**Lại Thị Phương Nhung**

**0850070036**

**08\_ĐH\_TTMT**

**Lý Thuyết Buổi 7**

**1.Animation**

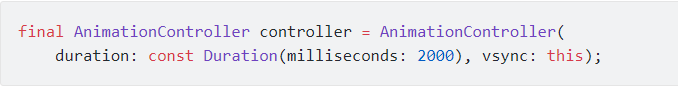
Animation là một trong những ưu điểm của Flutter so với các công nghệ lập trình đa nền tảng khác. Với Flutter lập trình viên có thể dễ dàng tạo ra các hiệu ứng chuyển động mượt mà và bắt mắt thông qua đó nâng cao trải nghiệm người dùng.

**Animation:** là một lớp trừu tượng, thể hiện giá trị và trạng thái của một animation (đã bắt đầu, hoàn thành, dừng, …), Khi muốn thêm animation cho các widget, widget thường được truyền một đối tượng Animation. Từ đối tượng đó widget có thể biết giá trị hiện tại của animation.

**Listener:** Bất cứ khi nào giá trị của một đối tượng Animation thay đổi, đối tượng Animation đó sẽ thông báo cho tất cả các listener được thêm bằng hàm addListener. Thường thì State object ứng với widget muốn thêm sẽ lắng nghe animation và gọi setState để vẽ lại animation đó.

**StatusListener:** được thêm bằng hàm addStatusListener, khi trạng thái của đối tượng Animation thay đổi (bắt đầu, kết thúc, đang chuyển động, đang chuyển động theo chiều ngược lại) các đối tượng Animation cũng sẽ thông báo.

**AnimationController:** AnimationController là một đối tượng Animation đặc biệt tạo ra một giá trị mới bất cứ khi nào phần cứng sẵn sàng cho một khung mới. Mặc định, AnimationContoder tạo tuyến tính các số từ 0,0 đến 1,0 trong một khoảng thời gian nhất định. Ví dụ sẽ tạo ra một AnimationController



[AnimationController](https://docs.flutter.io/flutter/animation/AnimationController-class.html) hỗ trợ phương thức để điều khiển animation.

* Điều khiển animation [forward](https://docs.flutter.io/flutter/animation/AnimationController/forward.html) hoặc [reverse](https://docs.flutter.io/flutter/animation/AnimationController/reverse.html), hoặc [stop](https://docs.flutter.io/flutter/animation/AnimationController/stop.html)
* Đặt animation thành một [value](https://docs.flutter.io/flutter/animation/AnimationController/value.html) cụ thể.
* Định nghĩa các giá trị [upperBound](https://docs.flutter.io/flutter/animation/AnimationController/upperBound.html) and [lowerBound](https://docs.flutter.io/flutter/animation/AnimationController/lowerBound.html) trong một animation.
* Tạo ra một [fling](https://docs.flutter.io/flutter/animation/AnimationController/fling.html) animation với hiệu ứng sử dụng một mô phỏng vật lý.

Khi tạo AnimationController, bạn truyền cho nó một đối số vsync, ngăn animations tiêu tốn tài nguyên không cần thiết bên ngoài màn hình. Bạn có thể sử dụng đối tượng trạng thái của mình làm vsync bằng cách thêm SingleTickerProviderStateMixin vào định nghĩa lớp. Ví dụ:



**AnimatedBuilder:** là một widget rất hữu ích khi xây dựng animation. Nó hiệu quả hơn việc gọi setState(), mỗi khi có sự thay đổi giá trị của animation.

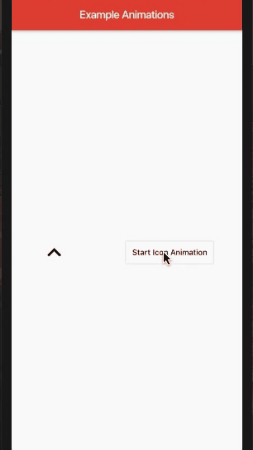
Ví dụ:



Đoạn code trên có thể thấy 2 thuộc tính của AnimatedBuilder:

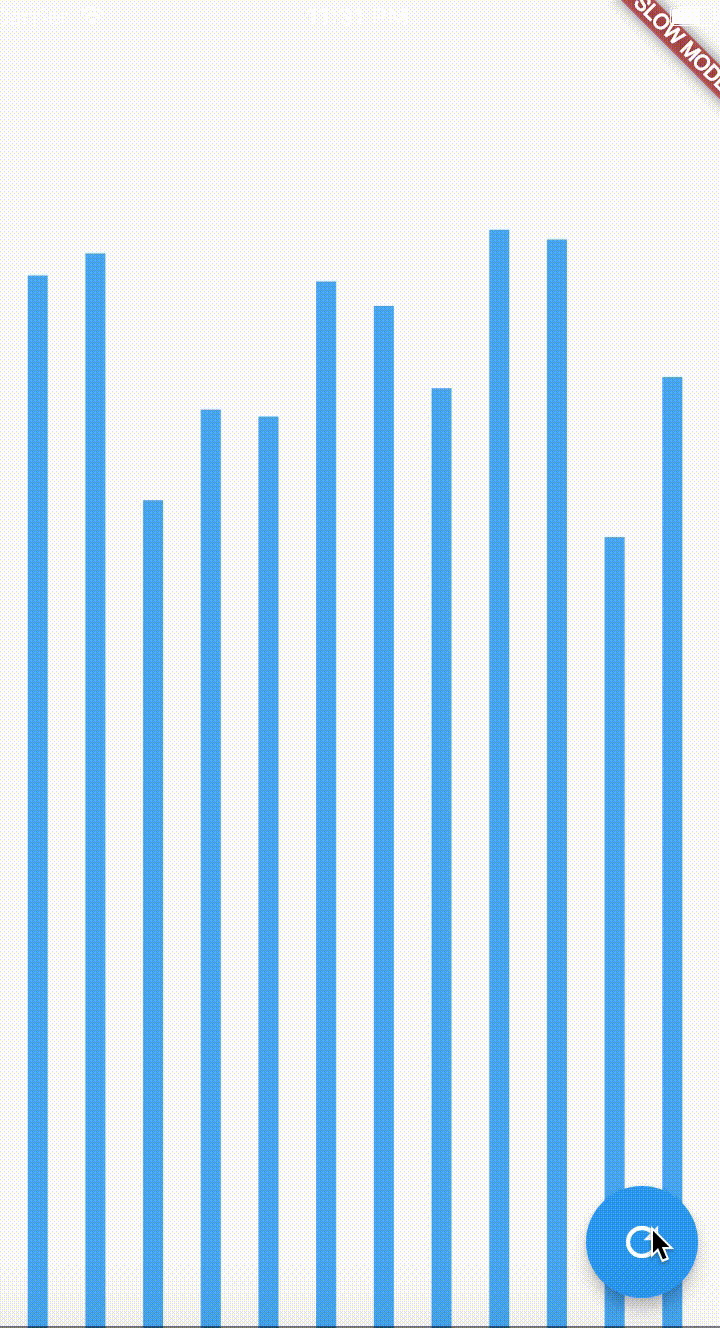
* animation: nó mong đợi một animationController chịu trách nhiệm kiểm soát animation. Trong trường hợp trên là \_arrowAnimationController sẽ điều khiển animation xoay icon mũi tên.
* builder: đây là một callback() được gọi mỗi khi giá trị của animation thay đổi. Trong hàm builder, trả về một widget icon được bọc trong widget Transform.rotate().

Widget còn lại là OutlineButton, animation sẽ bắt đầu khi nút được nhấn

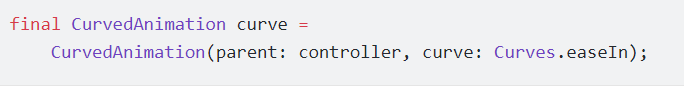


AnimatedWidget:

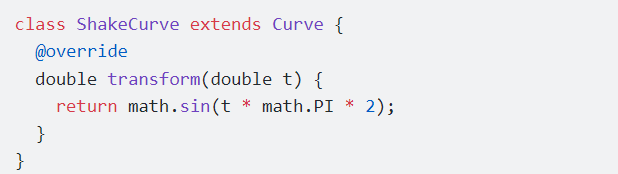
**CurvedAnimation:** một CurvedAnimation miêu tả tiếng trình động giống như một đường cong phi tuyến tính. Ví dụ:



Khởi tạo hay định nghĩa một curvedAnimation như sau:



Có thể tự tạo một Curve riêng dựa vào lớp Curves

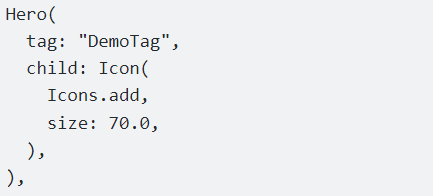


CurvedAnimation và AnimationController đều là kiểu Animation<double>, vì thế chúng ta có thể sử dụng thay thế cho nhau. CurvedAnimation sẽ chứa đối tượng nó đang thay đổi, không cần một lớp con AnimationController để thực hiện một curve.

**Hero Animation:**

Hero Animation là một widget giúp chúng ta tạo các hiệu ứng khi chuyển màn. Hero Animation sử dụng một biểu tượng hiện được gọi là “Hero” và khi quá trình chuyển đổi trang được kích hoạt, thường bằng cách nhấp vào biểu tượng, trên màn hình sẽ xuất hiện hoạt ảnh “bay” đến trang tiếp theo. Khi người dùng điều hướng trở lại trang trước đó, hoạt ảnh sẽ đi theo hướng khác và biểu tượng quay trở lại vị trí ban đầu của nó. Hero Animations là một trong những hiệu ứng dễ làm nhất trong Flutter và không yêu cầu thiết lập nhiều. Nhìn vào ví dụ dưới đây, chúng ta có thể thấy rằng cùng một biểu tượng tồn tại trên cả hai trang. Tất cả những gì chúng ta cần là nói cho Flutter rằng cả hai được liên kết với nhau. Chúng tôi thực hiện việc này bằng cách bao bọc một widget như một biểu tượng trong Hero widget.

Ví dụ:

****