

# Bài 6: Hệ thống UI trong Unity

*Giảng viên:*

# MỤC TIÊU

- Cấu trúc Project
- A. Khởi tạo và cấu hình dự án Game 2D
- B. Tạo các đối tượng cơ bản
  - 1. *Game Object*
  - 2. *Sprite*
  - 3. *Animation và điều khiển hành động nhân vật*
  - 4. *Prefab*
  - 5. *Script và điều khiển máy trạng thái*
  - 6. *Thành phần vật lý và xử lý va chạm*
  - **7. Hệ thống UI**
  - 8. *Sử dụng Particle System*
  - 9. *Chuyển đổi màn chơi*
  - 10. *Sound*
  - 11. *Design Pattern trong Game*

# Nội dung

- Cấu trúc Project
- A. Khởi tạo và cấu hình dự án Game 2D
- B. Tạo các đối tượng cơ bản
  - 1. *Game Object*
  - 2. *Sprite*
  - 3. *Animation và điều khiển hành động nhân vật*
  - 4. *Prefab*
  - 5. *Script và điều khiển máy trạng thái*
  - 6. *Thành phần vật lý và xử lý va chạm*
  - **7. Hệ thống UI**
  - 8. *Sử dụng Particle System*
  - 9. *Chuyển đổi màn chơi*
  - 10. *Sound*
  - 11. *Design Pattern trong Game*

# Hệ thống UI trong Unity

- Giao diện người dùng (User Interface – UI) là thành phần không thể thiếu đối với bất kỳ game nào.
- UI cung cấp các thông tin trực quan cần thiết cho người chơi, giúp người chơi có cái nhìn toàn diện về các khả năng của mình (thời gian, điểm, "máu", ...) và có chiến thuật thích hợp để vượt qua được các thử thách trong game.
- Việc thiết kế giao diện người dùng đơn giản là sử dụng các assets có sẵn (hình ảnh, font chữ, các hiệu ứng, ...) và sắp xếp chúng theo một bố cục được Designer thiết kế.
- Các assets này có thể được tìm thấy trên các website, Assets Store của Unity, hoặc do chính các Designer, Artist trong dự án thiết kế (thường gặp ở những dự án lớn hoặc vừa).
- Các thành phần cơ bản trong thiết kế UI bao gồm Canvas, Text, Image, Button, ....

# Hệ thống UI trong Unity- Canvas

- **Canvas** có thể hiểu là một vùng riêng cho phép chứa đựng thành phần giao diện người dùng (UI). Là một game object có một component là Canvas, và tất cả các UI phải là con của một Canvas.
- Các thành phần UI khác khi được khởi tạo bắt buộc phải nằm trong một canvas.
- Khi khởi tạo một thành phần UI (Text, Image, ...), Unity sẽ tự động tạo ra một canvas nếu chưa tồn tại một canvas nào trong Scene.
- Để khởi tạo một canvas, trong cửa sổ Hierarchy, ta chọn **Create** → **UI** → **Canvas**. Các đối tượng UI khác cũng được khởi tạo tương tự.
- Các đối tượng con của một Canvas sẽ được render theo thứ tự từ trên xuống dưới trong cửa sổ Hierarchy. Đối tượng nào ở trên sẽ được render trước và có thể bị che khuất bởi đối tượng ở dưới.

# Hệ thống UI trong Unity- Canvas

## Các chức năng quan trọng của Canvas

- **Render mode:** Có 3 tùy chọn hiển thị canvas:
  - **Screen Space – Overlay.** Canvas sẽ được vẽ lên layer cao nhất của màn hình và nằm trên mọi game object khác. Canvas với render mode này hoàn toàn không phụ thuộc vào camera.
  - **Screen Space – Camera.** Đối với mode này, ta cần chỉ định một camera cho canvas, nó sẽ được render theo camera. Nếu như không có camera được chỉ định thì canvas và các thành phần bên trong sẽ không được render.
  - **World Space.** Với tùy chọn này, đối tượng canvas sẽ được xem như một game object thông thường. Tùy chọn này sử dụng event camera thay vì render camera. Ngoài các chức năng như render camera, event camera còn có thêm chức năng bắt sự kiện, dựa trên thứ tự render, toạ độ z, ... của các đối tượng UI.

# Hệ thống UI trong Unity- Canvas

## Các chức năng quan trọng của Canvas

- **Render mode:**
- Đối với các tùy chọn render theo Screen Space, Unity cung cấp tính năng Pixel Perfect, tăng khả năng hiển thị sắc nét và khử vết mờ.

# Hệ thống UI trong Unity- Canvas

## Các chức năng quan trọng của Canvas

### ■ Rect Transform

- Tương tự như thành phần Transform trong các game object khác. Rect Transform được sử dụng để xác định kích thước, vị trí và luân chuyển trong hệ thống điều khiển giao diện người dùng
- Đối với các tùy chọn render mode Screen Space – Overlay và Screen Space – Camera, thành phần Rect Transform sẽ được khoá lại và không thể tùy chỉnh. Canvas sẽ điều chỉnh các thông số một cách tự động để phù hợp với độ phân giải màn hình game.



# Hệ thống UI trong Unity - Canvas

## Các chức năng quan trọng của Canvas


### ■ Graphic Raycast

- Hỗ trợ bắt sự kiện.
- Khi nhận được tín hiệu (mouse click, touch, ...), một tia nhìn tại vị trí tương tác sẽ được tạo ra. Bằng cách này, chúng ta dễ dàng xác định được đối tượng mà người chơi muốn tương tác, thông qua tọa độ z của đối tượng.



**DEMO**

Canvas

A large white hand cursor icon, resembling a computer mouse pointer, is positioned below the word 'DEMO'. The index finger of the hand is pointing upwards towards the letter 'O' in 'DEMO'.

# Hệ thống UI trong Unity- Rect Transform

## Rect Transform:

- **Rect Transform** là một thành phần mới mà được dùng để thay thế cho thành phần Transform trên tất cả các UI mới mà Unity bổ sung ở phiên bản 4.6.
- Thành phần Transform của một game object trong Unity có 3 yếu tố là **position** – căn chỉnh kích thước, **rotation** – điều chỉnh độ xoay và **scale** dùng căn chỉnh tỷ lệ của một đối tượng trong Scene.

# Hệ thống UI trong Unity- Rect Transform

- **Position** cho thấy khoảng cách bằng pixel từ các điểm anchor đến pivot, dọc theo trục X và Y, Z là khoảng cách dọc trục Z local (chiều sâu) và thường được để ở giá trị 0. Width và Height cho biết kích thước của UI theo pixel.
- **Rotation**: thì xoay object, thường thì trong 2D mình chỉ sử dụng xoay trục Z.
- **Scale**, bạn phải hiểu được sự khác nhau giữa scale và size. Size thay đổi kích thước của nó, không ảnh hưởng đến UI object bên trong, còn scale sẽ canh chỉnh lại tỷ lệ của object và tất cả các thành phần con. Mình thấy resize một yếu tố UI thì tốt hơn là scale nó. Scale thường được sử dụng cho các hiệu ứng động hay các mục đích đặc biệt khác

# Hệ thống UI trong Unity- Rect Transform

## Rect Transform:

- **Rect Transform** thể hiện một hình chữ nhật được xác định bởi chiều rộng và chiều cao liên quan đến một điểm tâm của nó (gọi là **pivot**).
- **Pivot** là điểm trụ của object, thông thường mặc định sẽ là ở tâm của object, nếu bạn xoay UI thì nó sẽ xoay quanh điểm tâm.
- Anchor(điểm neo): một Rect Transform có thể được anchored tới đối tượng cha của nó, nếu cha của nó cũng có một thành phần Rect Transform.
- **Anchor** cho phép chúng ta di chuyển hoặc kéo dài UI dựa trên position và size của thành phần Rect Transform của UI cha. Cần phải chú ý rằng tất cả các UI sau cùng cũng phải trở thành là một con của một Canvas.
- Theo một cách nào đó thì UI con sẽ được neo vào UI cha dựa vào thành phần Rect Transform, tất cả UI đều có Rect Transform.

# Hệ thống UI trong Unity- Rect Transform

- **Anchor:** Trong scene, neo được nhận biết bởi bốn hình tam giác nhỏ, mà theo mặc định, được nhóm lại với nhau ở trung tâm của thành phần rect transform UI cha.
- Cụ thể về Anchor:
  - Rect transform có thể được neo vào parent object nếu parent object cũng có một thành phần Rect Transform.
  - Trong scene view: neo được đại diện bởi bốn hình tam giác nhỏ, mỗi tam giác đại diện cho một neo. Và mỗi neo kết hợp với một góc của rect transform tương ứng. (chụp hình)
  - Neo có thể được di chuyển bằng cách kéo thả trên Gizmo tam giác, Nhấp ở trung tâm của một nhóm các hình tam giác sẽ di chuyển các hình tam giác như là một nhóm. Nhấp vào hình tam cá nhân sẽ di chuyển một neo đầu tại một thời điểm.
- Neo có một mối quan hệ cố định với rect transform của nó, nó đang được neo. Neo có mối quan hệ linh hoạt với Rect Transform của object cha, mà họ đang neo đầu.

# Hệ thống UI trong Unity- Rect Transform

## ■ Vị trí Anchor

- Khi bạn thiết lập một neo, khoảng cách giữa neo và góc liên kết của rect transform là một giá trị cố định.
- Vị trí của neo là ở bên trong Rect Transform của đối tượng cha, tuy nhiên nó có thể cân đối tùy theo kích thước object cha và bị ảnh hưởng bởi thay đổi kích thước trong các Rect Transform của đối tượng cha.
- Vị trí của neo cân đối với đối tượng cha là một số phần trăm kích thước của đối tượng cha dọc theo trục X và Y, (chụp hình Anchor).

# Hệ thống UI trong Unity- Rect Transform

- Để thực hiện các thiết lập nhanh chóng và dễ dàng Rect Transform bao gồm cửa sổ neo Presets.
- Điều này có thể được truy cập bằng cách nhấp vào Nút neo Presets.

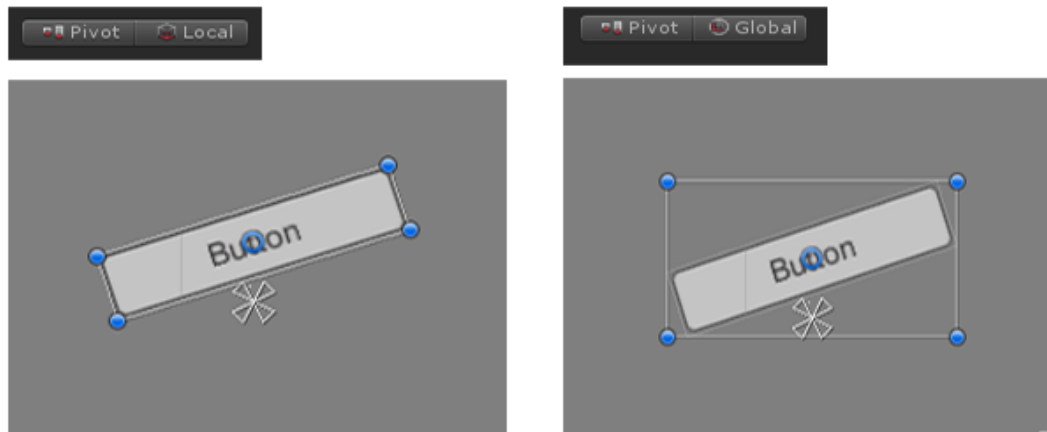


# Hệ thống UI trong Unity- Rect Transform

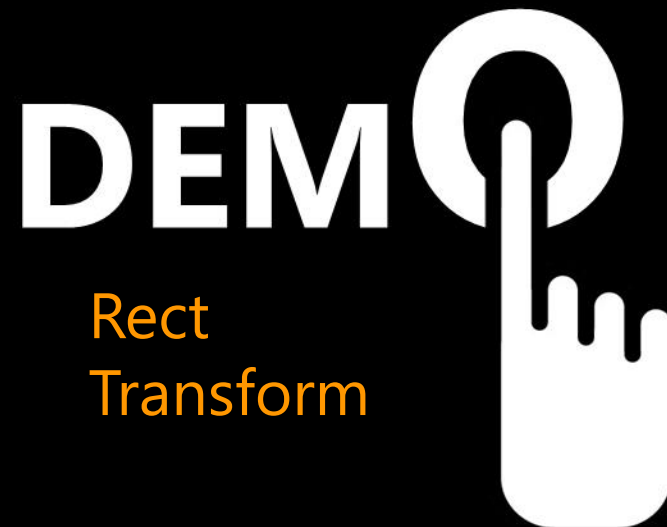
- Trong Scene, để thao tác với một UI, cách tốt nhất là sử dụng Rect Tool, với phím tắt là Shift + T.
- Với Rect tool được chọn có thể để di chuyển, thay đổi kích thước và xoay bất kỳ UI nào bằng cách giữ chuột và kéo, Giữ phím Shift sẽ buộc các UI thay đổi kích thước tương ứng.
- Kéo chuột ở gần bất kỳ góc nào của hình chữ nhật để xoay.

# Hệ thống UI trong Unity- Rect Transform

- Global and Local space trên thanh công cụ).



- Khi chỉnh sửa một UI trên local space, hình chữ nhật và xử lý phải sắp thẳng hàng, (phủ vừa đủ) với UI.
- Còn trong global space, hình chữ nhật và xử lý phải được sắp thẳng hàng với tổng thể (hay tổng thể thôi) trên toàn cầu, và hình chữ nhật cho thấy giới hạn của UI. Điều này phát hiện rõ nhất khi chúng ta xoay một UI.



# Hệ thống UI trong Unity- Text

- UI Text được sử dụng để hiển thị các thông tin trên màn hình như Score, Lives, Times, ...
- Một số game sử dụng các texture riêng để hiển thị thông tin thay cho text, tuy nhiên text vẫn là lựa chọn phổ biến hơn vì tính đơn giản và dễ thao tác.
- Để khởi tạo một text, trong cửa sổ Hierarchy, ta chọn Create → UI → Text.

# Hệ thống UI trong Unity- Text

- Sau khi khởi tạo một đối tượng Text, cửa sổ Inspector có dạng như sau:
  - Rect Transform: quản lý vị trí, kích thước, góc quay, ... của Text.
  - Text: lưu trữ chuỗi ký tự cần hiển thị ra màn hình.
  - Unity hỗ trợ một số chức năng để tùy biến chuỗi như font, kiểu chữ (thường, in đậm, in nghiêng, ...), cỡ chữ, màu chữ, ...
  - Tùy chọn **Best Fit** sẽ tự động điều chỉnh kích thước font chữ phù hợp với kích thước được quy định trong Rect Transform.

# Hệ thống UI trong Unity- Image

- UI Image được sử dụng rất phổ biến, như thiết kế background, các button, title, ...
- Cách sử dụng rất đơn giản, chúng ta chọn hình ảnh và kéo thả vào khung Source Image. Ngoài ra còn có một số tùy chọn thay đổi màu sắc, chất liệu, ...
- Để khởi tạo một text, trong cửa sổ Hierarchy, ta chọn Menu-GameObject → UI → Image.

# Hệ thống UI trong Unity- Image

## *Các thuộc tính của một UI Image:*

- **Source Image:** tham chiếu đến sprite mà hình ảnh hiển thị
- **Color:** Màu sắc, màu này được nhân với màu sắc của sprite, như vậy nó có thể được sử dụng để làm mờ dần hay tô màu sprite
- **Material:** Vật liệu, mặc định thì hình ảnh không cần một vật liệu, thuộc tính này được sử dụng khi bạn muốn thêm một thuộc tính shader (đổ bóng) cho hình ảnh. Unity cung cấp một số shaders có thể được sử dụng với UI System.

# Hệ thống UI trong Unity- Image

## ***Các thuộc tính của một UI Image:***

- **Image Type:** Hình ảnh được sử dụng như thế nào.
  - ***Simple:*** có nghĩa là sprite sẽ chỉ kéo dài để phù hợp với kích thước của Rect Transform.
  - ***The Preserve Aspect:*** ở đây chỉ đơn giản có nghĩa là hình ảnh sẽ được lớn nhất có thể trong giới hạn của Rect Transform và luôn giữ đúng một tỉ lệ ban đầu.
  - ***Set Native Size:*** Phục hồi kích thước Rect Transform như kích thước gốc của sprite từ ảnh nguồn.
  - ***Sliced:*** có nghĩa là hình ảnh sẽ được hiển thị bằng cách sử dụng phương pháp 9 lát cắt.



# Hệ thống UI trong Unity- Image

## ■ **Các thuộc tính của một UI Image:**

- **Tiled:** Hiển thị đúng phần hình ảnh theo kích thước Rect Transform, địa chỉ offset được tính từ góc dưới bên trái của Rect.
- **Fill:** Loại này cho phép bạn hiển thị một phần hình ảnh được xác định bởi thuộc tính Fill Amount. Và thêm một số tùy chọn:
  - *Fill Method:* Xác định hướng sẽ được lấp đầy khi Fill Amount tăng lên
  - *Fill Origin:* Xác định điểm đầu tiên đại diện khi Fill Amount là 0
  - *Clockwise:* theo chiều kim đồng hồ.



# Hệ thống UI trong Unity- Button

- UI Button là một thành phần quan trọng, giúp người chơi tương tác với game.
- Nhận input từ người dùng và bắt một sự kiện (Hover, Press, Release)
- Giống như UI khác – một button phải là một thành phần con của canvas.
- Một số button quen thuộc có thể thấy trong nhiều game là Play, Pause, Resume, Replay, ...

# Hệ thống UI trong Unity- Button

## ***Các thành phần chính trong một Object UI Button***

- ***Rect Transform***: một thành phần mà object UI nào cũng có, bạn có thể xem lại ở phần trước.
- ***Image script***: Ở phần trước mình đã nói rõ về UI Image, các bạn có thể xem lại ở phần trước.
- ***Text***: Khi chúng ta tạo một Button, một thành phần con là Text được tự động tạo ra, chúng ta có thể sử dụng hay xóa nếu thấy không cần thiết.
- ***Button script***: thành phần chính làm nên một UI Button

# Hệ thống UI trong Unity- Button

## ***BUTTON SCRIPT:***

- ***Interactable:*** Một biến bool, xác định sự chuyển tiếp (transition) của button, nếu Interactable là False, button sẽ ở trạng thái Disabled, mặc định thì button ở trạng thái Normal với Interactable được chọn.
- ***Transition mode:*** Có 4 tùy chọn về sự chuyển tiếp của một button, và mỗi chuyển tiếp ta có một sự luân chuyển khác nhau giữa các trạng thái của button.
  - ***None:*** Không có sự chuyển tiếp, lúc nào button cũng ở trạng thái normal, ta sẽ khó phân biệt khi có sự tác động vào button (khi người dùng hover, press,..)

# Hệ thống UI trong Unity- Button

## ***BUTTON SCRIPT:***

- **ColorTint:** Giá trị mặc định khi button được tạo ra
- 1.Target graphic: là thành phần hình ảnh được nhuộm màu (tạm hiểu được tô màu khi có tác động từ người dùng), bạn có thể thấy rằng khi ta tạo một button, image này sẽ được tự động tạo theo.
- 2.Normal: Button ở trạng thái bình thường,
- 3.Highlighted: Khi kéo rê chuột trên button,
- 4.Pressed (but not released): Khi click chuột trên button,

# Hệ thống UI trong Unity- Button

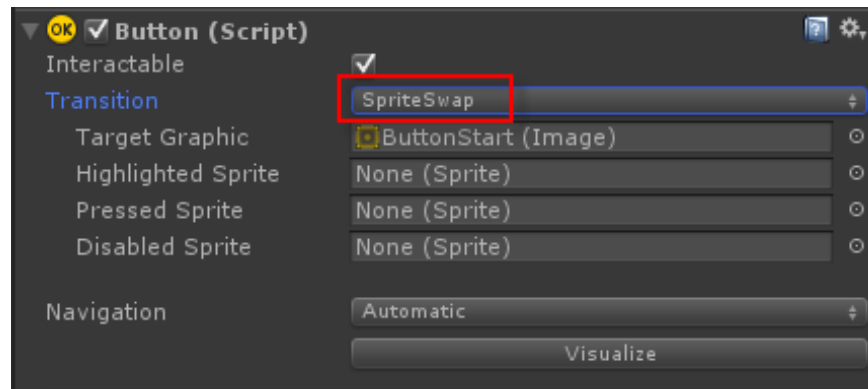
## ***BUTTON SCRIPT:***

- **ColorTint:**
  - 5.Disabled: Không thao tác được button này.  
➔ Normal Color, Highlighted Color, Pressed Color, Disable Color sẽ định nghĩa cách nhuộm màu như thế nào tới button với trạng thái tương ứng.
  - 6.Color Multiplier: Điều chỉnh độ sáng màu sắc hay alpha màu với giá trị từ 1 – 5, sẽ thay đổi với từng trạng thái tương ứng.
  - 7.Fade Duration: là khoảng thời gian tính bằng giây, chuyển đổi giữa các trạng thái.

# Hệ thống UI trong Unity- Button

## ***BUTTON SCRIPT:***

- **SpriteSwap:** Sử dụng sprite khác cho mỗi trạng thái.
  - *Target graphic*: giống như trên colorTint
  - Highlighted Sprite, Pressed Sprite, Disable Sprite tham chiếu tới những sprite sẽ hiển thị tương ứng với mỗi trạng thái.





# Hệ thống UI trong Unity- Button

## ***BUTTON SCRIPT:***

- **Animation:** Cho phép mỗi sự chuyển tiếp sử dụng animation.
- Khi sử dụng sự chuyển tiếp này, button cần có một thành phần animator, để control những animation khi chuyển tiếp giữa các trạng thái. Animator controller có thể được thêm vào thủ công nhưng Unity đã hỗ trợ ta tự động generate ra thành phần này, ta có thể điều chỉnh theo ý mình nhưng mình thấy điều này là không cần thiết.

## ■ ***Navigation:***

# Hệ thống UI trong Unity- Button

## ***XỬ LÝ SỰ KIỆN KHI CLICK BUTTON***

- Bạn sẽ thấy có một thành phần On Click() bên trong Button Script, nó bao gồm một danh sách các hàm có thể được gọi khi click button.

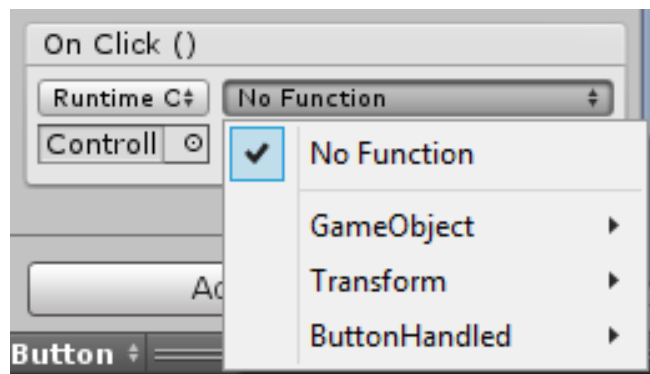
# Hệ thống UI trong Unity- Button

## ***XỬ LÝ SỰ KIỆN KHI CLICK BUTTON***

- Để thêm một hàm vào list, click “+” icon, checkbox để xác định là có sử dụng hàm này khi click trên button hay không.
  - *Trường đầu tiên:* là một object, ta có thể kéo các object vào đây hay có thể sử dụng picker bên cạnh để chọn các object.
  - *Trường thứ 2:* là function list, khi ta chọn object cho trường đầu tiên, một danh sách các thành phần của object này sẽ được liệt kê ra, và với mỗi thành phần sẽ có một menu đính kèm các hàm mà script của thành phần đó có, chọn hàm mà ta muốn.
  - *Trường cuối cùng:* là tham số của hàm trên, tham số phải là float, int, string, bool or là Unity object.
- *Một điều phải lưu ý hàm thích hợp phải là Publish, return kiểu Void và có 1 hoặc không có tham số.*

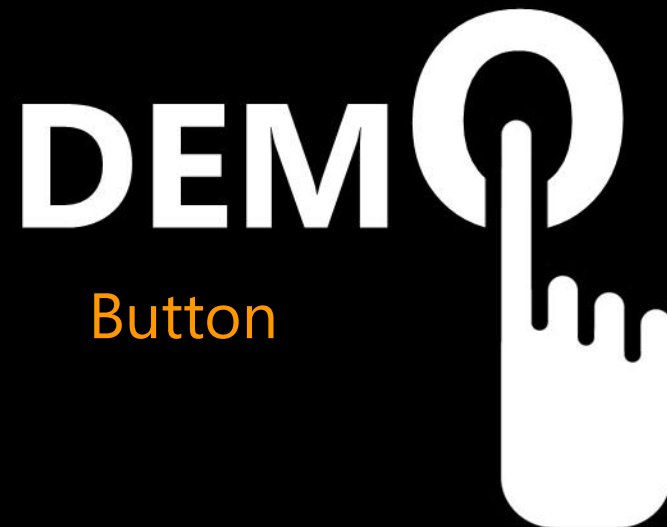
# Hệ thống UI trong Unity- Button

- Khi click vào button, tất cả các hành động được thiết lập sẵn trong event On Click sẽ được thực hiện, do đó ta có thể thực hiện đồng thời nhiều hành động tùy theo nhu cầu.
- Các đối tượng được thêm vào có thể là một game object trong cửa sổ Hierarchy hoặc chính button đó.
- Khi đó event sẽ tự động bắt được các thành phần của game object và hiển thị trong menu như trong hình dưới.



# Hệ thống UI trong Unity- Button

- *Lưu ý: nếu cần gọi một hàm trong script, script đó phải được gắn vào một game object trong cửa sổ Hierarchy và hàm này cần có phạm vi truy cập là public.*
- Ngoài ra, còn hàng loạt event khác cho button được cung cấp trong thành phần Event Trigger (Add Component -> Event -> Event Trigger). Cách sử dụng tương tự như với event On Click.



# Hệ thống UI trong Unity- Scroll Rect

- Scroll Rect là một hình chữ nhật mà có thể được cuộn theo chiều ngang hay theo chiều dọc.
- Scroll Rect thường được sử dụng để di chuyển cuộn một hình ảnh lớn hoặc panel của một UI element, chẳng hạn như một danh sách các button hoặc rich text (một đoạn văn bản lớn) mà vùng chứa nó không hiển thị đủ ta phải cuộn (scroll) thì mới thấy hết được.
- Scroll Rect được thiết kế để làm việc với một thanh cuộn scrollbar. Nó thường được sử dụng với một UI Mask, và có thể kèm theo đó là một UI Image được sử dụng để kiểm soát hình dạng của thành phần UI Mask.

# Hệ thống UI trong Unity- Scroll Rect

## *Các thuộc tính của một thành phần Scroll Rect:*

- **Content:** Đây là một tham chiếu đến thành phần Rect Transform của object UI được cuộn, ví dụ như một hình ảnh lớn, hay khi ta sử dụng danh sách các button, thì Content ở đây là thành phần Rect Transform của object cha chứa các UI button.
- **Movement Type:**
  - **Unrestricted:** Không giới hạn, khi bạn chọn kiểu chuyển động này thì nội dung (các item) có thể sẽ vượt ra ngoài phạm vi của Scroll Rect, lúc này bạn không thấy item nữa.
  - **Elastic:** Loại chuyển động đàn hồi, co giãn các nội dung sẽ không thể vượt ra ngoài phạm vi của Scroll Rect. Khi chọn Elastic, nội dung (các item) sẽ có tính đàn hồi trở lại khi bạn scroll nó đạt đến cách Scroll Rect.
    - **Elasticity:** Độ co giãn.
  - **Clamped:** Bạn sẽ không thể scroll từ trái sang phải nếu item đầu tiên đã hiển thị đủ, hay ngược lại, bạn không thể scroll từ phải qua trái nếu item cuối đã hiển thị đủ.



# Hệ thống UI trong Unity- Scroll Rect

## *Các thuộc tính của một thành phần Scroll Rect:*

- **Horizontal:** Cho phép cuộn ngang.
- **Vertical:** Cho phép cuộn dọc.
- **Inertia:** Quán tính, nếu không chọn có quán tính thì nội dung sẽ không được cuộn nữa mà dừng lại ngay lập tức, thiết lập chế độ có quán tính khi ta muốn nội dung tiếp tục di chuyển thêm 1 đoạn theo quán tính khi ta dừng kéo chuột.
  - Deceleration Rate: Tỷ lệ giảm tốc độ khi ta chọn chế độ scroll có quán tính.
- **Horizontal Scrollbar:** Tham chiếu đến một thành phần thanh cuộn ngang.
- **Vertical Scrollbar:** Tham chiếu đến một thành phần thanh cuộn dọc.

# Hệ thống UI trong Unity- ScrollBar

- **Khái niệm Scroll Bar:**
- Thanh điều khiển Scrollbar cho phép người dùng di chuyển một hình ảnh hay một khung nhìn nào đó mà kích thước nội dung của nó khá lớn vượt ra ngoài tầm khung nhìn, không thể xem hoàn.
- Ví dụ quen thuộc ta thường thấy như thanh cuộn dọc trong trình soạn thảo văn bản hay thanh cuộn ngang lúc xem một bản đồ...

# Hệ thống UI trong Unity- ScrollBar

## Các thuộc tính của một UI Scroll bar:

- **Interactable:** Cho phép sử dụng thành phần này hay không, nếu bạn không tick chọn thì thành phần này sẽ bị disabled.
- **Transition:** Xác định các phản ứng trực quan của control (thường là thay đổi màu sắc, sprite) tới hành động của người dùng. Các hành động của người dùng như nhấn chuột, hover chuột...tương ứng với các hành động thì sẽ có các phản ứng khác nhau.

Ở đây có 4 chế độ chuyển tiếp

- *None:* Không có phản ứng trực quan nào lại hành động của người dùng.
- *ColorTint:* Chế độ mặc định, cho phép đổi màu handle tương ứng với từng tác động từ người dùng
- *SpriteSwap:* Cho phép thay đổi sprite tương ứng với từng tác động của người dùng.
- *Animation:* Cho phép sử dụng animation thay đổi linh hoạt tương ứng với từng tác động của người dùng.

# Hệ thống UI trong Unity- ScrollBar

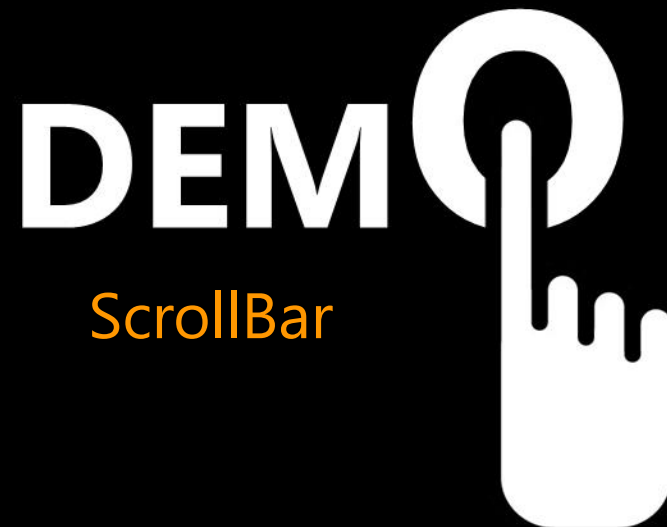
## Các thuộc tính của một UI Scroll bar:

- **Navigation:** Thuộc tính xác định thứ tự của các control.
- **Handle Rect:** Đồ họa được sử dụng cho thanh xử lý trượt, một phần của chính UI Scrollbar
- **Direction:** Hướng mà trong đó giá trị của Scrollbar sẽ tăng lên khi handle được kéo. Các tùy chọn là Left To Right, Right To Left, Bottom To Top và Top To Bottom.
- **Value:** Giá trị ban đầu của Scrollbar, trong khoảng 0.0 đến 1.0
- **Size:** Kích thước phân đoạn của handle trong Scrollbar, trong khoảng 0.0 đến 1.0
- **Number Of Steps:** Số lượng các vị trí di chuyển riêng biệt được cho phép của Scrollbar.

# Hệ thống UI trong Unity- ScrollBar

## Sự kiện có thể bắt trên một UI Scrollbar:

- ***On Value Changed***: Được gọi bất cứ khi nào giá trị vị trí của Scrollbar thay đổi, một kết quả khi kéo handle. Giá trị được truyền cho hàm trả về là một kiểu float.



# Kết luận

- Cấu trúc Project
- A. Khởi tạo và cấu hình dự án Game 2D
- B. Tạo các đối tượng cơ bản
  - 1. *Game Object*
  - 2. *Sprite*
  - 3. *Animation và điều khiển hành động nhân vật*
  - 4. *Prefab*
  - 5. *Script và điều khiển máy trạng thái*
  - 6. *Thành phần vật lý và xử lý va chạm*
  - **7. Hệ thống UI**
  - 8. *Sử dụng Particle System*
  - 9. *Chuyển đổi màn chơi*
  - 10. *Sound*
  - 11. *Design Pattern trong Game*

# Chuẩn bị bài sau

- Cấu trúc Project
- A. Khởi tạo và cấu hình dự án Game 2D
- B. Tạo các đối tượng cơ bản
  - 1. *Game Object*
  - 2. *Sprite*
  - 3. *Animation và điều khiển hành động nhân vật*
  - 4. *Prefab*
  - 5. *Script và điều khiển máy trạng thái*
  - 6. *Thành phần vật lý và xử lý va chạm*
  - 7. *Hệ thống UI*
  - **8. *Sử dụng Particle System***
  - **9. *Chuyển đổi màn chơi***
  - **10. *Sound***
  - **11. *Design Pattern trong Game***





**FPT POLYTECHNIC**

THANK YOU!

[www.poly.edu.vn](http://www.poly.edu.vn)