EXPERIMENT NO.6

Aim: Develop a simple UI menu with Images, Canvas, Sprites and Buttons

Theory: There are several frameworks and libraries across different programming languages that you can use to create a simple UI menu with images, canvas, sprites, and buttons. Here are some popular ones:

1. Unity (C#)

Unity is widely used for game development and interactive UI design.

- Canvas: Used for placing UI elements.
- Images, Sprites: Managed through the Image component and Sprite Renderer.
- Buttons: Interactable UI elements with onClick events.

2. HTML5 and JavaScript with Canvas

HTML5 Canvas and JavaScript are used for web-based interactive applications.

- Canvas: HTML5 <canvas> element.
- Images, Sprites: Drawn using the CanvasRenderingContext2D methods like drawImage().
- Buttons: HTML <button> elements with event listeners.

3. Qt (C++)

Qt is a powerful framework for cross-platform application development.

- Canvas: QGraphicsView and QGraphicsScene for drawing.
- Images, Sprites: QGraphicsPixmapItem for images.
- Buttons: QPushButton for interactive buttons.

4. WPF (C#)

Windows Presentation Foundation is used for Windows desktop applications.

- Canvas: Canvas control in XAML.
- Images, Sprites: Image control.
- Buttons: Button control with Click event handlers.

5. React Native (JavaScript)

React Native is used for mobile app development.

- Canvas: Not directly available, but libraries like react-native-canvas can be used.
- Images, Sprites: Image component.
- Buttons: Button component or Touchable components for more customization.

6. Godot (GDScript)

Godot is an open-source game engine with its own scripting language, GDScript.

- Canvas: Control node for UI elements.
- Images, Sprites: Sprite node for images.
- Buttons: Button node with signal connections.

7. Flutter (Dart)

Flutter is used for cross-platform mobile and web applications.

- Canvas: CustomPainter for drawing on a canvas.
- Images, Sprites: Image widget.
- Buttons: RaisedButton, FlatButton, or IconButton.

8. Tkinter (Python)

Tkinter is a standard GUI toolkit for Python.

- Canvas: Canvas widget.
- Images, Sprites: PhotoImage or Canvas.create_image.

- Buttons: Button widget.

9. SwiftUI (Swift)

SwiftUI is used for building UI for iOS and macOS applications.

- Canvas: Canvas view.
- Images, Sprites: Image view.
- Buttons: Button view.

10. JavaFX (Java)

JavaFX is used for building rich internet applications.

- Canvas: Canvas class.
- Images, Sprites: ImageView class.
- Buttons: Button class.

Each of these frameworks and libraries has its own strengths and is suited to different types of projects.

Problem Statement:

Develop a simple UI (User Interface) menu with images, canvas, sprites & button. Write a C# program to interact with UI menu through VR trigger button such that on each successful trigger interaction display a score on scene.

Steps:

VR by using XR Interaction Toolkit:

1. Unity has created a package called "XR Interaction Toolkit" by which we can create VR enabled applications for a wide range of VR Headsets.

2. To install the XR Interaction Toolkit package go to Window -> Package Manager.

3. Click on the "Packages:Unity Project" dropdown in the top left corner & select "Unity Registry".

4. Click on the plus icon on the top left corner of the package manager window & click "Add package by name" & type the following: com.unity.xr.interaction.toolkit

5. After installation, update the package if necessary.

6. Additionally search & install these packages if they

aren't already installed:

- XR Plugin Management
- Oculus XR Plugin

7. Unity might ask to restart the project, click yes if it

does.

VR by using XR Interaction Toolkit:

1. Now close the package manager window & go to Edit -> Project settings.

2. Click on XR Plug In Management Tab in the bottom left.

3. On the right side, Check the oculus checkbox as said in the Image.

Creating the XROrigin:

1. The XR Origin represents the center of worldspace in an XR scene.

2. The purpose of the XR Origin is to transform objects and trackable features to their final position, orientation, and scale in the Unity scene. It specifies an Origin, a Camera Floor Offset Object, and a Camera.

3. In the hierarchy, right click -> XR -> Device Based - > XROrigin.

4. Set the Transform position of the XROrigin at 0, 0, 0. Set the Tracking Origin Mode field of the XR Origin component to Floor.

5. Expand the hierarchy of XROrigin (Tip: left arrow besides the object in Hierarchy).

6. Select the LeftHand Controller & remove the XR Ray Interactor, XR Interactor Line Visual, Line Renderer components from it. Add XR Direct Interactor & Sphere Collider to it. Set the Sphere Collider values acc to the image. Do the same for the RightHand Controller.

Writing the GameManager Script:

1. Create a new Folder "Experiment 6" & then a folder called "Scenes" inside it. Duplicate Exp 5 scene by clicking on it in the Project window & pressing Ctrl + D. Remove the sphere & table gameobjects.

2. Move the Exp 6 scene to the Experiment 6's Scenes folder.

3. Create a "GameManager" script inside the Experiment 6 -> Scripts folder.

4. Double click to open it in Visual Studio.

5. Write the script according to the image:

6. Read the comments to understand the script.

Creating the UI:

1. In the hierarchy, right click in the empty area UI -> Canvas.

2. Create a panel game object as a child of it after right clicking on the Canvas gameobject. Set the color of the panel to black with the alpha (transparency) set to 0.5.

3. Now create a Button - TextMeshPro & Text - TextMeshPro under the Canvas gameObject.

4. Create 2 image gameObjects. (UI -> Images) under the Canvas gameObject.

Positioning the UI Elements:

1. Set the text of the TextMeshPro - Text to "Score :0".

2. Create a folder named "Images" under the experiment's folder in the Project view. To set the images of the 2 image gameobjects, import 2 images from outside the Unity project into the project.

3. Click on all the images in the Images folder & set it's texture type to "Sprite (2D and UI)".

4. Now in the hierarchy, select one of the Image gameObjects & set it's Source Image property to the new images imported by clicking on the small circle besides the field & then searching for the image. Do the same for the other image gameobject.

5. Set the position of the buttons & Images through it's RectTransform to look like the Game view image below.

Interacting with the UI Elements:

1. To interact with the UI elements, click on the canvas gameobject & in the inspector window. Then right click on the Graphic Raycaster component & select Remove Component. Add Tracked Graphic Raycaster to it.

2. Create a new empty GameObject in the hierarchy by right click-> Create Empty. Name it "GameManager".

3. Add the script we created earlier - GameManager as a component to it. Set ScoreText field to the Score gameobject.

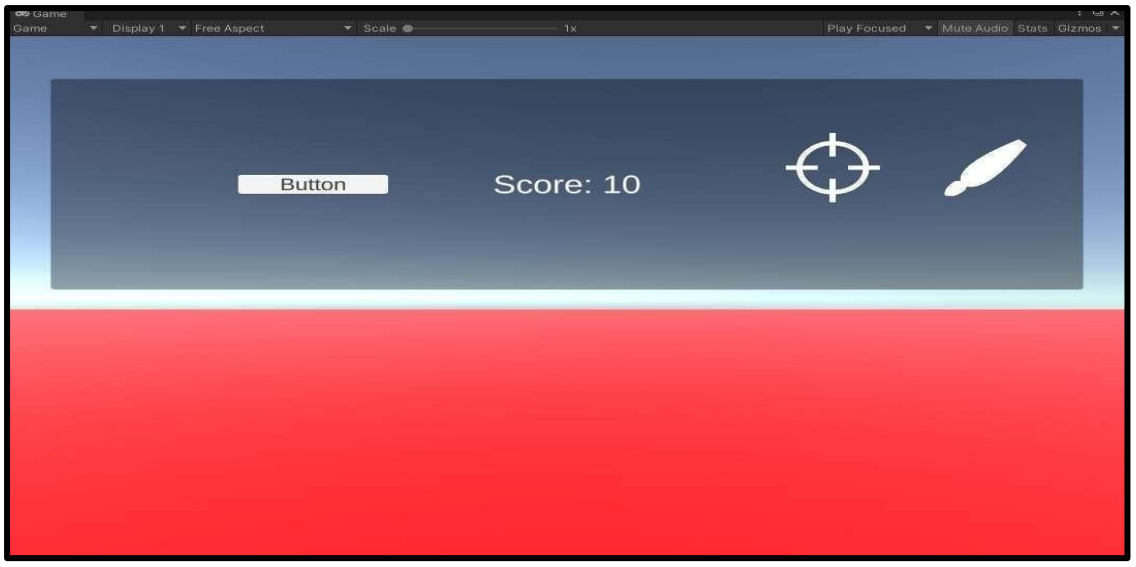
4. Select the 2 Image GameObjects & Add Component- Event Trigger. Select a single image gameobject.

5. Click the "Add New Event Type" button on the bottom of the Event Trigger component & select Pointer Down option. Click on the plus icon on the bottom of Pointer Down event.

6. Drag & drop the GameManager from the hierarchy into the "None (Object)" field created. From the dropdown select GameManager -> OnUIClick()

7. Do the same thing for the button OnClick event.

Output:



Conclusion:

What are the different elements that you can use in your UI?

In a User Interface (UI), various elements are used to create interactive and functional designs. Here are the main types of UI elements:

1. Input Controls:

- Buttons: Trigger actions (e.g., submit, cancel).
- Text Fields: Allow users to input text.
- Checkboxes: Enable selection of one or multiple options.
- Radio Buttons: Allow selection of one option from a set.
- Dropdown Menus: Provide a list of options for users to choose from.

- Sliders: Adjust values within a range.

- Toggles/Switches: Toggle between two states (on/off).

2. Navigational Components:

- Navigation Bar: Guides users to different sections.

- Tabs: Allow users to switch between content sections.

- Breadcrumbs: Show users their navigation path.

- Pagination: Split content into multiple pages.

- Sidebar/Menu: Provide access to different features or categories.

3. Informational Components:

- Tooltips: Provide additional information when users hover over elements.

- Notifications/Alerts: Display important messages or alerts.

- Progress Bars: Show progress of tasks.

- Modals/Pop-ups: Display additional information or ask for input without navigating away from the current page.

- Message Boxes: Show system messages or confirmations.

4. Containers:

- Cards: Organize content into sections or groups.

- Accordion: Expand/collapse sections to show/hide content.

- Panels: Group content or UI elements for structure.

- Lists: Organize multiple items vertically or horizontally (e.g., bullet lists, number lists).

5. Visual Elements:

- Icons: Represent actions or information visually.
- Images: Add visual content or backgrounds.
- Videos: Embed visual media for dynamic interaction.

These elements collectively enhance the usability, accessibility, and aesthetics of a UI.