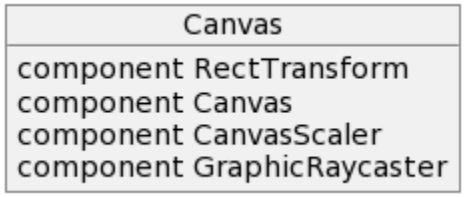
Canvas Guide

Overview

Canvas is a Game Object which should hold all of the UI elements. When you add UI elements to your game they will automatically be children of the canvas, allowing you to easily position and manage them. The canvas is used to organize and provide a layout for UI elements like buttons. It is responsible for making information quickly readable, meaning that Canvas contents can vary wildly between games. In DOOM, the canvas is used to communicate information to the player such as health, weapon choice, ammo, and other in-game events like item pickups.

(Source: DOOM Wiki)
https://doomwiki.org/wiki/Heads-up_display
https://doomwiki.org/wiki/Status_bar
https://doomwiki.org/wiki/Ammo
https://doomwiki.org/wiki/Weapon
(the following videos served as a reference for the HUD in action)
https://youtu.be/8mEP4cflrd4
https://youtu.be/e5UD5QQH_as

GameObjects



Settings

- Rect Transform
    - Anchors: Changes how UI elements are attached and positioned relative to the canvas. Useful for making your UI responsive and flexible across various screen sizes and resolutions.
    - Pivot: Defines the point which UI elements are rotated around. Useful if you want to be precise with animations and specify how you want your UI elements to move.
    - Rotation: Adjusts the rotation of the canvas. Useful for allowing you to rotate all of your UI elements together.
    - Scale: Determines the size of the canvas. Useful because this affects all of your UI elements so if you wanted to make a size change to everything without editing them one by one.
- Canvas
    - Renderer Mode: Determines how the canvas will be rendered within the scene (Screen Space Overlay, Screen Space Camera, or World Space). Screen space overlay is used for traditional 2D while the other two are for using UI elements in a 3D environment. This setting is useful because it gives you the flexibility to use UI elements in different kinds of projects (2D vs 3D vs hybrid).
    - Pixel Perfect: Makes the elements in the canvas align with pixels if selected. This is useful for making your UI elements appear sharper, but may not work well if you animate the UI elements.
    - Sort Order: Specifies the order that UI elements are rendered. Useful for controlling how your UI is layered with the rest of the game.
    - Target Display: Specifies the display monitor or screen where the canvas is rendered. Useful if you have multiple monitors/screens or VR applications.
    - Additional Shader Channels: Allows you to select all of the shader features you want. This is useful if you want to enhance the visual fidelity of your UI elements.
- Canvas Scaler
    - UI Scale Mode: Determines the scaling mode used by the Canvas Scaler (scale with screen size, constant pixel size, or constant physical size) . Useful because it

allows you to select which scaling works best for your project, so you may want to change it to scale with screen size if you want your project to be used easily on many screen sizes.
    - Scale Factor: Sets the overall scale factor for the UI, adjusting its size relative to the screen or reference resolution.
    - Reference Pixels Per Unit: Makes a reference point for how UI elements scale, this is useful because it helps you make sure your UI is consistent on different devices/resolutions.
- Graphic Raycaster
    - Script: Allows you to specify a script for your graphic raycaster. This is useful because you can customize this script however you want to make the raycaster work exactly as you want it to for your project.
    - Ignore Reversed Graphics: If this is enabled it will make the raycaster ignore UI elements which face away from the camera. This is useful for optimizing your project as it will eliminate unnecessary operations.
    - Blocking Objects: This is a list of game objects which block raycasts from messing with UI elements. This would be useful if you want to make the UI interactions not happen in some situations, like if the UI were behind a wall.
    - Blocking Mask: Allows you to control which layers block raycasts from interacting with UI elements. Useful because it allows you more control over which exact objects impede UI interaction so you can prevent unintended interactions.

Quick Reference

Introduction

Canvas is important to understand because you will need to deal with it anytime you want to use UI game objects. UI game objects are children of the canvas, meaning the changes you make with the canvas will affect all of these objects.

Canvas Render Modes:

- An important setting to consider on the Canvas component is the render mode as this determines whether Rect Transform can be edited, the 3 options are:
    - **Screen Space Overlay**: Default mode. Rect Transform values can't be edited. Used for UI that overlays the scene.
    - **Screen Space Camera**: Renders canvas as part of scene viewed by a camera. Allows editing of Rect Transform values. Useful for UI that moves with the camera.
    - **World Space**: Renders canvas as a 3D object in scene. Allows full editing of Rect Transform values. Ideal for integrating UI into 3D environments.
- Each project is unique and has different needs for the UI so this setting is important because it will help customize your UI for your use case. For simple 2D projects like what we have been doing in class, the default setting will work fine.

Canvas Scaler:

- Adjusts the scale of UI elements to fit different screen resolutions and aspect ratios.
- UI Scale Mode: Determines how UI elements scale relative to the camera or reference resolution.

Graphic Raycaster:

- Determines how UI elements interact with raycasts from input devices.
- Ignore Reversed Graphics: Ignores UI elements facing away from the camera.
- Blocking Objects/Mask: Specifies objects or layers that block raycasts from interacting with UI elements.

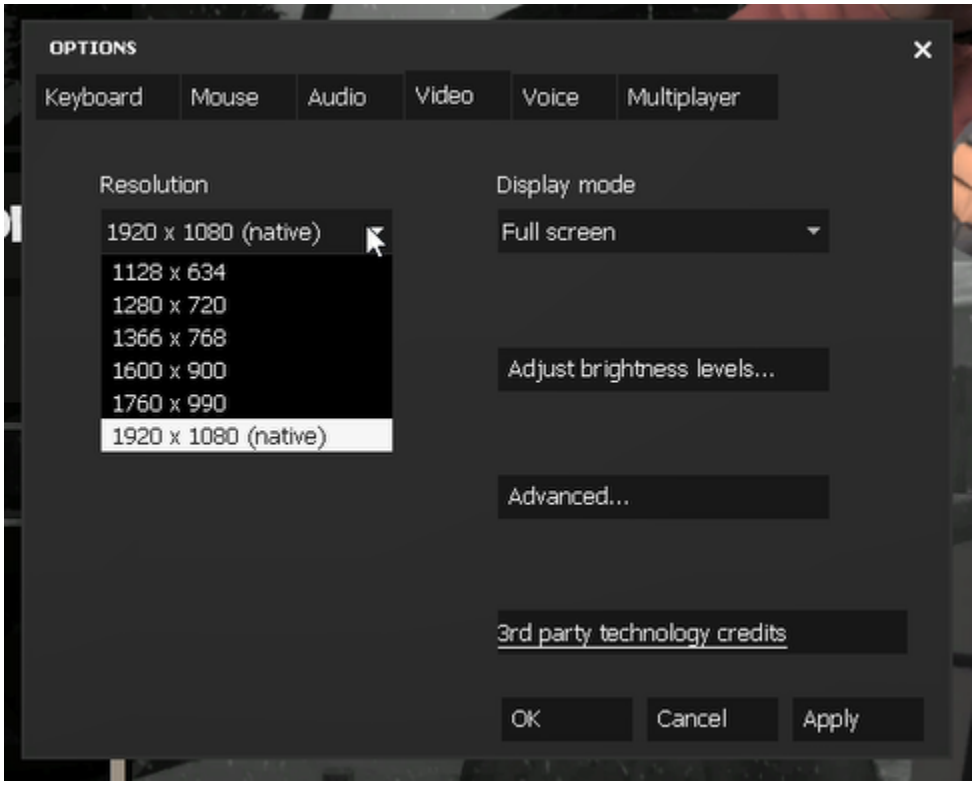
Conclusion:

Most projects will automatically have a canvas. So, using canvas effectively and understanding all of the options and settings will allow your project to have a more immersive and better UI.

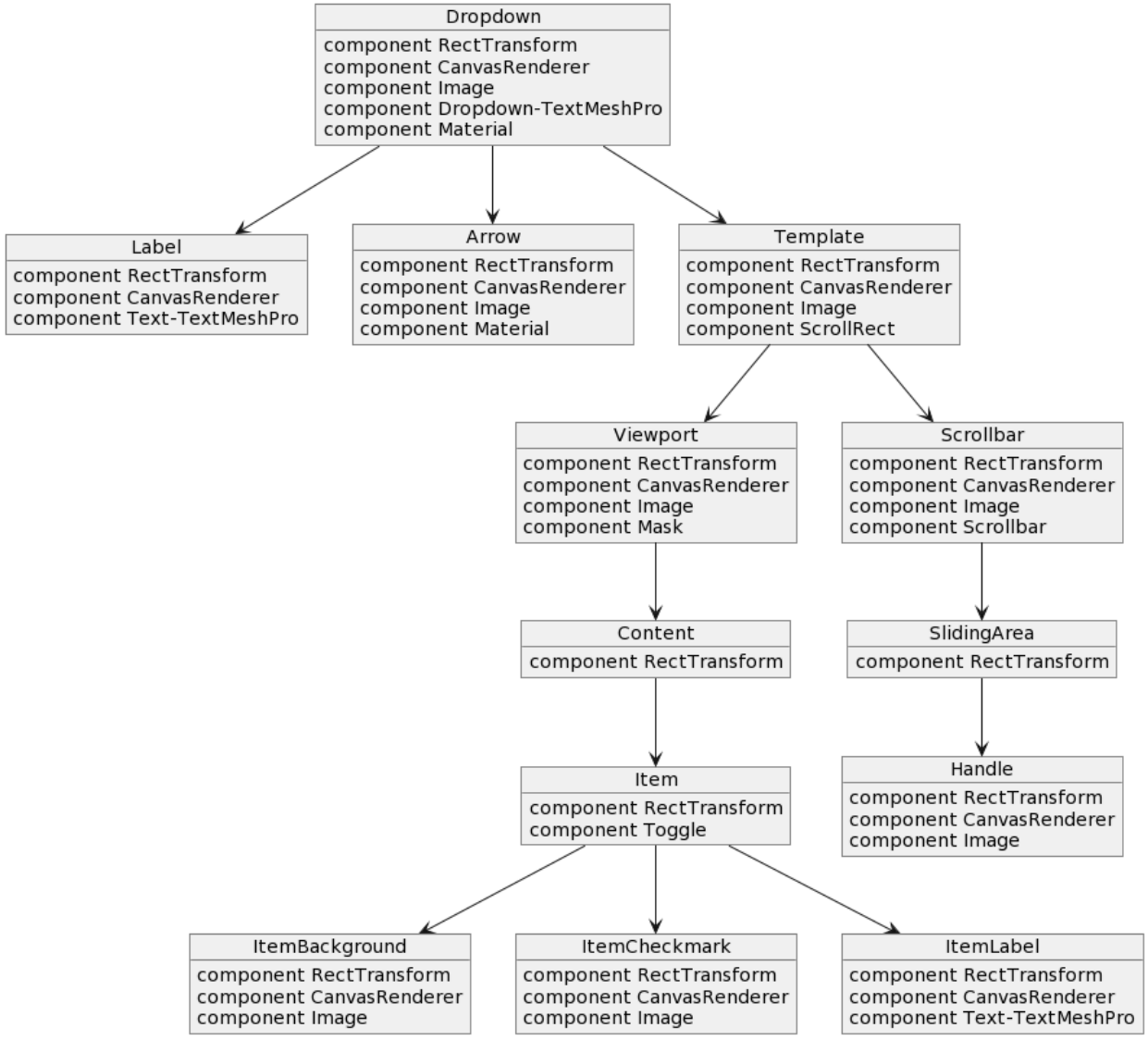
Dropdown Guide

Overview

Allows you to create an interactive dropdown menu which users can select options from. In Team Fortress 2 and many other games, this menu type is used in settings menus to allow the user to select from a set of preselected options.



GameObjects



Settings

- Dropdown - TextMeshPro
    - Interactable: Whether the dropdown can be interacted with. This is useful because there may be times when you want to make it so the dropdown cannot be used, for example if a setting can't be changed at a certain point in your game.
    - Transition: Determines the transition between colors. This is useful for providing visual feedback to your users.
    - Colors: Which color the main menu will change on different events. This is useful for customizing your UI's appearance, and for improving the user's experience through a dynamic UI. The following events are available:
        - Normal

        - Highlighted
        - Pressed
        - Selected
        - Disabled
    - Fade Duration: Determines the rate of fade between colors
    - Navigation: Determines whether the dropdown is oriented horizontally or vertically
    - Options: The list of options selectable from the dropdown. Each option can have a text label and/or an associated sprite
    - On Value Changed: The list of functions that will be triggered when a new option is selected from the dropdown.
- Label/Item Label - Text - TextMeshPro
    - Text Input: The text input is automatically changed at runtime, so this option has no effect
    - Text Style: The style of the displayed text. Uses HTML/markdown styling notations
    - Font Asset: The font used in the label
    - Font Style: Whether the text is bold, underlined, etc
    - Font Size: The size of the font. Can be auto-sized to fit different resolutions
    - Vertex Color: The color of the font. Can be given a color gradient, and it can be set to override any previously applied color tags
    - Spacing Options: The amount of margin spacing
    - Alignment: The text alignment within the label
    - Wrapping: Whether the text will wrap to the next line
    - Overflow: Determines how text overflow is handled
- Template - ScrollRect
    - Content: A reference to the content of the dropdown
    - Horizontal / Vertical: Whether the scroll bar(s) should scroll horizontally or vertically
    - Movement Type: How the scroll bar should move
    - Inertia / Deceleration Speed: Control how the scroll bar should move and how it should maintain velocity
    - Scroll Sensitivity: The sensitivity of the scrolling
    - Viewport: A reference to the viewport of the dropdown
    - Horizontal Scrollbar: A reference to the horizontal scrollbar (if any)
    - Vertical Scrollbar: A reference to the vertical scrollbar (if any)
    - Visibility: Determines when and how the scrollbar should become visible
    - Spacing: The spacing of the scrollbar
    - On Value Changed: A list of functions that should be called when the scroll bar changes position or the menu is otherwise scrolled.

- Scrollbar - Scrollbar
    - See Dropdown - TextMeshPro for information about the colors and navigation options
    - Handle Rect: A reference to the dropdown's handle
    - Direction: Which direction the scroll bar should scroll
    - Value: The starting scroll bar position
    - Size: The size of the scroll bar
    - Number of Steps: How many steps the scroll bar should have. A value of 0 will result in smooth, stepless motion.
    - On Value Changed: A list of functions that should be triggered when the scroll bar's position is changed
- Item - Toggle
    - See Dropdown - TextMeshPro for information about the colors and navigation options
    - Is On: Whether this item is visible and interactable
    - Toggle Transition: Which transition should take place when the item's state changes
    - Graphic: A reference to the image indicating whether an item is selected (Item Checkmark)
    - Group: The group this item belongs to
    - On Value Changed: A list of functions that should be triggered when this item is selected

Quick Reference

Introduction:

Dropdown is an important UI component in Unity, commonly used for presenting a list of selectable options to the user. This reference will highlight some of the most important settings that you will want to consider when using a dropdown

Options:

- Specifies the list of options displayed in the dropdown.

On Value Changed:

- Event triggered when the selected option changes. Use this to respond to user selection and trigger certain actions when things are selected.

Template:

- Template used for rendering each dropdown item.

Caption Text:

- Text component displaying the currently selected option.

Dropdown List:

- Panel containing the selectable options.

Dropdown Image:

- Image component for the dropdown button.

Navigation:

- Specifies navigation behavior between dropdown items using arrow keys or gamepad.

How to customize:

- Customize the appearance of the dropdown through adjusting colors, fonts, and sizes of various components.
- Use custom templates and scripts to implement complex dropdown functionality, such as dynamic option generation or filtering.

Conclusion:

Dropdown is a versatile UI component that provides a convenient way to present selectable options to users. Understanding and effectively utilizing dropdown settings and functionality will enhance the interactivity and usability of your project's UI.