CT3536
(Games Programming using Unity3D)

Multi-Scene projects
User Interfaces (GUIs)

Multi-Scene Projects

- Certain types of game suit the approach we have been taking so far, i.e. instantiate and destroy all game objects as the game runs
- Other types of game may benefit from having multiple complex manually-configured scenes which are switched between as the game runs
  - They will still dynamically instantiate certain game objects such as enemies, bullets, and explosions
- The SceneManager class provides methods to load scenes (normally, unloading previous ones)

SceneManager

- https://docs.unity3d.com/ScriptReference/SceneManagement.SceneManager.html
- http://www.alanzucconi.com/2016/03/23/scene-management-unity-5/
- To load a scene:

```
using UnityEngine.SceneManagement;
SceneManager.LoadScene("scene4");
```

- When a new scene is loaded with this method, the previous one is unloaded (all its objects destroyed).
  - Any object which you want to survive this process should have DontDestroyOnLoad() called for it (see next slide)
- The game will freeze for a little bit while the new scene is being loaded and activated.
- It's possible to load a scene asynchronously and therefore display e.g. a loading bar while it loads, rather than freezing the game.. See:
- https://docs.unity3d.com/ScriptReference/SceneManagement.SceneManager.LoadSceneAsync.html

DontDestroyOnLoad( )

- https://docs.unity3d.com/ScriptReference/Object.DontDestroyOnLoad.html
- Makes the specified object not be destroyed automatically when loading a new scene.
- If the object is a component or game object then its entire transform hierarchy will not be destroyed either.
- E.g.:

```
using UnityEngine;
using System.Collections;

public class ExampleClass : MonoBehaviour {
    void Awake() {
        DontDestroyOnLoad(gameObject);
    }
}
```

- Often used for singletons such as GameManager, that can therefore be used to carry data between scenes

Unity UI

- Canvas
- Positioning UI elements
- Visual Components include:
  - Text
  - Image
- Interactive Components include:
  - Button
  - Toggle
  - InputField

Here's a good Unity UI tutorial:

- https://www.raywenderlich.com/149464/introduction-to-unity-ui-part-1

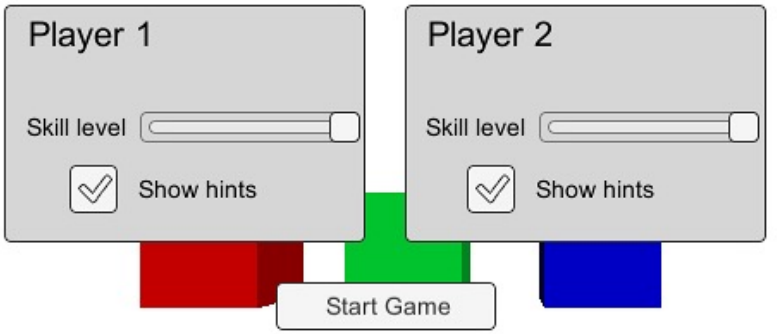
Canvas

- https://docs.unity3d.com/Manual/UICanvas.html
- The Canvas is a Game Object with a Canvas component on it, and all UI elements must be children of such a Canvas.
- Shown as a rectangle in the scene view
- UI elements in the Canvas are drawn in the same order they appear in the Hierarchy
- Canvases can be displayed in Screen Space or World Space (see next slide)

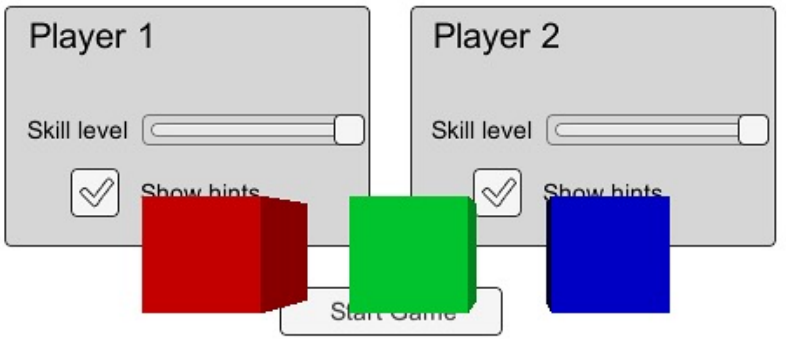
Canvas Render Modes

https://docs.unity3d.com/Manual/class-Canvas.html

- **Screen Space: Overlay**
- the Canvas is scaled to fit the screen and then rendered directly without reference to the scene or a camera
- If the screen's size or resolution are changed then the UI will automatically rescale to fit. The UI will be drawn over any other graphics such as the camera view.

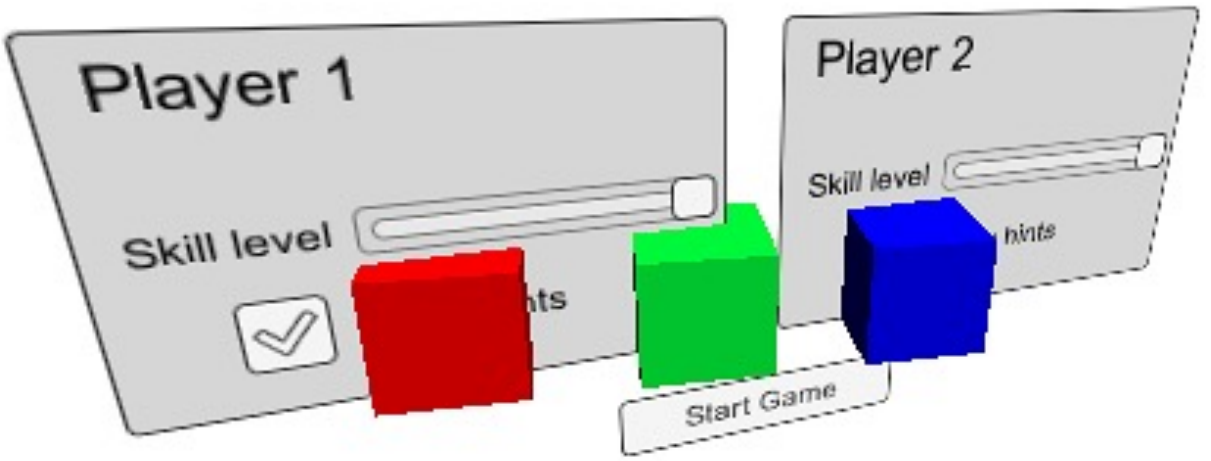


- **Screen Space: Camera**
- the Canvas is rendered as if it were drawn on a plane object some distance in front of a given camera.
- The onscreen size of the UI does not vary with the distance since it is always rescaled to fit exactly within the camera frustum



- Any 3D objects in the scene that are closer to the camera than the UI plane will be rendered in front of it

Canvas Render Modes



- **World Space**
- renders the UI as if it were a plane in the 3D scene
- the plane need not face the camera and can be oriented however you like
- The size of the Canvas can be set using its Rect Transform but its onscreen size will depend on the viewing angle and distance of the camera.
- Other scene objects can pass behind, through or in front of the Canvas.

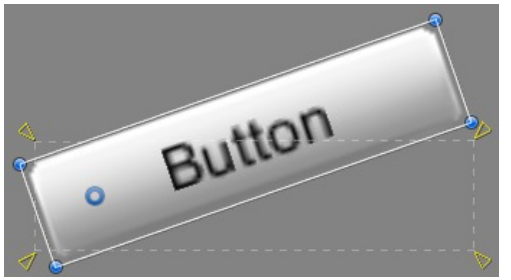
Positioning UI Elements

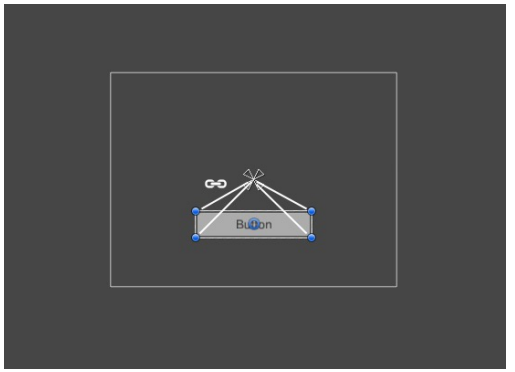
- https://docs.unity3d.com/Manual/UIBasicLayout.html
- Every UI element is represented as a rectangle for layout purposes. This rectangle can be manipulated in the Scene View using the Rect Tool in the toolbar. 

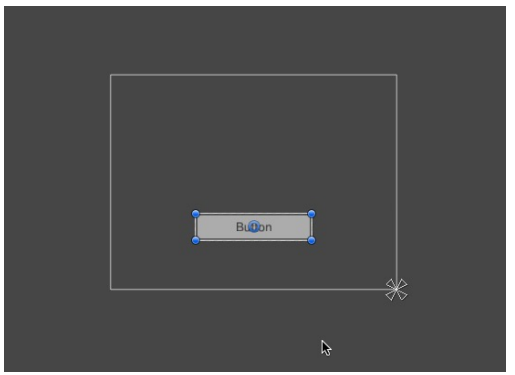
- For positioning and scaling, UI elements use **RectTransform** rather than Transform (which is what normal game objects use)
- This adds the ability to specify size directly, and to specify positioning relative to the screen edges .. which is very useful when dealing with varying screen sizes
- Continued next slide..

RectTransform
(also see previous slide)

- https://docs.unity3d.com/Manual/class-RectTransform.html
- Rotations, size, and scale modifications occur around the **pivot** so the position of the pivot affects the outcome of a rotation, resizing, or scaling
- RectTransforms include a layout concept called **anchors**, which are shown as four small triangular handles in the Scene View
- A child RectTransform can be anchored to its parent RectTransform in various ways. For example, the child can be anchored to the center of the parent, or to one of the corners.
- Continued next slide..

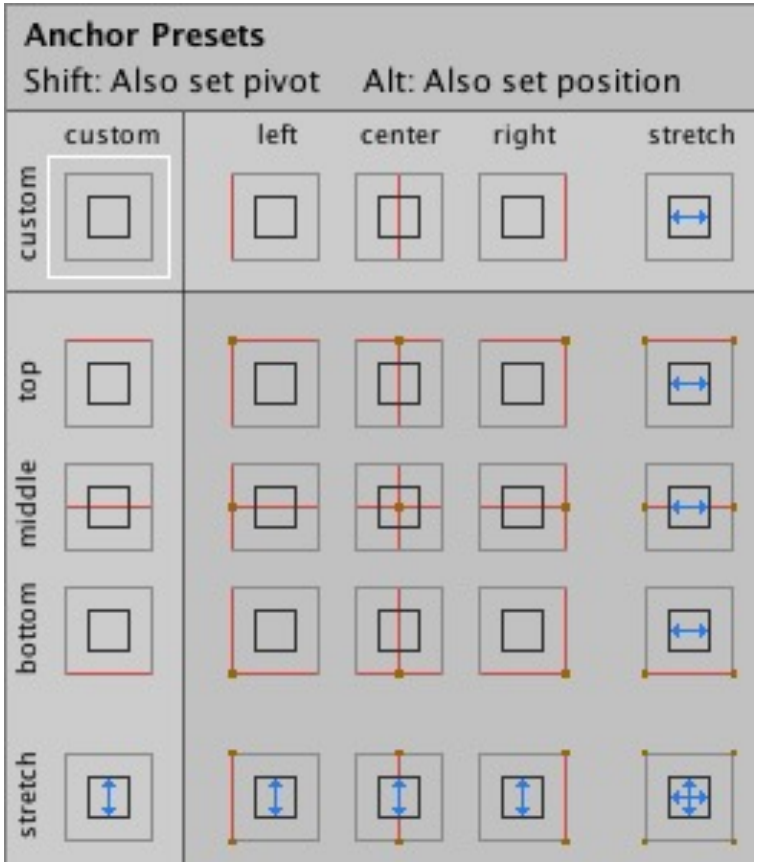






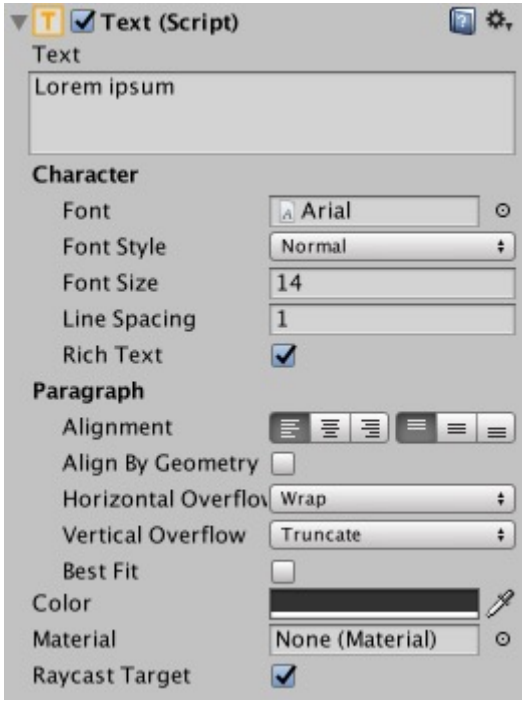
RectTransform
(also see previous slides)

- Anchor presets allow you to quickly choose from common anchor settings
- You can anchor the UI element to the sides or middle of the parent, or stretch together with the parent size. The horizontal and vertical anchoring is independent.



Text

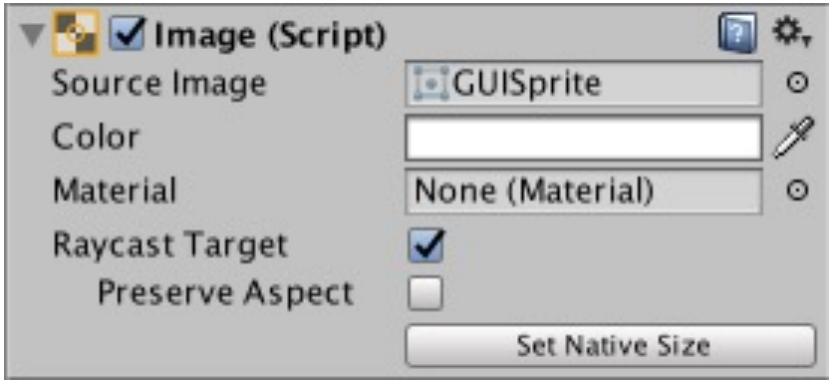
- https://docs.unity3d.com/Manual/UIVisualComponents.html
- https://docs.unity3d.com/Manual/script-Text.html
- For displaying text on the screen
- To change the displayed text at runtime, obtain a reference to your game object's <Text> component, and use that component's .text property




```
using UnityEngine.UI;

go.GetComponent<Text>().text = "Hello";
```

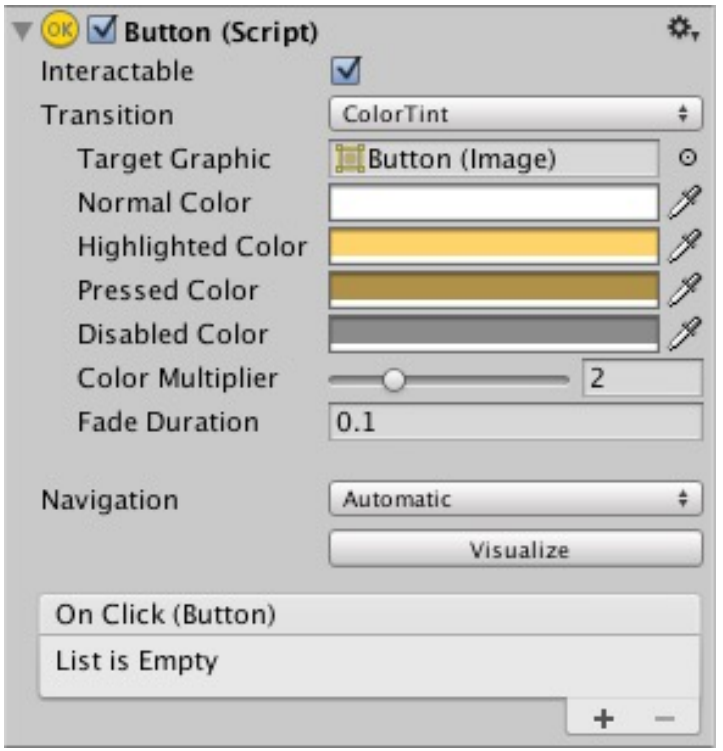
Image

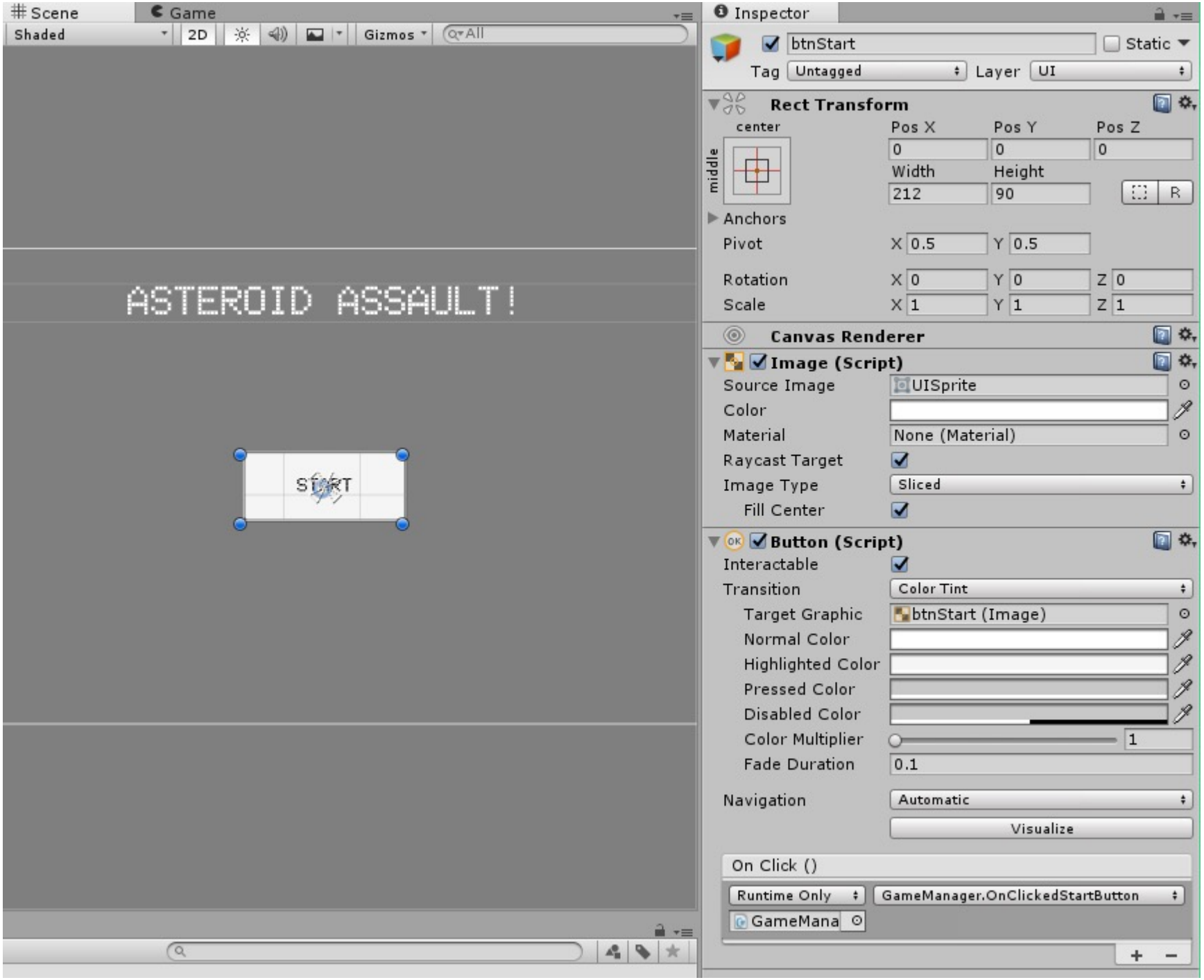
- https://docs.unity3d.com/Manual/UIVisualComponents.html
- https://docs.unity3d.com/Manual/script-Image.html
- An image (typically a png file in your project's assets) can be applied to the Image component under the SourceImage field, and its colour can be set in the Color field
- Raycast Target defines whether you want this UI element to consume mouse clicks



Button

- https://docs.unity3d.com/Manual/UIInteractionComponents.html
- https://docs.unity3d.com/Manual/script-Button.html
- Clickable button with colour properties allowing you to define how it reacts visually when clicked, etc.
- The text on the button is actually rendered using a child game object with a Text component. This is automatically created when you create a button.
- Buttons have an OnClick event list, which is how you bind a button click to a method call on one of your game's objects
- (See example on next slide)

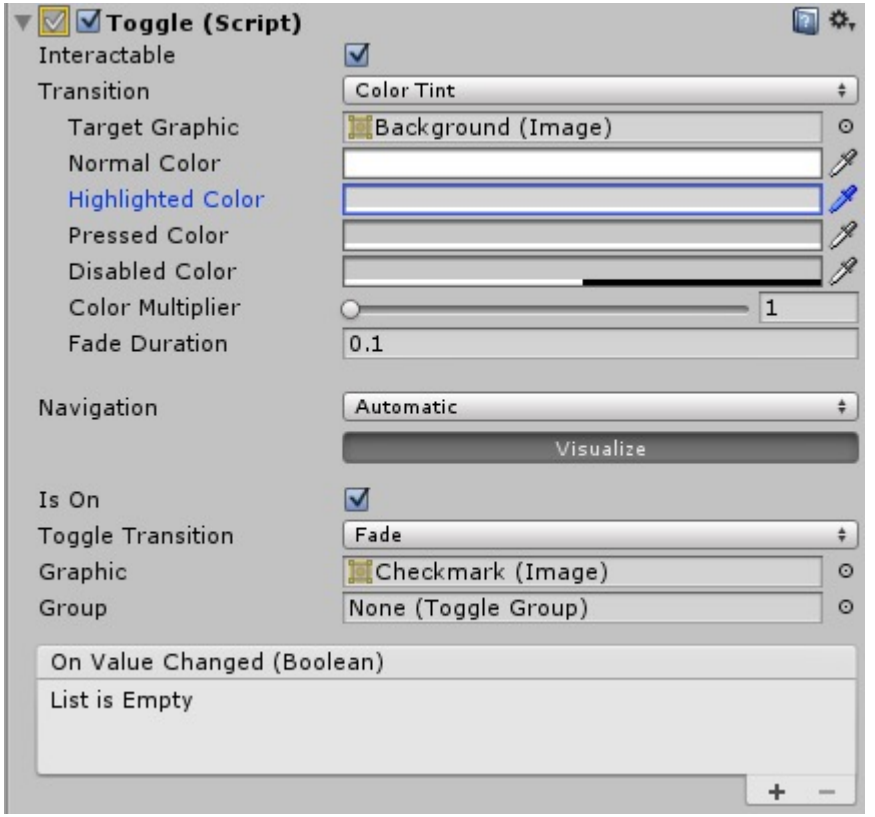



Scene
Game
Shaded
2D
Gizmos
All
ASTEROID ASSAULT!
START
Inspector
btnStart
Static
Tag
Untagged
Layer
UI
Rect Transform
center
middle
Pos X
Pos Y
Pos Z
0
0
0
Width
Height
212
90
R
Anchors
Pivot
X 0.5
Y 0.5
Rotation
X 0
Y 0
Z 0
Scale
X 1
Y 1
Z 1
Canvas Renderer
Image (Script)
Source Image
UISprite
Color
Material
None (Material)
Raycast Target
Image Type
Sliced
Fill Center
Button (Script)
Interactable
Transition
Color Tint
Target Graphic
btnStart (Image)
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier
1
Fade Duration
0.1
Navigation
Automatic
Visualize
On Click ()
Runtime Only
GameManager.OnClickedStartButton
GameMana

Toggle

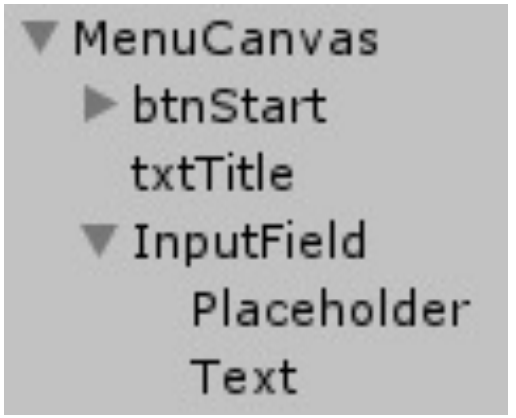
- https://docs.unity3d.com/Manual/UIInteractionComponents.html
- https://docs.unity3d.com/Manual/script-Toggle.html
- This is a checkbox
- Has an OnChanged event
- Read/write the .isOn property (Boolean) to get/set the checkbox state

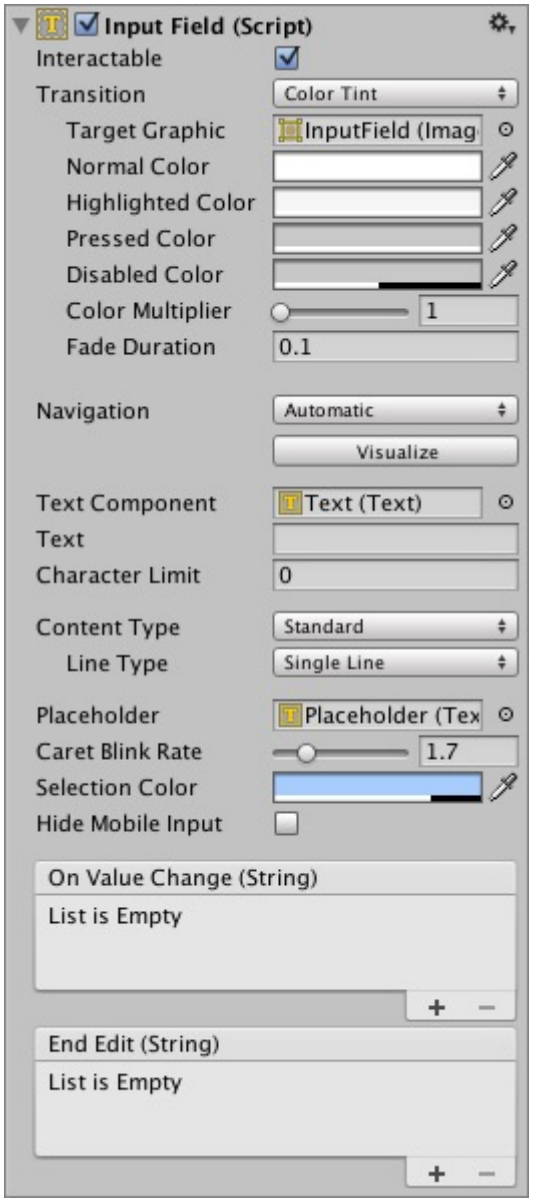
```
using UnityEngine.UI;

if (go.GetComponent<Toggle>().isOn)
    ApplyOption();
```



InputField

- https://docs.unity3d.com/Manual/UIInteractionComponents.html
- https://docs.unity3d.com/Manual/script-InputField.html
- A text-box into which the user can type
- Read/write the .text property to get/set the contents of the input field (as a string)
- The text (and, optionally, a placeholder for when the text is empty) are actually rendered through child game objects with Text components. These are automatically created when you create an InputField





TextMeshPro

TextMeshPro provides much more flexible and crisp-looking text rendering than the normal Text component.

- Renders the text as a dynamically-built 3D mesh rather than as a bitmap (which is what normal Unity Text uses)

The TMP plugin is included as part of projects when you create them

You must include this line at the top of any C# files where you need to refer to TextMeshPro objects:

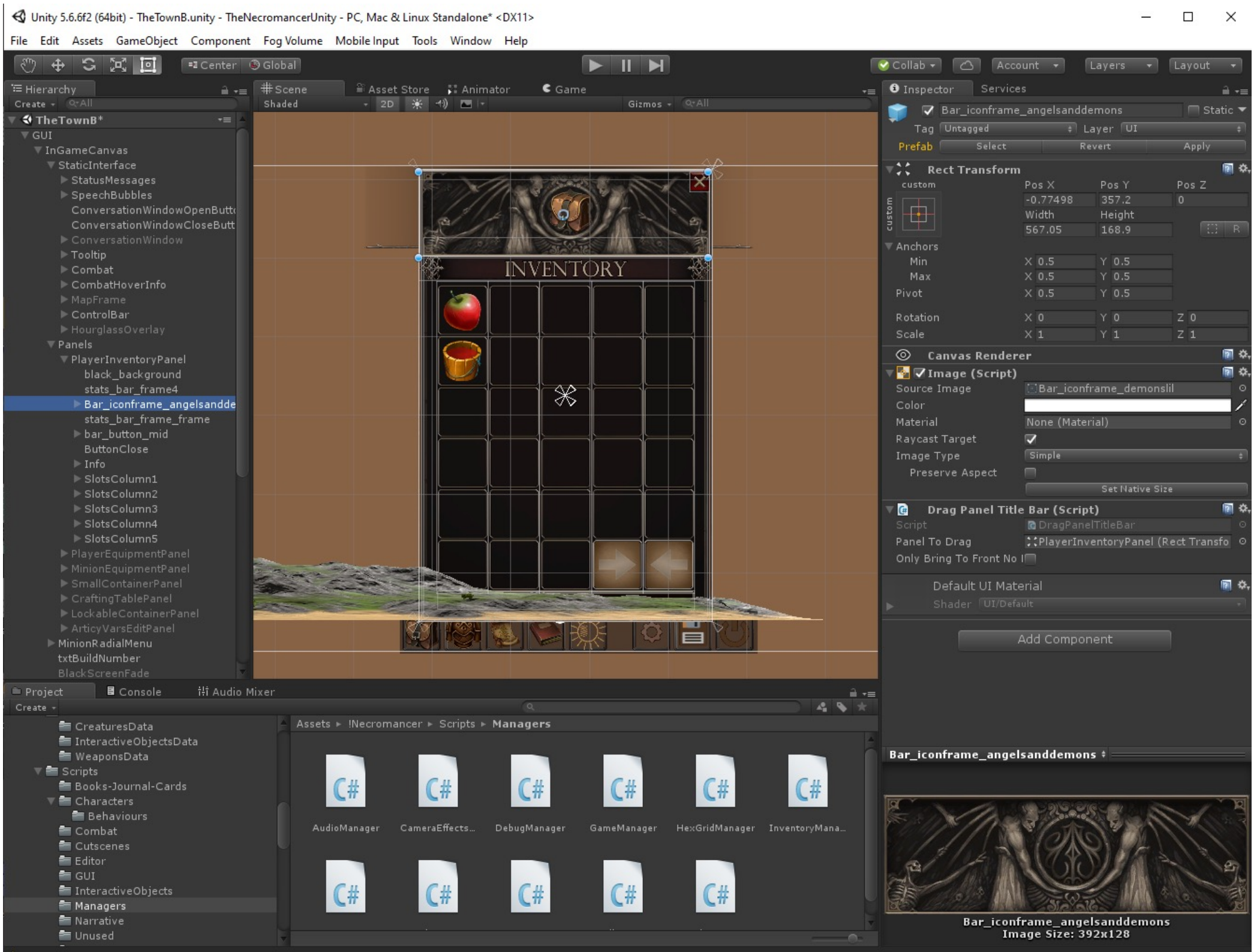
```
using TMPro;
```

Some GUI Examples from..

http://www.psychicsoftware.com/thenecromancer/
https://store.steampowered.com/app/1315320/The_Necromancers_Tale/

An RPG (roleplaying game) I'm working on

Draggable Windows


Unity 5.6.6f2 (64bit) - TheTownB.unity - TheNecromancerUnity - PC, Mac & Linux Standalone* <DX11>
File Edit Assets GameObject Component Fog Volume Mobile Input Tools Window Help
Hierarchy
TheTownB*
GUI
InGameCanvas
StaticInterface
StatusMessages
SpeechBubbles
Tooltip
Combat
CombatHoverInfo
MapFrame
ControlBar
HourglassOverlay
Panels
PlayerInventoryPanel
black_background
stats_bar_frame4
Bar_iconframe_angelsandde
stats_bar_frame_frame
bar_button_mid
ButtonClose
Info
SlotsColumn1
SlotsColumn2
SlotsColumn3
SlotsColumn4
SlotsColumn5
PlayerEquipmentPanel
MinionEquipmentPanel
SmallContainerPanel
CraftingTablePanel
LockableContainerPanel
ArticyVarsEditPanel
MinionRadialMenu
txtBuildNumber
BlackScreenFade
Scene
Asset Store
Animator
Game
INVENTORY
Inspector
Services
Bar_iconframe_angelsanddemons
Static
Tag Untagged
Layer UI
Prefab Select Revert Apply
Rect Transform
Pos X -0.77498
Pos Y 357.2
Pos Z 0
Width 567.05
Height 168.9
Anchors
Min X 0.5 Y 0.5
Max X 0.5 Y 0.5
Pivot X 0.5 Y 0.5
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Canvas Renderer
Image (Script)
Source Image Bar_iconframe_demonslil
Color
Material None (Material)
Raycast Target
Image Type Simple
Preserve Aspect
Set Native Size
Drag Panel Title Bar (Script)
Script DragPanelTitleBar
Panel To Drag PlayerInventoryPanel (Rect Transfo
Only Bring To Front No I
Default UI Material
Shader UI/Default
Add Component
Project
Console
Audio Mixer
Assets ▸ !Necromancer ▸ Scripts ▸ Managers
CreaturesData
InteractiveObjectsData
WeaponsData
Scripts
Books-Journal-Cards
Characters
Behaviours
Combat
Cutscenes
Editor
GUI
InteractiveObjects
Managers
Narrative
Unused
AudioManager
CameraEffects...
DebugManager
GameManager
HexGridManager
InventoryMana...
Bar_iconframe_angelsanddemons
Bar_iconframe_angelsanddemons
Image Size: 392x128

```
using System.Collections;
using UnityEngine;
using UnityEngine.UI;
using UnityEngine.EventSystems;

public class DragPanelTitleBar : MonoBehaviour, IPointerDownHandler, IBeginDragHandler, IDragHandler, IEndDragH
andler {

    // inspector settings
    public Transform panelToDrag;
    //

    private Vector3 offsetFromMouse;

    public void OnPointerDown(PointerEventData eventData) {
        // bring panel to the front of all panels
        panelToDrag.SetAsLastSibling();
    }

    public void OnBeginDrag(PointerEventData eventData) {
            offsetFromMouse = panelToDrag.position - Input.mousePosition;
    }

    public void OnDrag(PointerEventData eventData) {
            panelToDrag.position = Input.mousePosition + offsetFromMouse;
    }

    public void OnEndDrag(PointerEventData eventData) {
    }
}
```

Note the additional interfaces related to mouse events

And the callbacks which are enabled by this

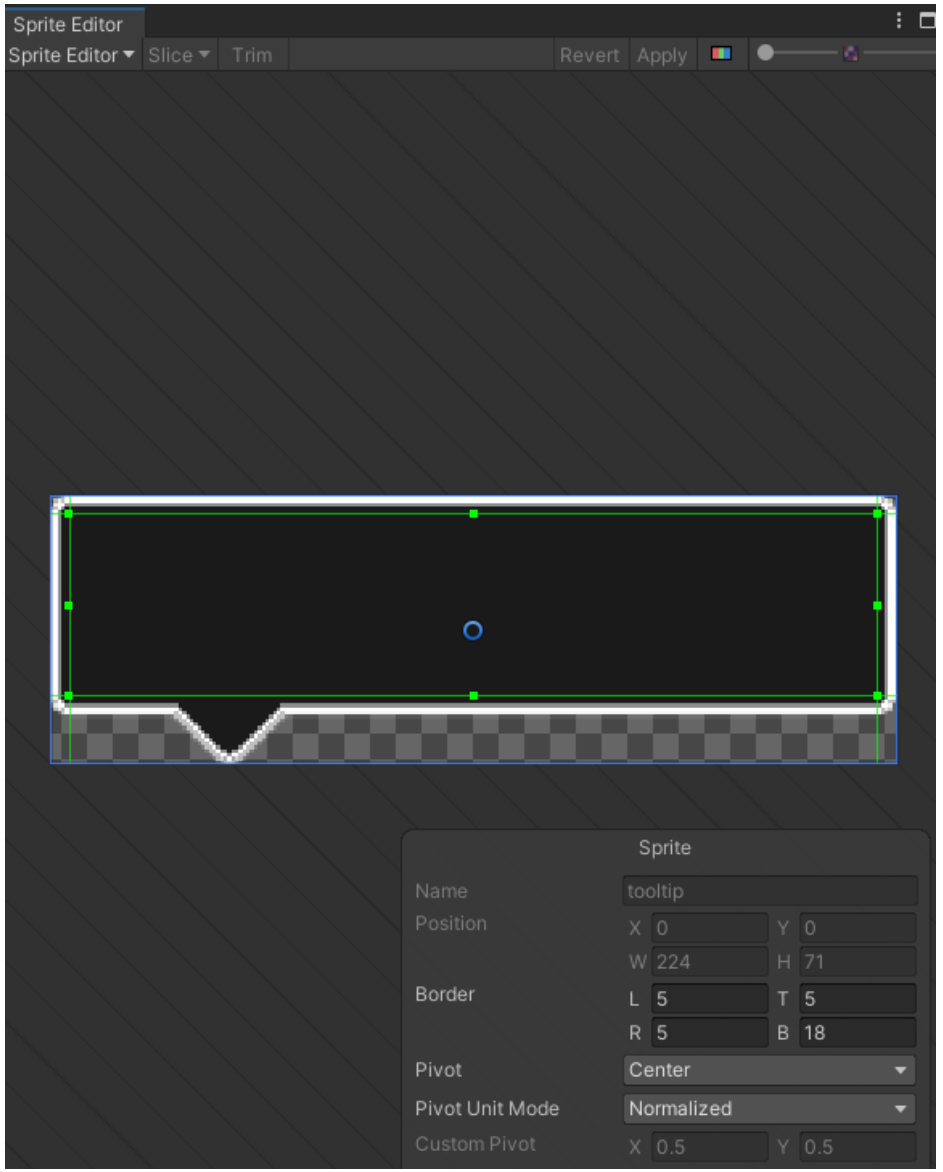
Example: Tooltips

The ‘tooltip speech bubble’ sprite is configured as a ‘9 slice’ image (see next slide)

9-Sliced Images

Allows resizing of an image while staying crisp at the edges

Click 'Sprite Editor' button on Texture asset



Tooltip GUI object (see script below)

TooltipWorldObject

attached to each 3D game object which needs a hover-mouse-over tooltip
(e.g. the Bakery Sign 3D object in the picture above)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.EventSystems;

public class TooltipWorldObject : MonoBe
haviour {

    // inspector
    public string tooltipMsg;
    //

    private bool mouseIsOverMe = false;

    void OnDisable() {
        if (mouseIsOverMe)
            OnMouseExit();
    }
```

Any GameObject with a collider receives this callback

```
    void OnMouseEnter() {
        if (EventSystem.current.IsPointerOverGa
meObject()) // mouse is over a UI element
            return;

        mouseIsOverMe = true;

        Tooltip.SetTooltipAtMouse(tooltipMsg);
    }

    void OnMouseExit() {
        mouseIsOverMe = false;
        Tooltip.Hide();
    }
}
```

Tooltip

attached to a singleton GUI Image object with a nested (child) TextMeshPro text object

```
using System.Collections;
using UnityEngine;
using TMPro;

public class Tooltip : MonoBehaviour {
    // inspector settings
    public TMP_Text txtTooltip;
    //

    private static RectTransform rt;
    public static Tooltip instance;

    void Start () {
        instance = this;
        rt = GetComponent<RectTransform>();
        Hide();
    }

    public static void SetTooltipAtMouse(string txt) {
        instance.txtTooltip.text = txt;
        instance.gameObject.SetActive(true);
        instance.Update();
    }

    public static void Hide() {
        instance.gameObject.SetActive(false);
    }

    void Update() {
        rt.position = Input.mousePosition;
    }
}
```