on the ‘Value’
- RadioButtonControl - Contains radio button and a description
- ScrollableArea - Area that will be scrollable. Simply drag into the scene to use. Consult the “Advanced - Setting up Scrollable Area for 3D Objects.” if you need to mix 2D and 3D objects in the scene.
- Scrollbar - Scrollbar that can be attached to scrollable area. When resizing exact length can be modified by dragging the ‘Scrollbar Length’ gizmo in the scene view.
- Slider - Bar that allows you to slide a button along it. When resizing exact length can be modified by dragging the ‘Scrollbar Length’ gizmo in the scene view.
- TextInput - When clicked, you can enter text into it via keyboard or touch keyboard. When resizing exact length can be modified by dragging the ‘Field Length’ gizmo in the scene view.
- ToggleButton - Toggles between on and off states.

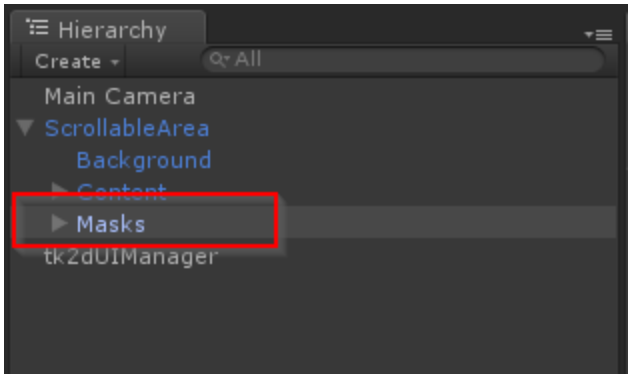
Advanced - Setting up Scrollable Area for 3D Objects.

Use this technique when you need to mix 3D and 2D objects in 2D Toolkit UI.

1. Follow instructions 1-6 in ‘Setting up UI’ tutorial
2. In the Project View, navigate to TK2DROOT/tk2d_UI_Demo/ControlPrefabs. Locate ScrollableArea prefab and drag it into the scene.

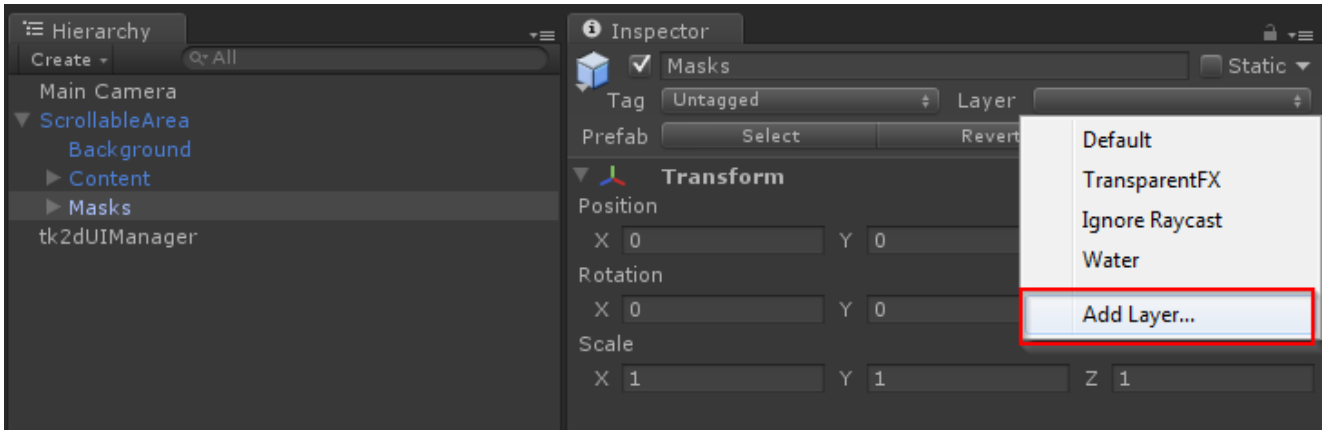


3. Click on ScrollableArea GameObject in the hierarchy and click the arrow to show children. Select the ‘Masks’ GameObject.

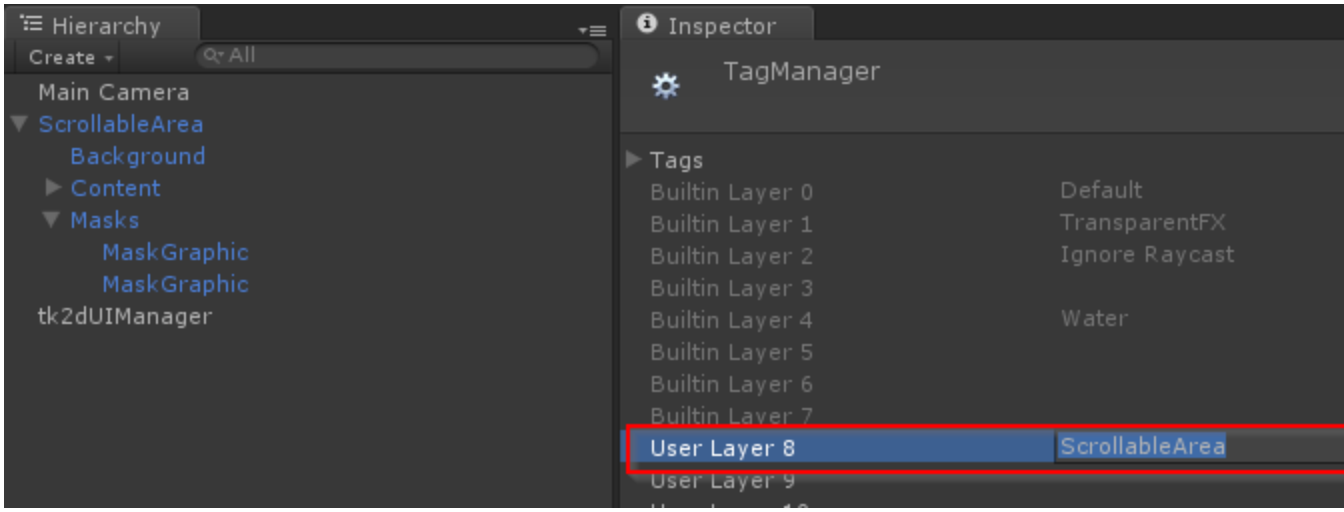


4. Select the mask objects under the ‘Masks’ GameObject, and switch the material on them to DepthMaskSolid.

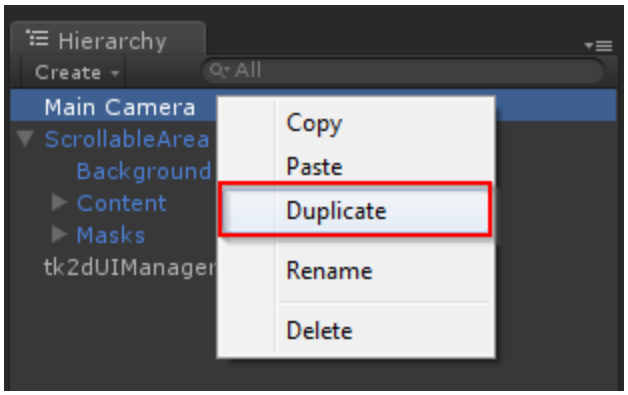
5. At the top of the inspector for the ‘Masks’ GameObject click the Layer dropdown and click Add Layers...



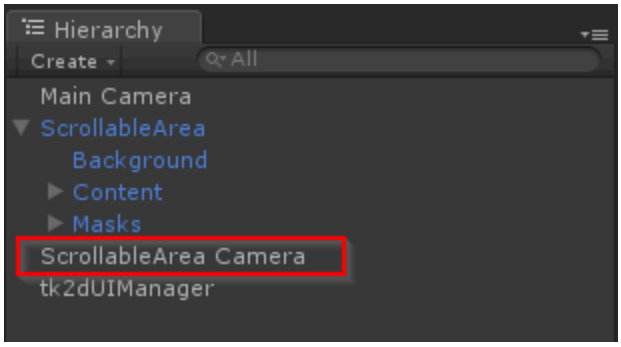
6. In the top layer (User Layer 8) type ‘ScrollableArea’. And then click back on the ‘Masks’ GameObject in the hierarchy. This should automatically set the ‘Masks’ GameObject and all of its childrens’ layer to ‘ScrollableArea’. Also the ‘Content’ GameObject above should also be set to ‘ScrollableArea’ layer along with its children.



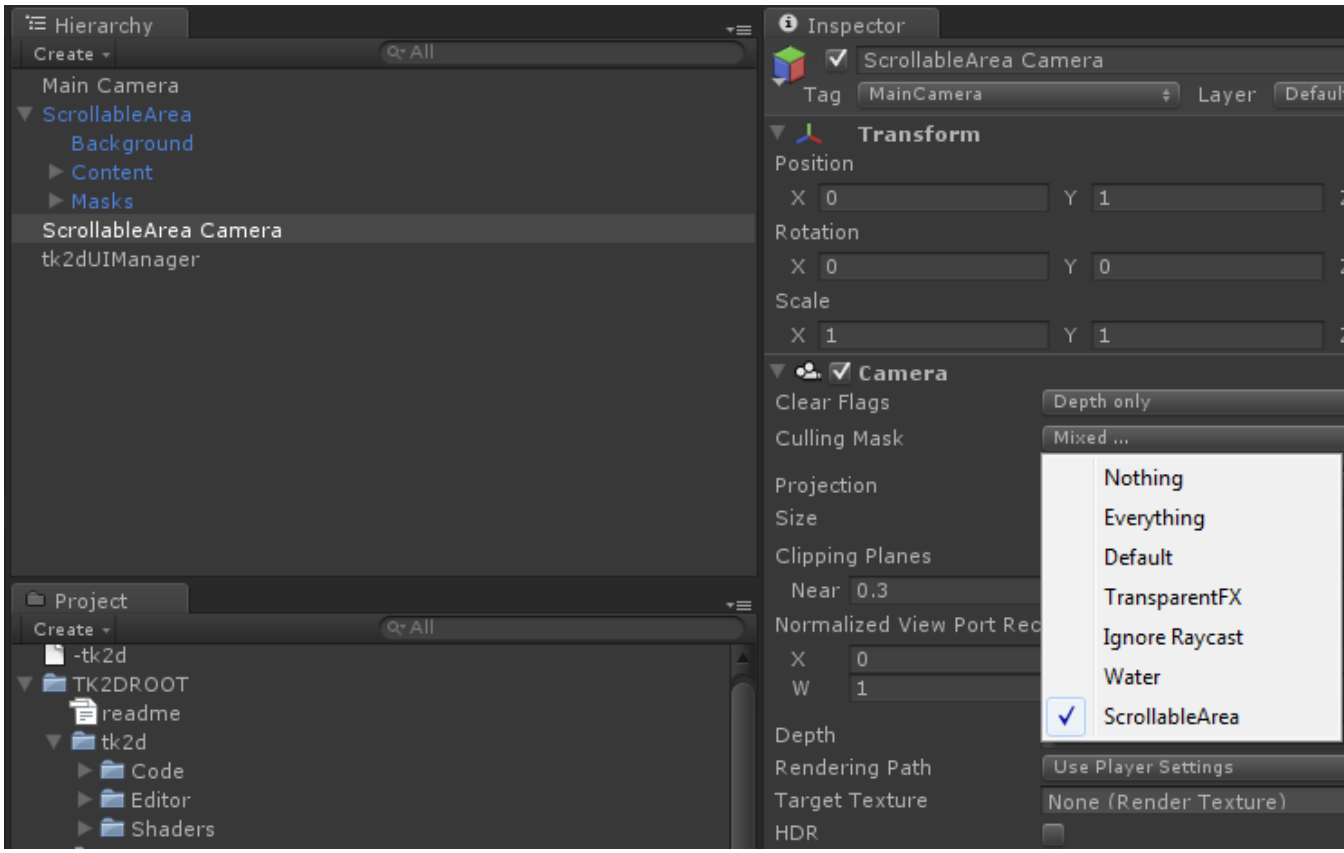
7. Right-click on the ‘Main Camera’ GameObject and hit duplicate.



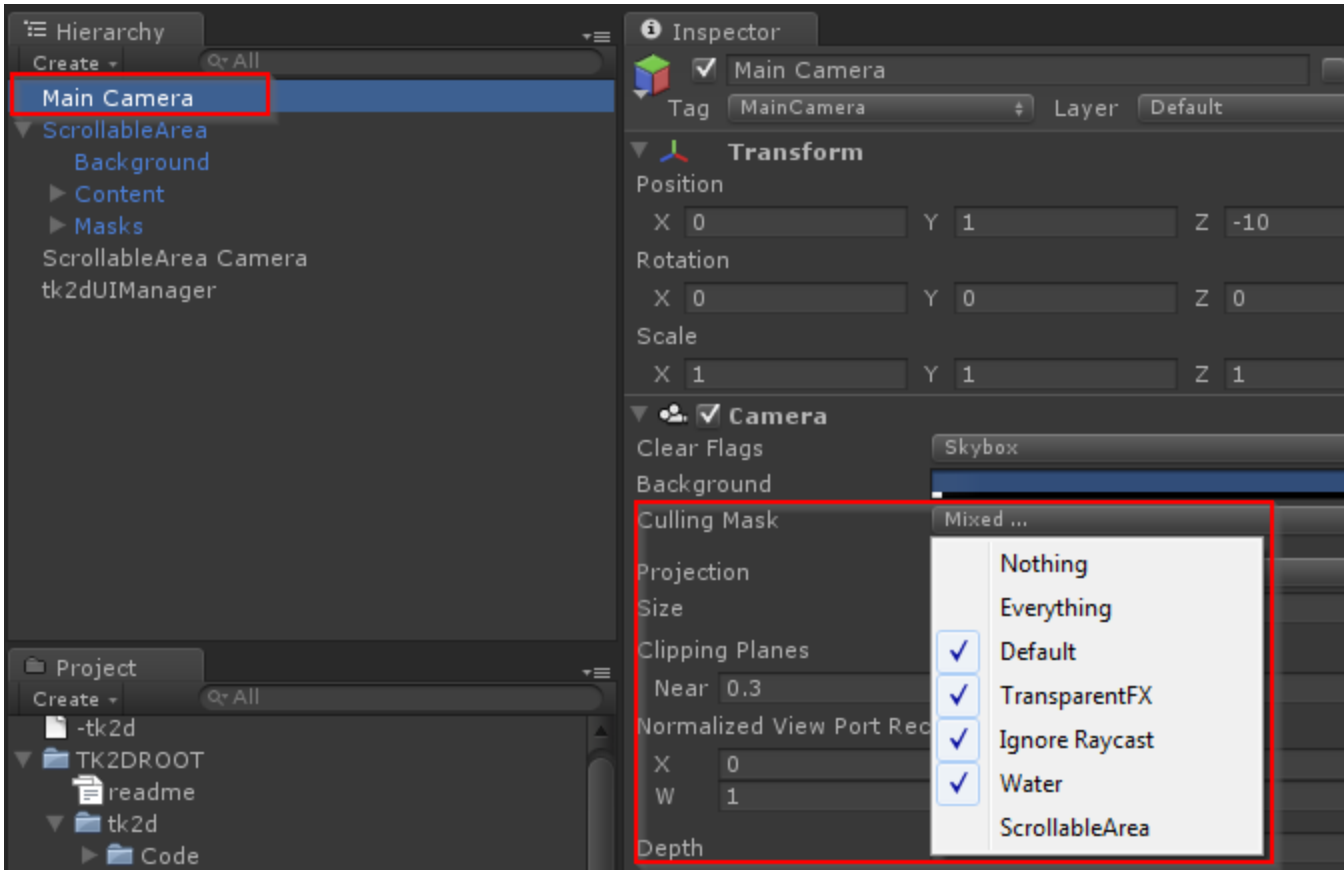
8. Rename one of the cameras to ‘ScrollableArea Camera’.



9. Select ‘ScrollableArea Camera’ in the hierarchy and in the inspector click the ‘Culling Mask’ dropdown. Deselect everything in the list except for ‘ScrollableArea’. Then click the ‘Clear Flags’ dropdown and select ‘Depth only’. Then set the ‘Depth’ field to 0. Last remove the ‘Audio Listener’ component (click the small dropdown on the right of the Audio Listener component and hit remove).

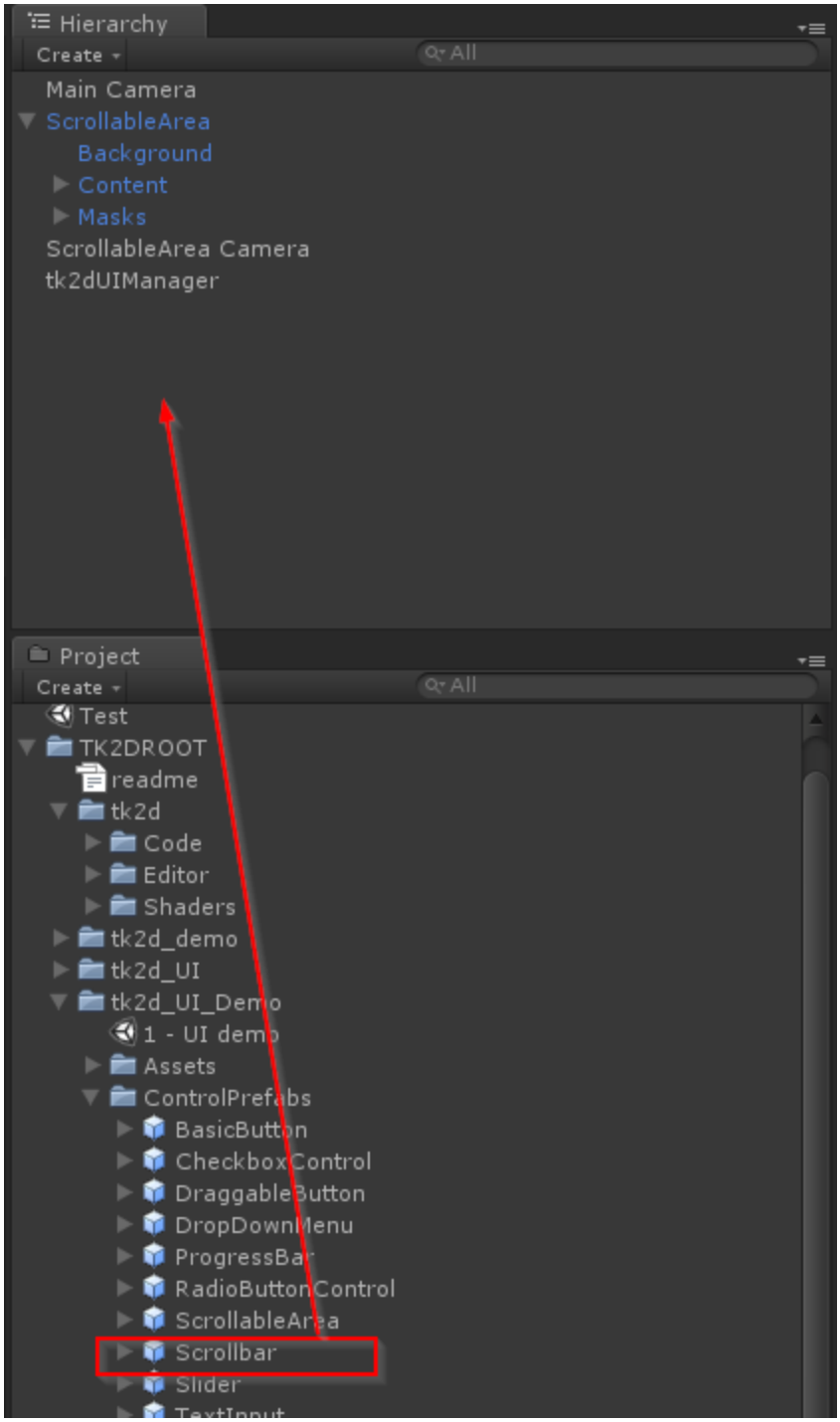


10. Select ‘Main Camera’ in the hierarchy and in the inspector click the ‘Culling Mask’ dropdown and deselect only ‘ScrollableArea’.



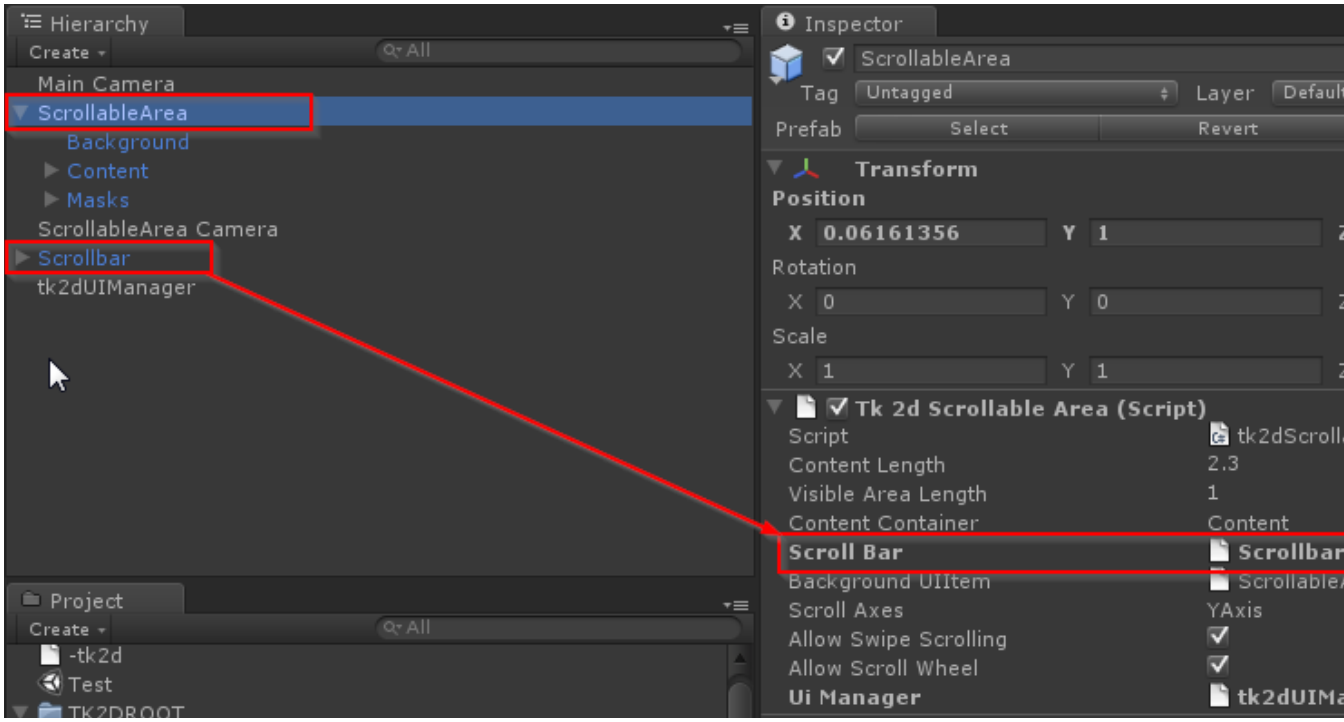
11. Hit play and the list should be scrollable by dragging and mouse wheel.

12. To add a scrollbar in the Project View, navigate to TK2DROOT/tk2d_UI_Demo/ControlPrefabs. Locate Scrollbar prefab and drag it into the



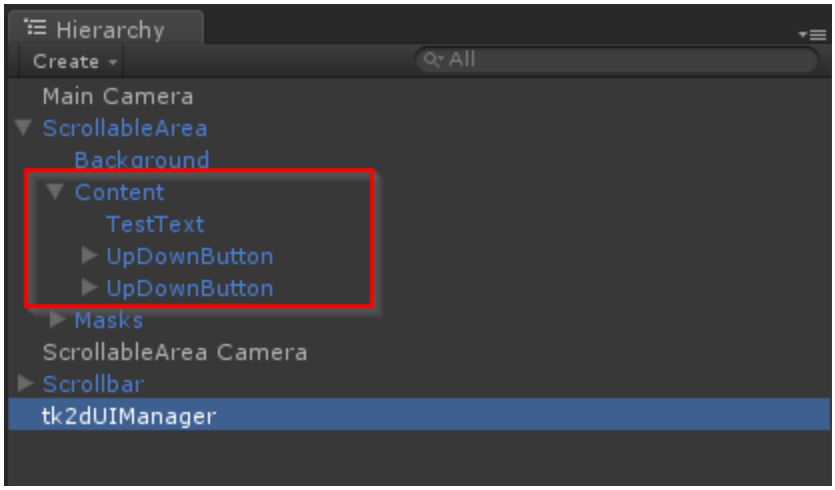
scene.

13. Select the ‘ScrollableArea’ GameObject in the hierarchy. Drag ‘Scrollbar’’ GameObject from the hierarchy into the ‘Scroll Bar’ field in the inspector.

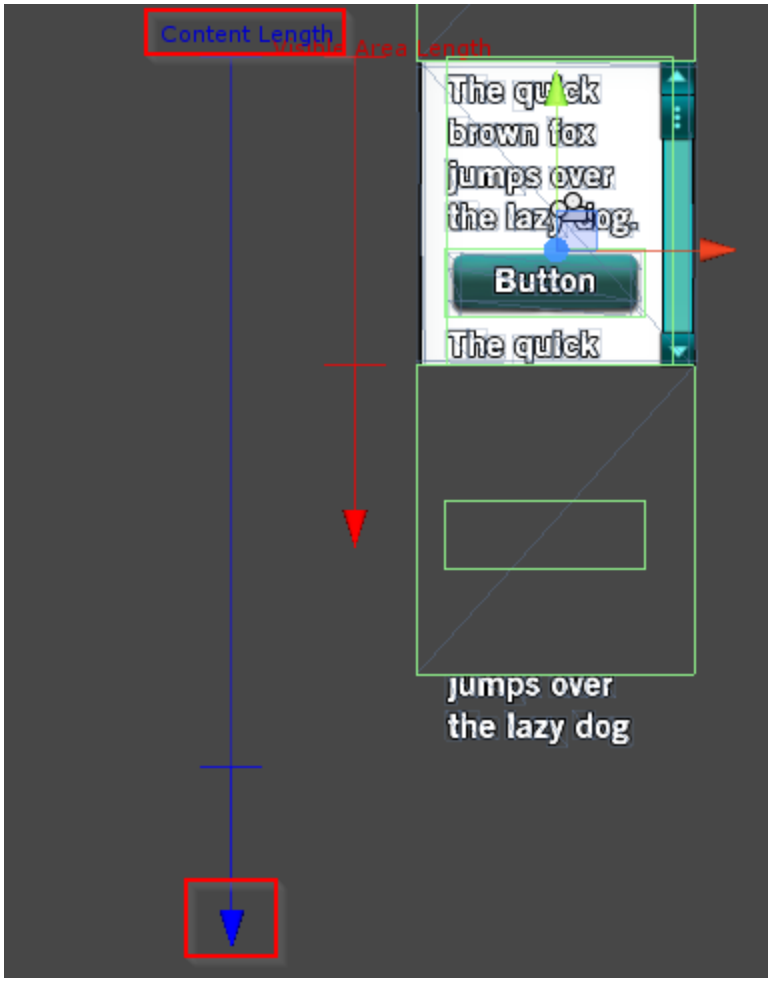


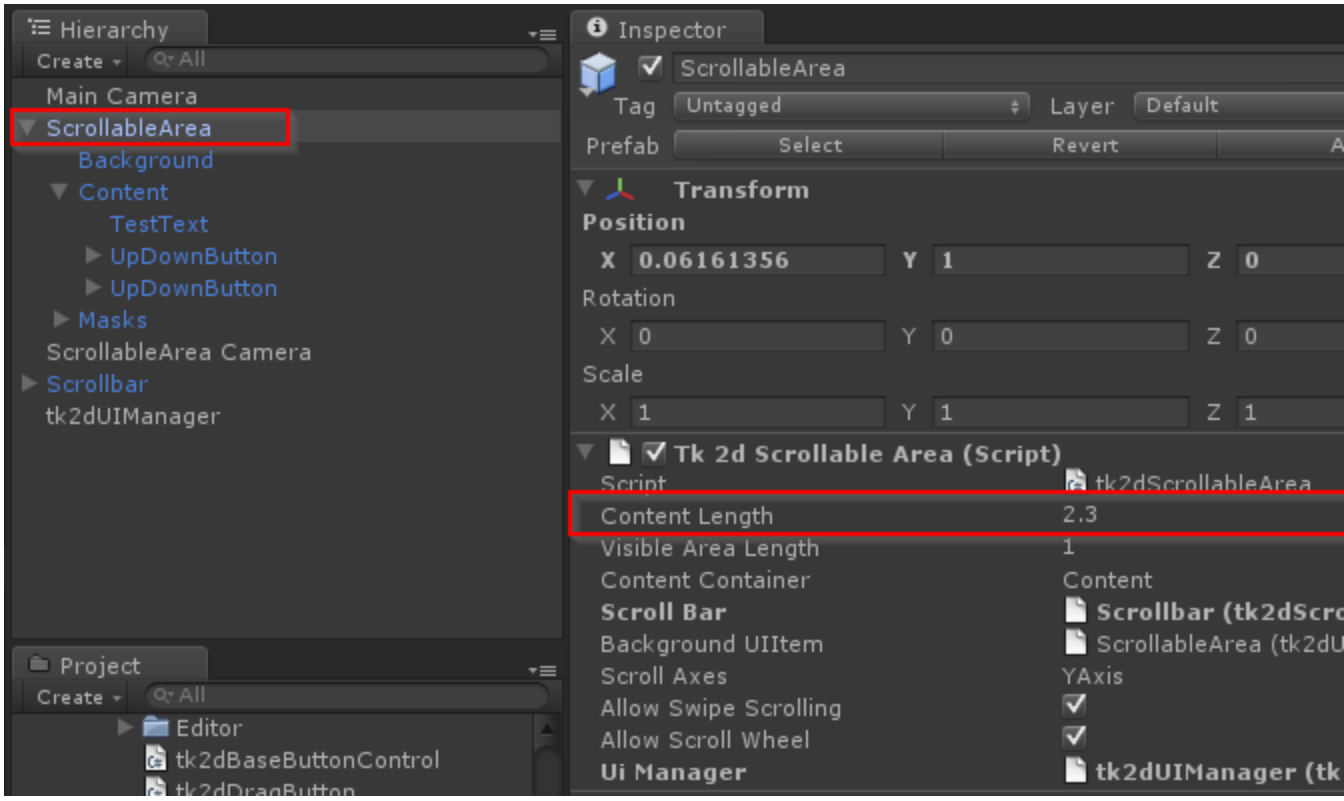
14. Hit play you should now be able to use the scrollbar to scroll through the list.

15. To edit content in the list. Select the ‘ScrollableArea’ GameObject in the hierarchy and click the arrow to show children. Select the ‘Content’ GameObject. All content must go in within this GameObject.



16. Then re-select the ‘ScrollableArea’ GameObject in the hierarchy and change the ‘Content Length’ field in the inspector to the length of new content added. This can also be edited in the Scene view by dragging the handle of the ‘Content Length’ gizmo.





Understanding UI System and Building your Own Components

Understanding tk2dUIManager

The whole UI system is governed by tk2dUIManager. It contains a loop that looks for touch and mouse events and does raycasts to look for colliders. If a raycast hits a collider, it will check to see if it has tk2dUIItem attached to it.

Understanding tk2dUIItem

tk2dUIItem is the base control for all interactive items. To make an item interactive simply add tk2dUIItem to any GameObject with a collider. Once this is complete whenever tk2dUIManager does a raycast (touch and/or mouse interaction) that hit that collider it will begin to fire events.

Listening for tk2dUIItem events

List of events you can listen for:

- **OnDown** - Pressed down
- **OnUp** - Unpressed, possibly on exit of collider area without releasing

- **OnClick** - Click - down and up while not leaving collider area
- **OnRelease** - After on down, when touch/click is released (this could be anywhere)
- **OnHoverOver** - On mouse hover over (only if using mouse and hover is enabled(tk2dUIManager.areHoverEventsTracked=true) and if multi-touch is disabled(tk2dUIManager.useMultiTouch=false))
- **OnHoverOut** - On mouse hover, leaving collider (only if using mouse and hover is enabled(tk2dUIManager.areHoverEventsTracked=true) and if multi-touch is disabled(tk2dUIManager.useMultiTouch=false))
- **OnDownUIItem** - Same as OnDown above, except returns this tk2dUIItem
- **OnUpUIItem** - Same as OnUp above, except returns this tk2dUIItem
- **OnClickUIItem** - Same as OnClick above, except returns this tk2dUIItem
- **OnReleaseUIItem** - Same as OnRelease above, except returns this tk2dUIItem
- **OnHoverOverUIItem** - Same as OnHoverOver above, except returns this tk2dUIItem
- **OnHoverOutUIItem** - Same as OnHoverOut above, except returns this tk2dUIItem

Example of listening to some events:

```
void OnEnable()
{
	uiItem.OnDown += ButtonDown;
	uiItem.OnClickUIItem += Clicked;
}

void ButtonDown()
{
	Debug.Log("Button Down");
}

void Clicked(tk2dUIItem clickedUIItem)
{
	Debug.Log("Clicked:" + clickedUIItem);
}

//Also remember if you are adding event listeners to events you need to also remove them:

void OnDisable()
{
	uiItem.OnDown -= ButtonDown;
	uiItem.OnClickUIItem -= Clicked;
}
```

Advanced tk2dUIItem

Within tk2dUIItem are some more advanced settings.

Child of Another UIItem
This will allow hierarchy parents to listen for events of this child. When checked it will look up the hierarchy and find the next tk2dUIItem and that will be it's ParentUIItem.

Register Events From Children
If a hierarchy child of this tk2dUIItem has 'Child of Another UIItem' set to true, this tk2dUIItem will receive all of its events. This allows for complex things like button within buttons systems.

Is Hover Events Enabled
By default hover events are disabled to increase performance. If you need hover events simply set this to true.

Send Message Target
While it is recommended you use events, there is built-in Send Message system in the inspector. Simply drag the GameObject (target) you wish to send a message to, into the 'Send Message Target' field. Once complete a list of fields will appear. Type the function name into the desired field. When this event occurs this function will be called via SendMessage.

Automatic Bounds Calculation
The tk2dUIItem has some additional functionality to help the editor best-guess a bounding collider. Bounds calculation recursively goes down the hierarchy until another collider is found. You can add objects to the "ignore bounds" array to ignore bounds of certain parts of the hierarchy. You can also consider additional hierarchies by adding them to the "extra bounds" array.

With these building blocks you are ready to build custom controls. Good Luck and have fun!