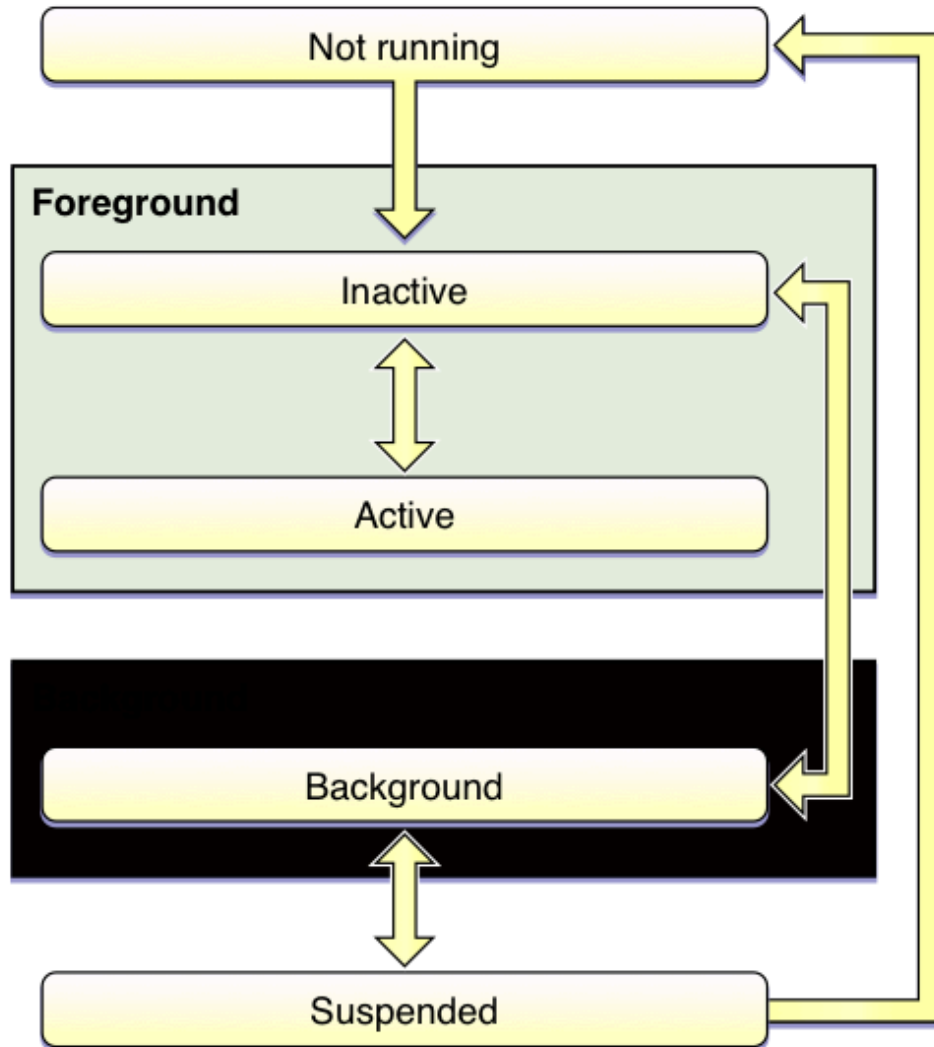


Lập trình giao diện cho iOS

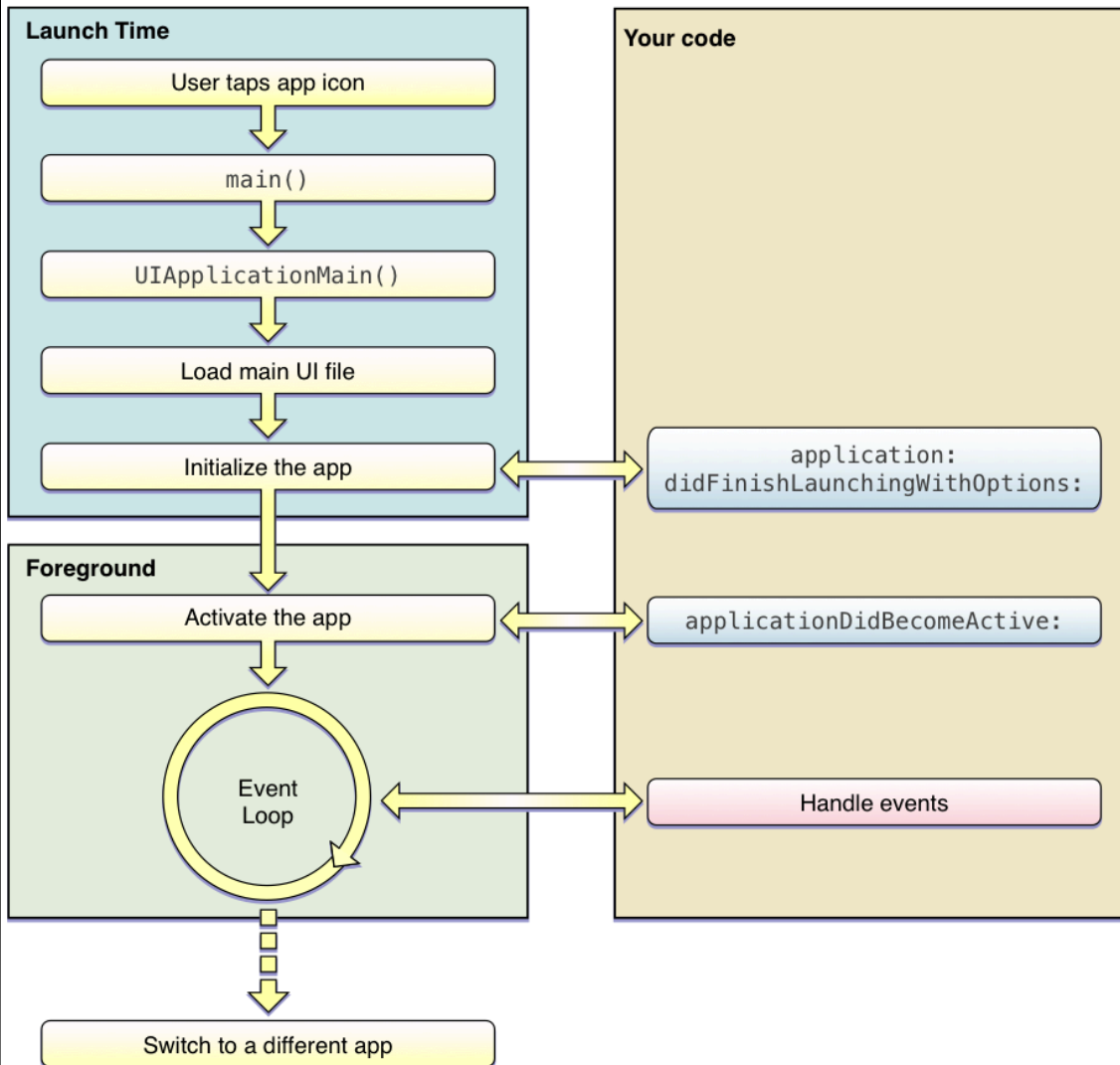




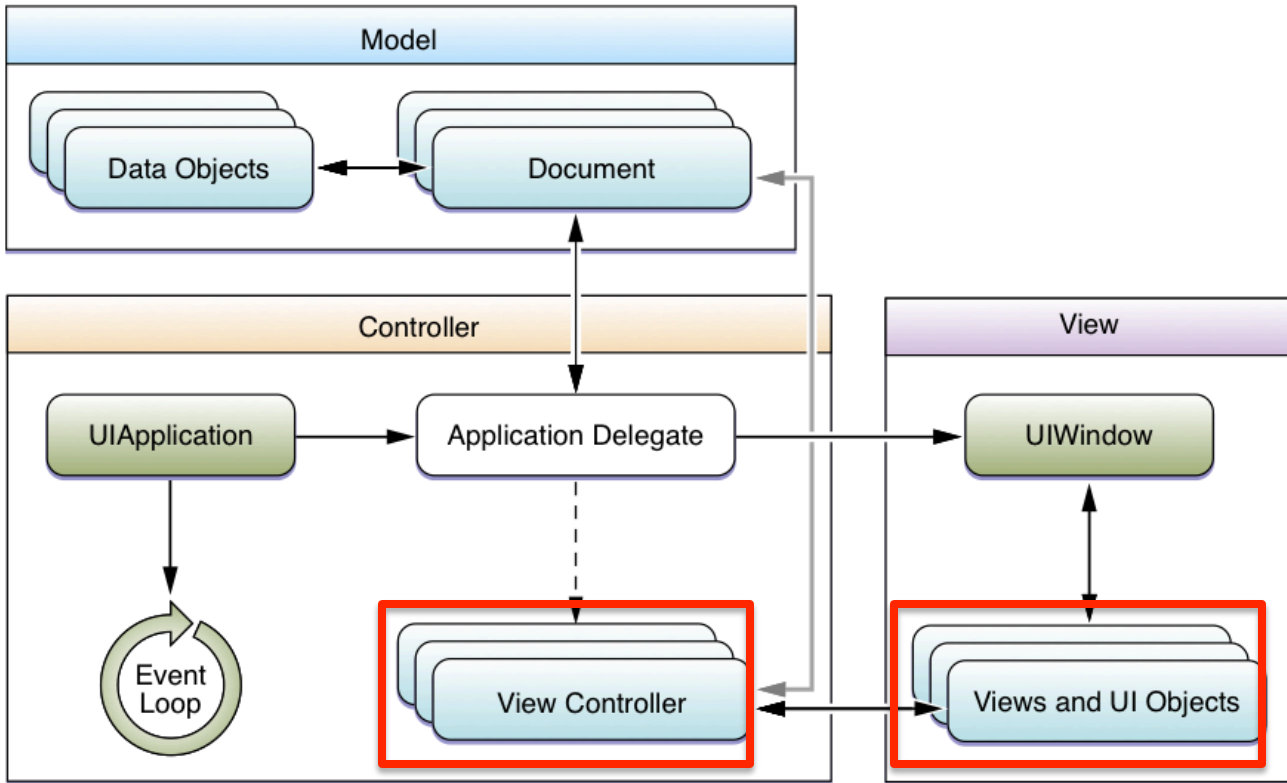
Các trạng thái chính của ứng dụng.
Thường quan tâm đến Active và Background




Một số trình chơi nhạc vẫn chạy Background
và phát nhạc.

Bài này sẽ không đi sâu về vấn đề này



Chuỗi tuần tự các sự kiện xảy ra khi người dùng bật một ứng dụng.



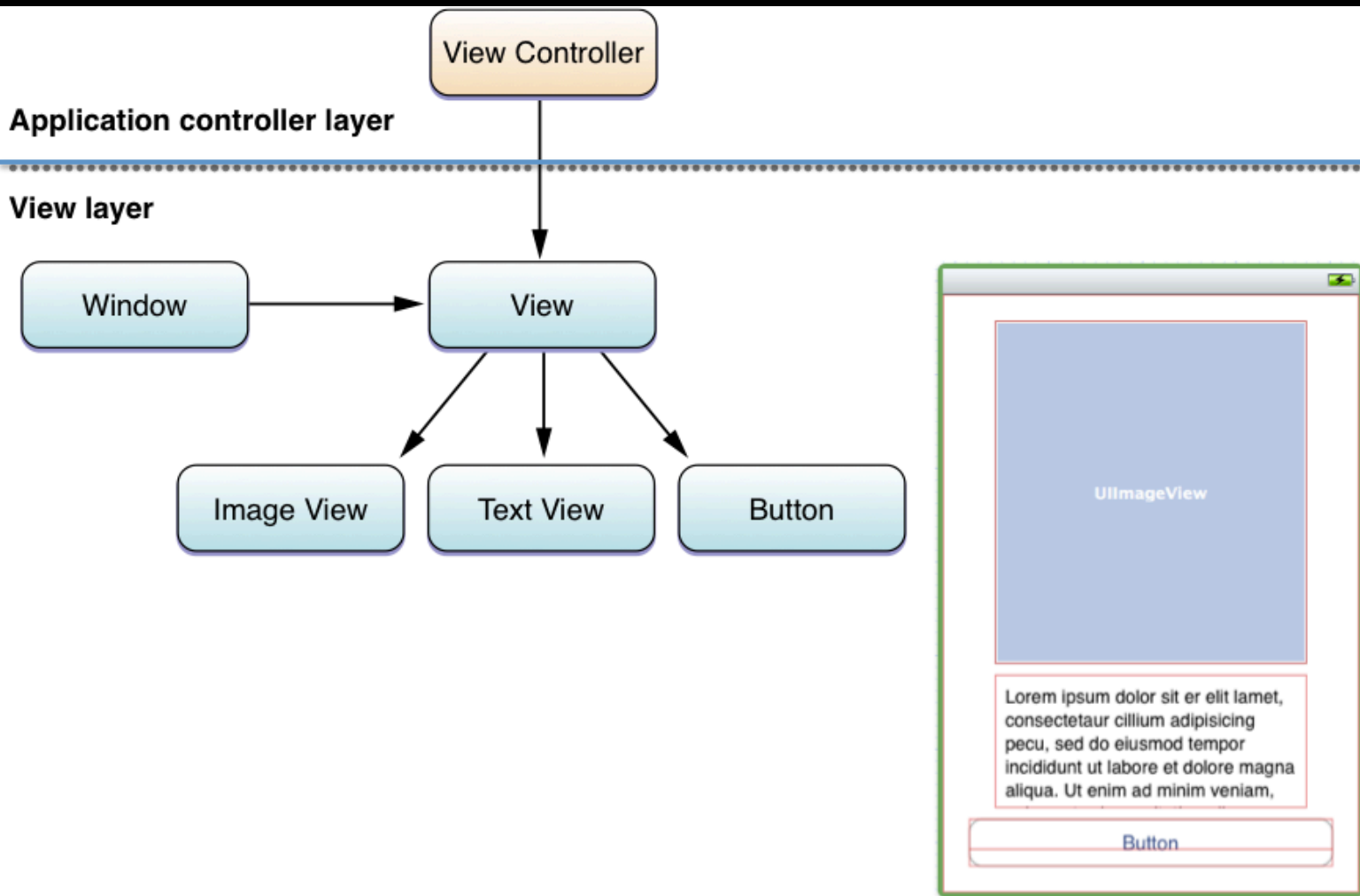
-  Custom Objects
-  System Objects
-  Either system or custom objects

Mô hình Model – View – Controller.

Model cung cấp mẫu, kiểu, nguồn dữ liệu

View trình bày dữ liệu, nhận tương tác người dùng, và chịu điều khiển từ Controller

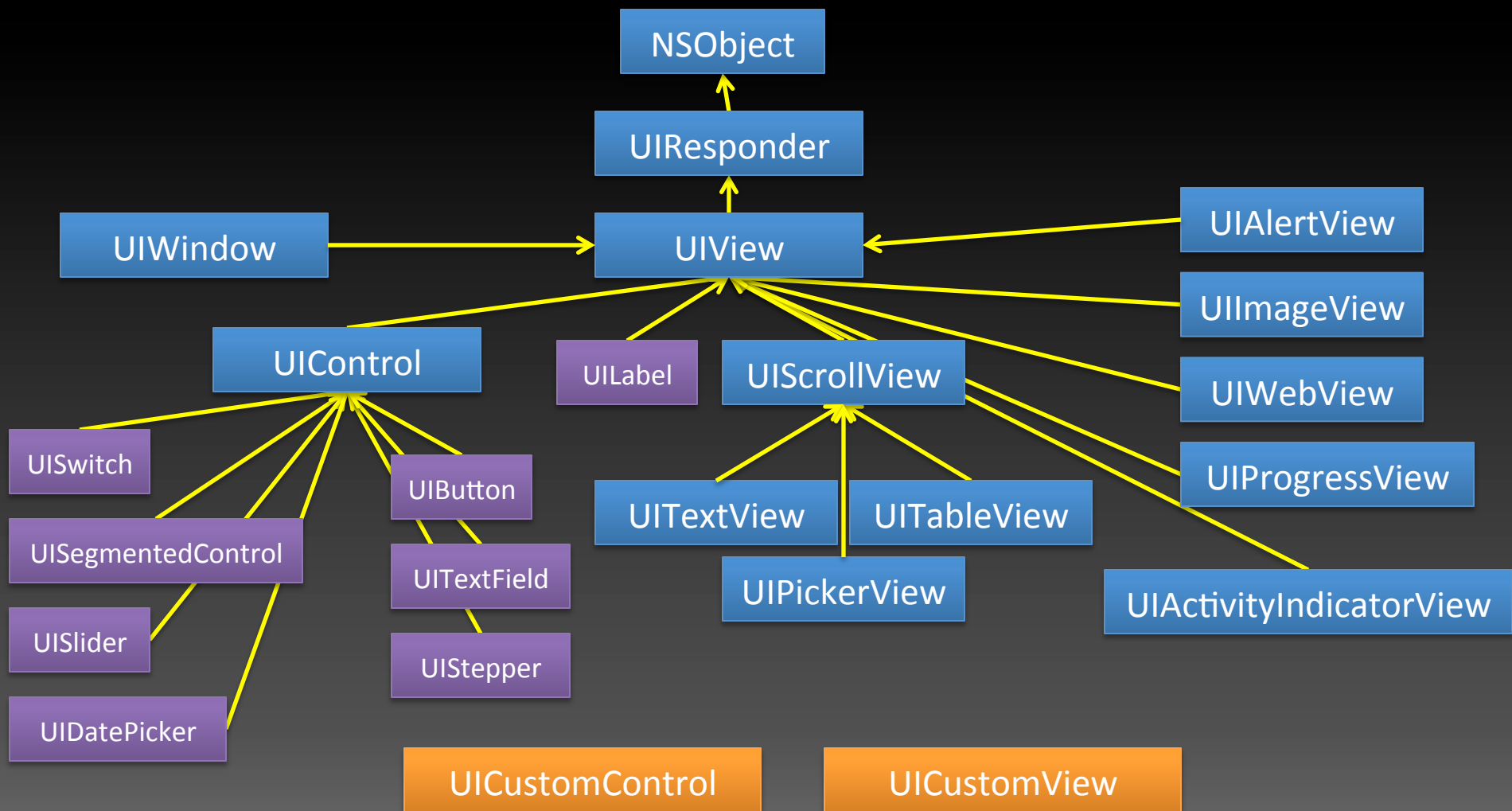
Controller là nơi lập trình viên viết mã lấy dữ liệu từ model, trình bày ra view

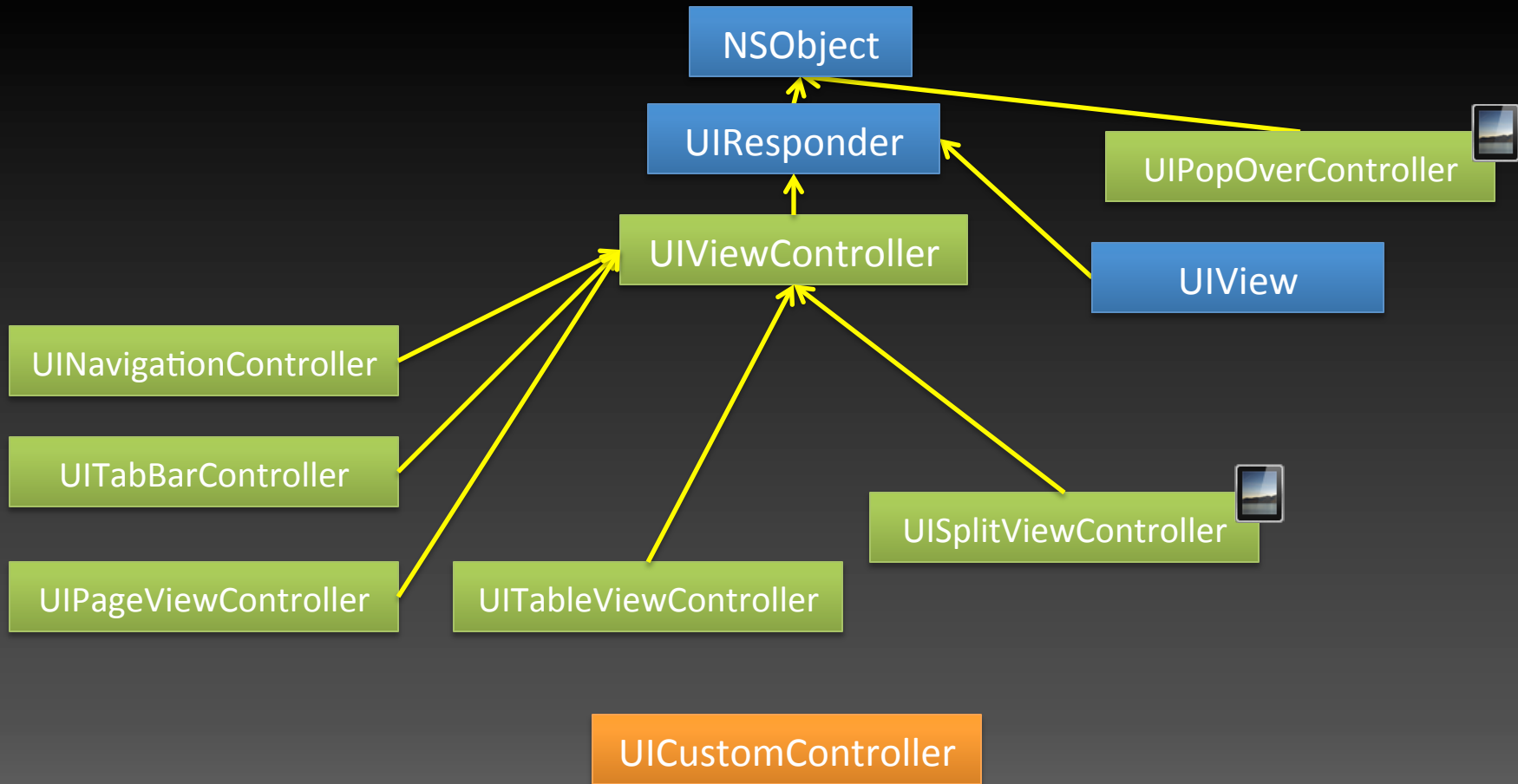


Lập trình logic
điều khiển

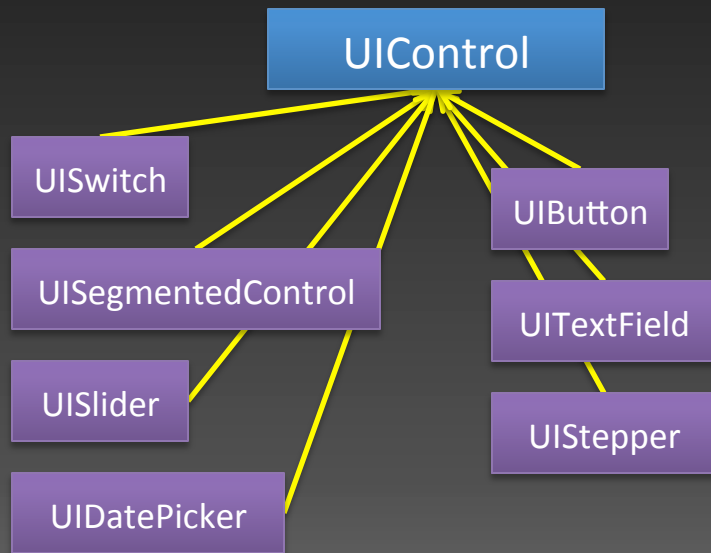
Thiết kế giao
diện

1. Tự viết mã
2. Kéo thả trên
InterfaceBuilder
3. Dùng StoryBoard



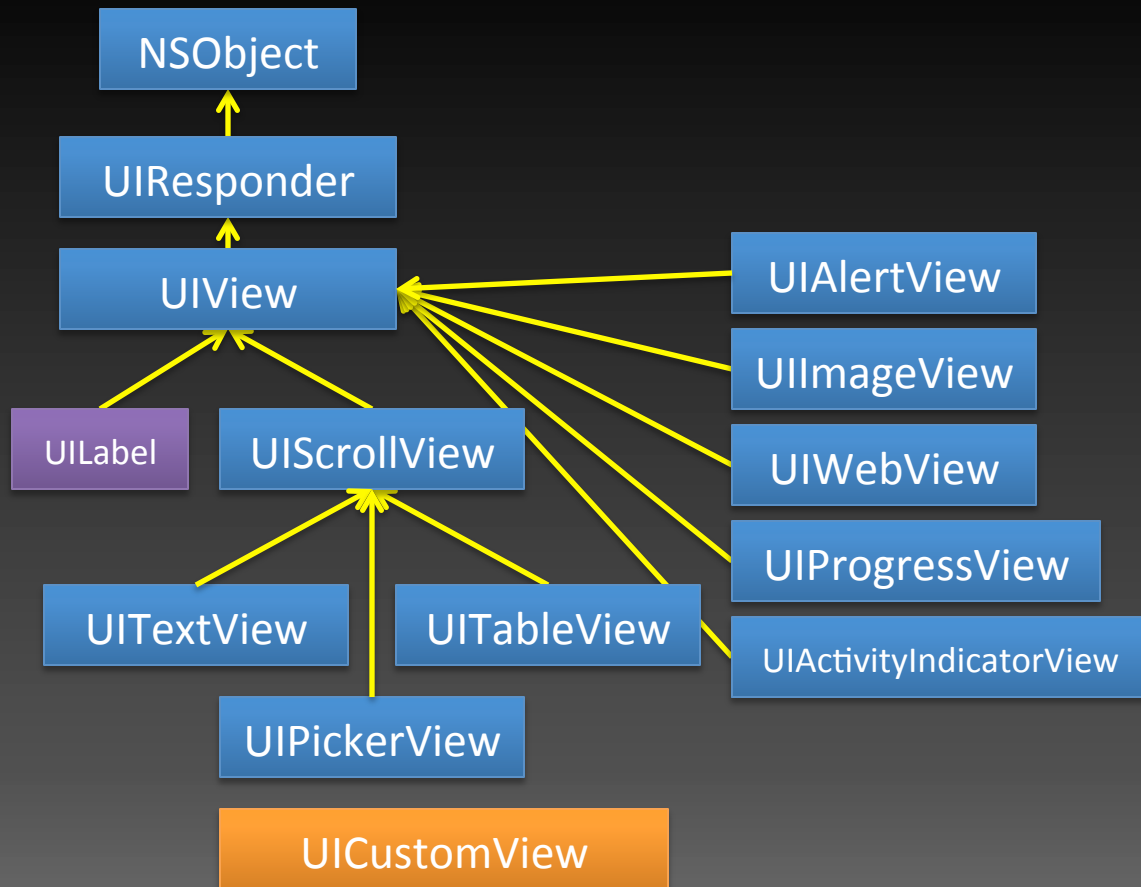


UIControl Derivatives

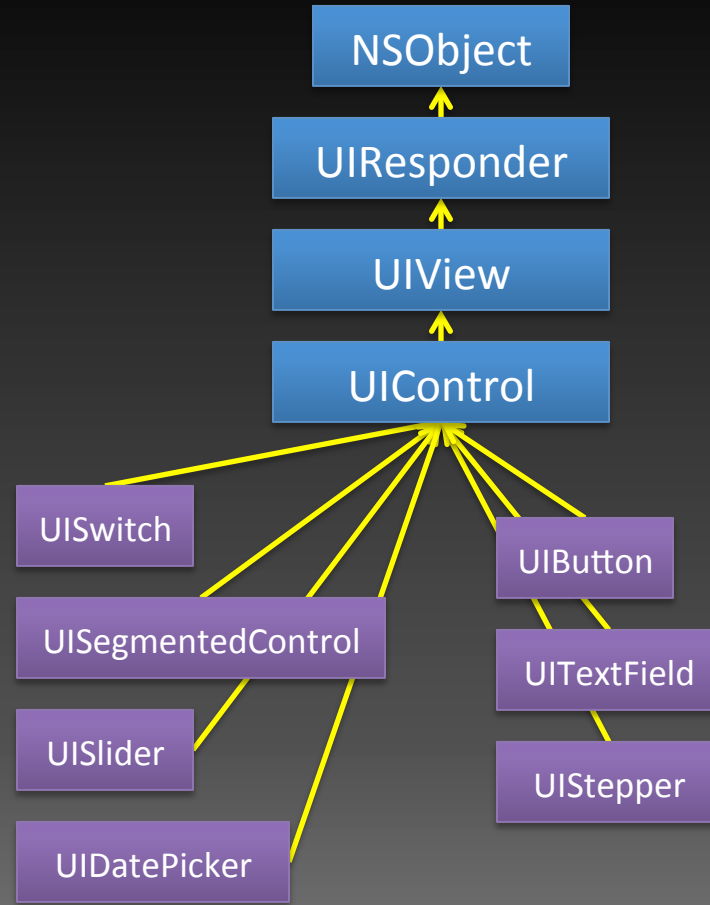


Views displays data

- UILabel
- UITextView
- UIImageView
- UIWebView
- UIScrollView
- UITableView
- MKMapView



Controls




Target - Action

Là cách để một đối tượng giao diện (thừa kế từ UIView) hứng sự kiện tương tác của người dùng, chuyển qua cho đối tượng Target, thực hiện bởi hàm chọn lựa qua Action.

Đối tượng Target thường là lớp thừa kế từ UIViewController

Target - Action

```
myButton addTarget:self  
action:@selector(buttonClicked:)  
forControlEvents:UIControlEventTouchUpInside  
];
```



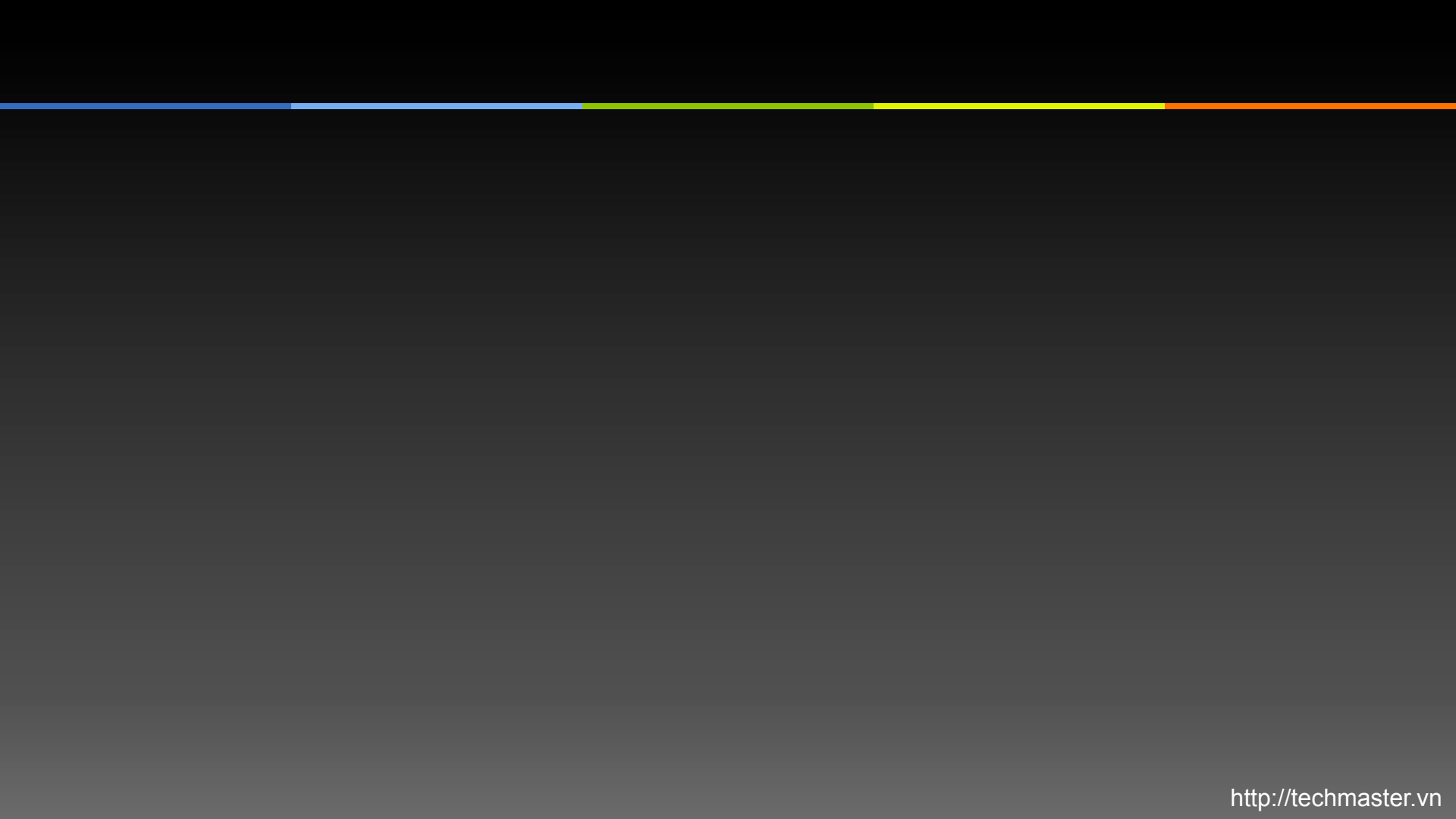
```
- (void)buttonClicked:(id)sender {  
    myLabel.text = @"Clicked";  
}
```

Target – Action khác gì so với Delegate

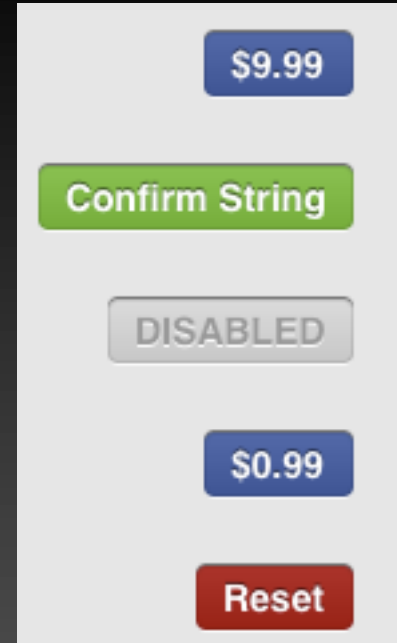
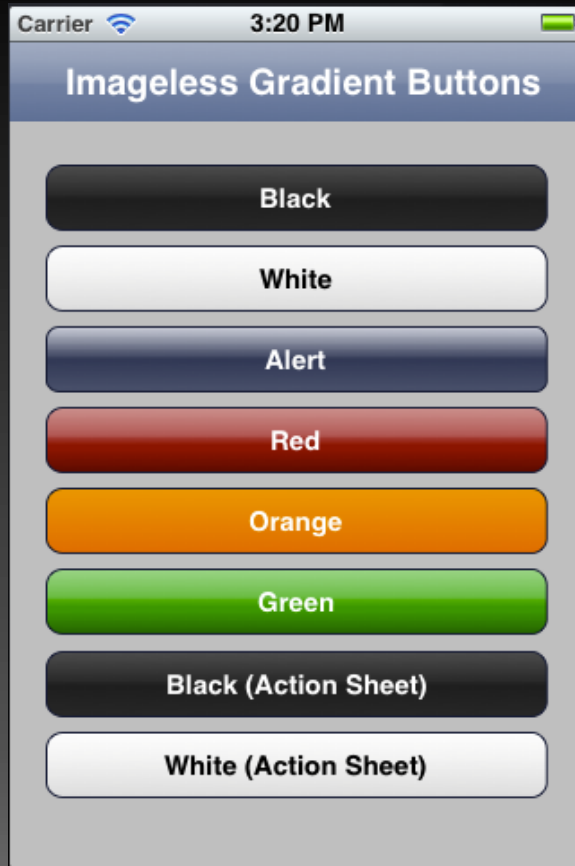
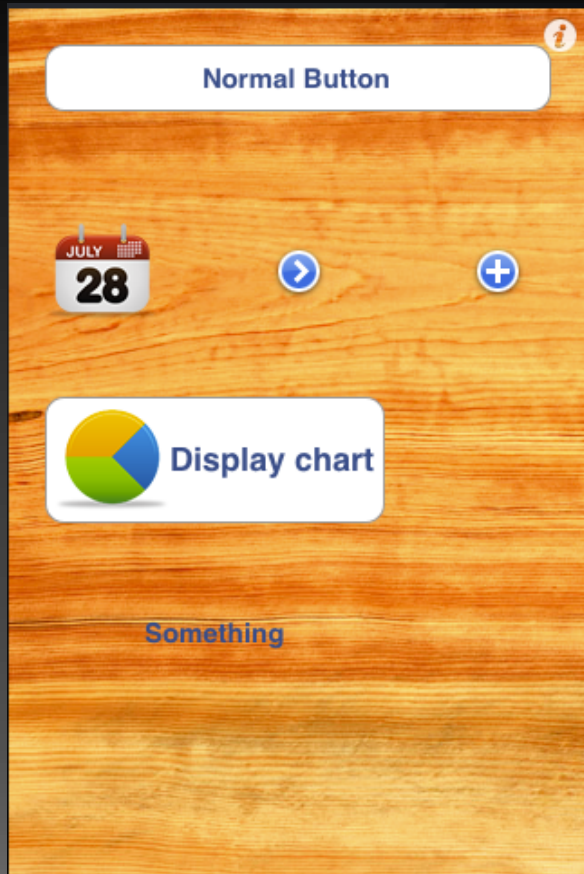
- Target – Action: chọn phương thức thực hiện qua @selector
 - Thường dùng để hứng sự kiện giao diện
- Delegate: yêu cầu đối tượng được ủy nhiệm (delegate) phải tuân thủ các hàm bắt buộc (required) trong protocol và tùy ý với hàm không bắt buộc (optional)
 - Dùng để hứng sự kiện giao diện, và nhiều việc khác nữa.



UIButton – UILabel – UIImageView - UITextField

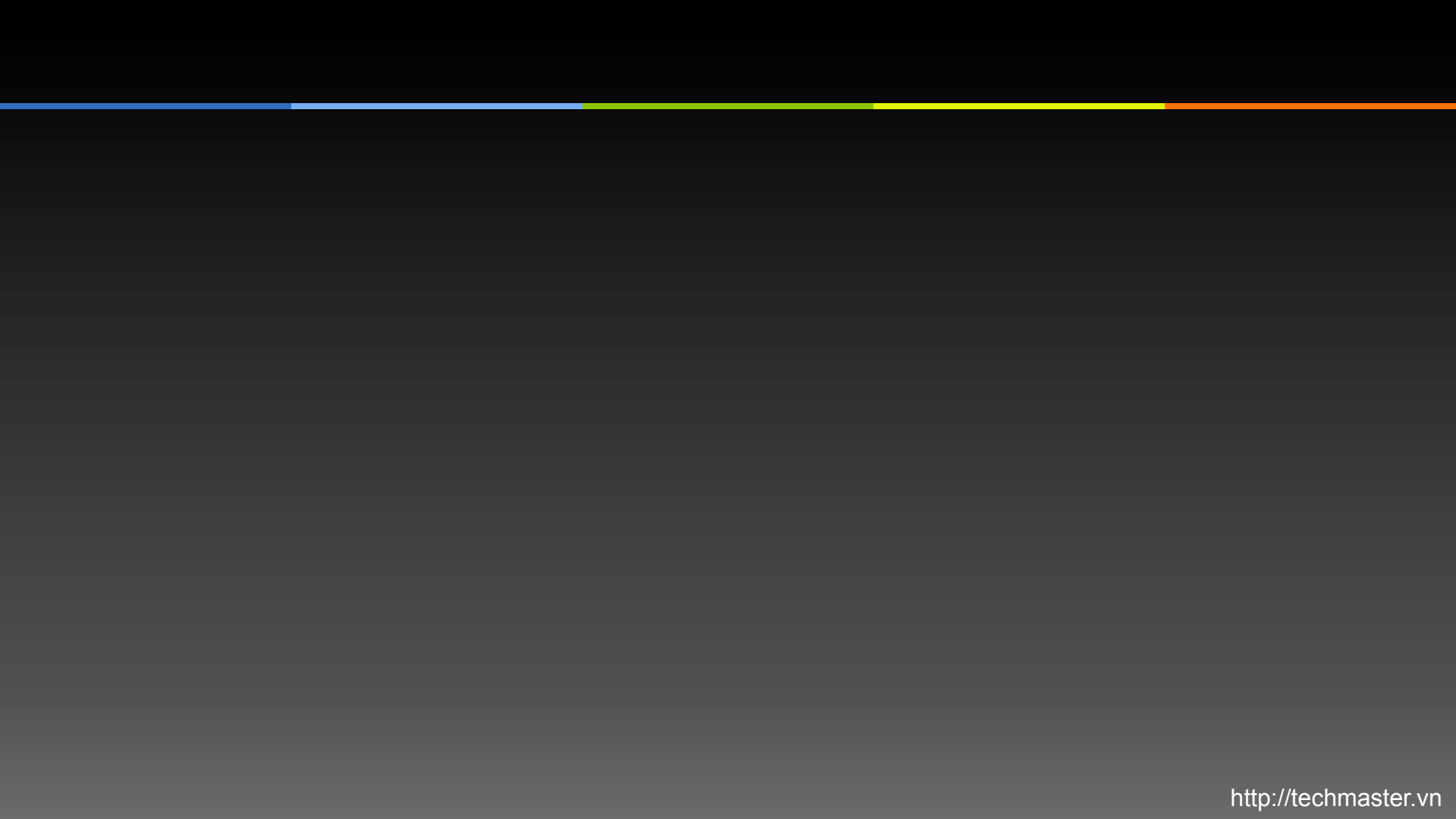


Buttons & Custom Buttons



UITextField

- UITextFieldDelegate
- Hiện và ẩn bàn phím ảo
- Đối với kiểu NumberPad



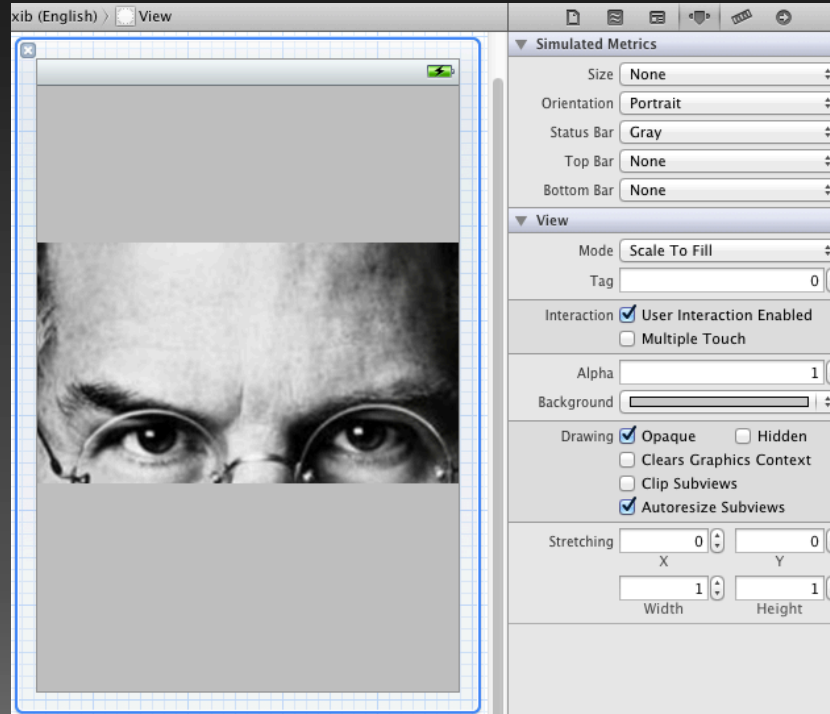
```
-(void)touchesBegan: (NSSet *)touches withEvent:(UIEvent *)event
{
    if (numberPadTextField.isFirstResponder) {
        [numberPadTextField resignFirstResponder];
    }
}
```

UIImageView

```
UIImageView *iconView = [[UIImageView alloc]  
initWithImage:[UIImage imageNamed:@"globe.png"]];
```

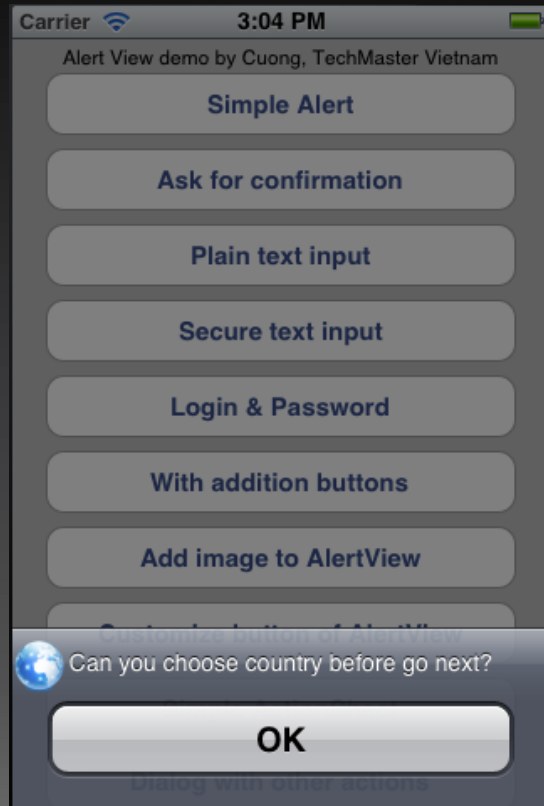
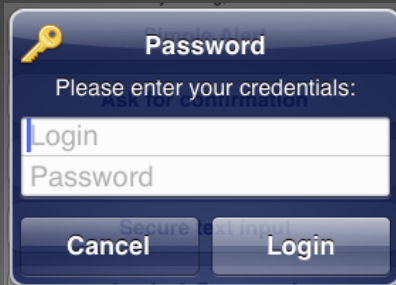
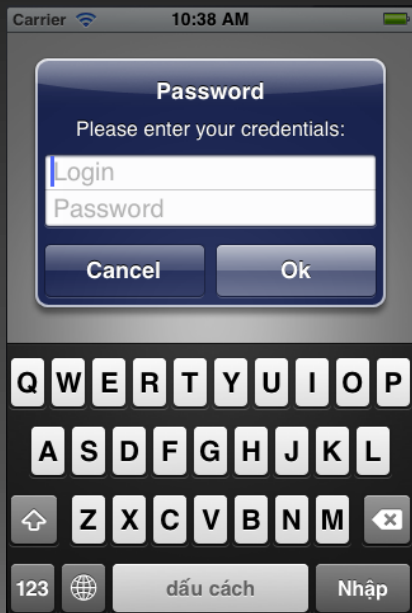
Demo

- ImageMode
- Bonfire

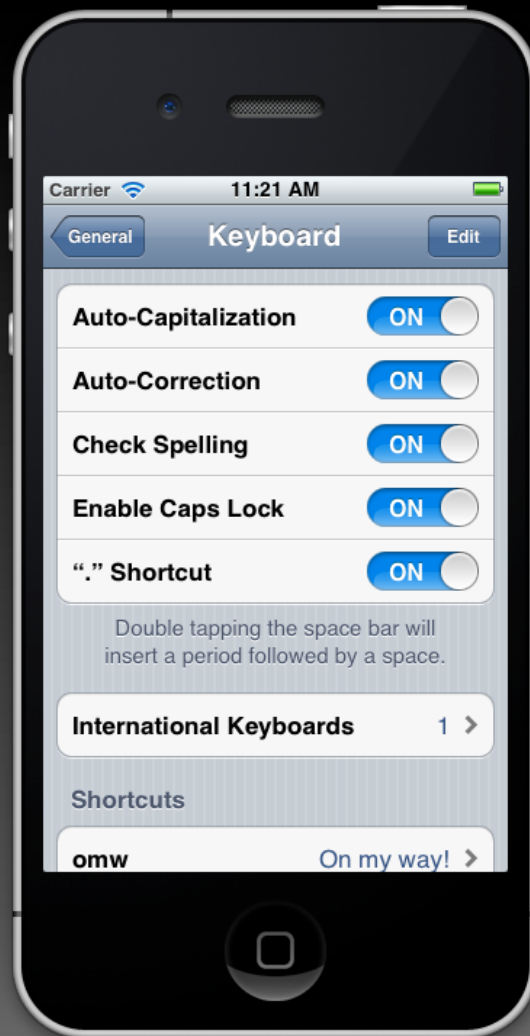


Views for making choice

- UIAlertView
- UIActionSheet



UISwitch



Set **ON**

```
[self.sdtSwitch setOn:YES];
```

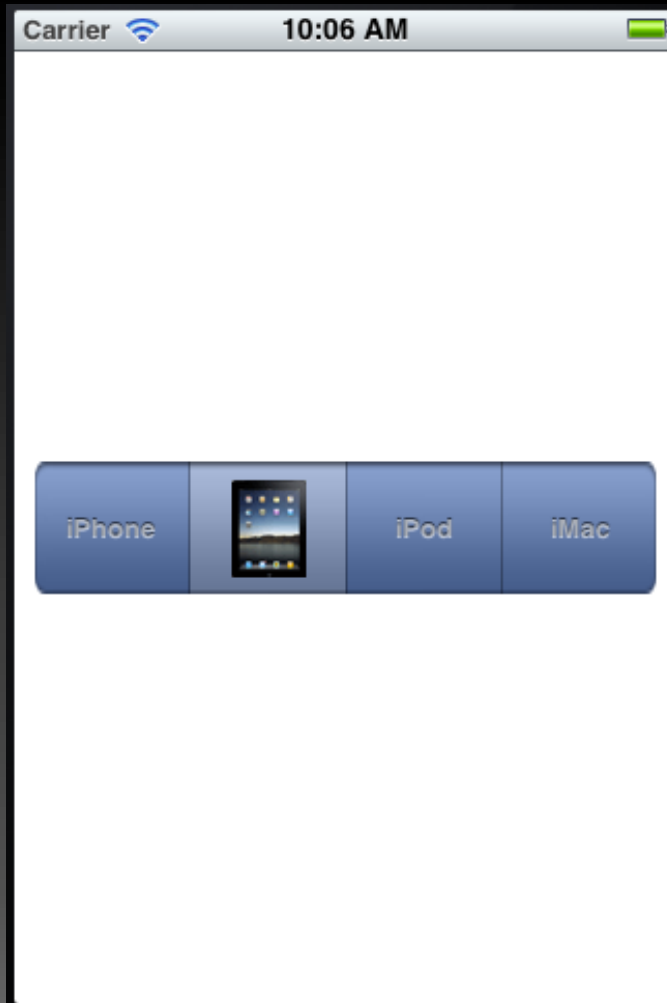
Set **OFF**

```
[self.sdtSwitch setOn:NO];
```

Get **current status**

```
if ([self.sdtSwitch isOn])  
    msg = @"Option is ON";  
else  
    msg = @"Option is OFF";
```

UISegmentedControl





```
NSArray *segments = [[NSArray alloc] initWithObjects:
    @"iPhone",
    [UIImage imageNamed:@"iPad.png"],
    @"iPod",
    @"iMac", nil];

self.mySegmentedControl = [[UISegmentedControl alloc]
    initWithItems:segments];

self.mySegmentedControl.segmentedControlStyle =
    UISegmentedControlStyleBezeled;
```

```
[self.mySegmentedControl addTarget:self  
    action:@selector(segmentChanged:)  
    forControlEvents:UIControlEventValueChanged];
```



```
- (void) segmentChanged:(UISegmentedControl *)paramSender{  
    if ([paramSender isEqual:self.mySegmentedControl]){  
        NSInteger selectedSegmentIndex = [paramSender selectedSegmentIndex];  
  
        NSString *selectedSegmentText =  
            [paramSender titleForSegmentAtIndex:selectedSegmentIndex];  
  
        NSLog(@"Segment %ld with %@ text is selected",  
            (long)selectedSegmentIndex,  
            selectedSegmentText);  
    }  
}
```

UISlider



minimumValue

maximumValue

value

How to customize the thumbnail image of a slider?

- (void)setThumbImage:(UIImage *)image
forState:(UIControlState)state;

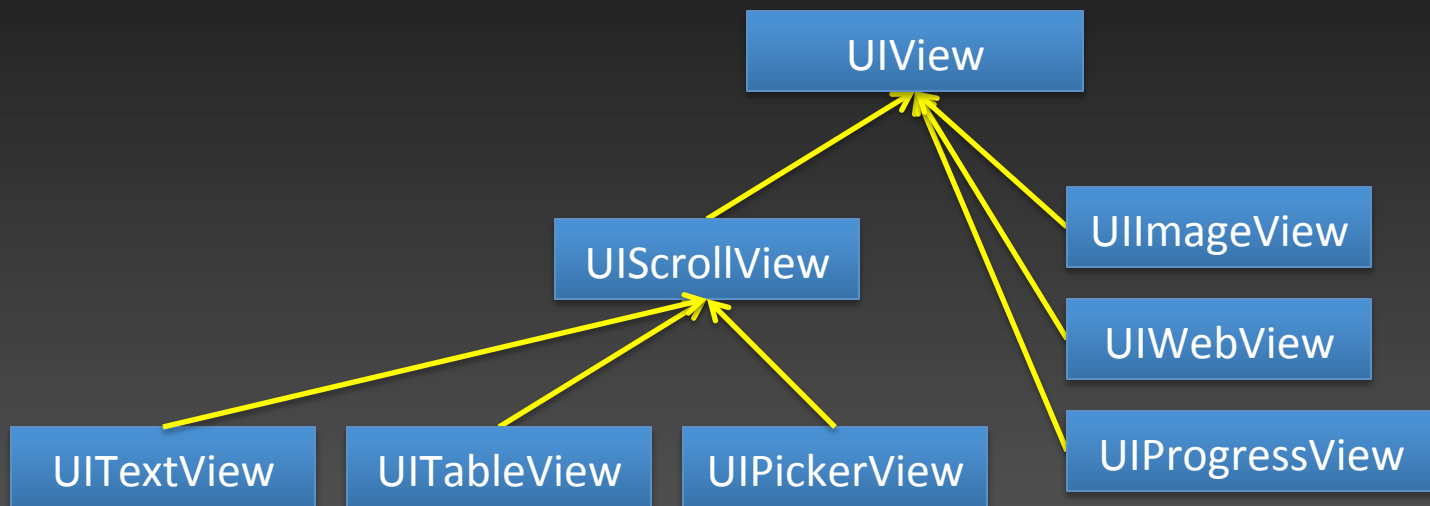


UISlider Demo

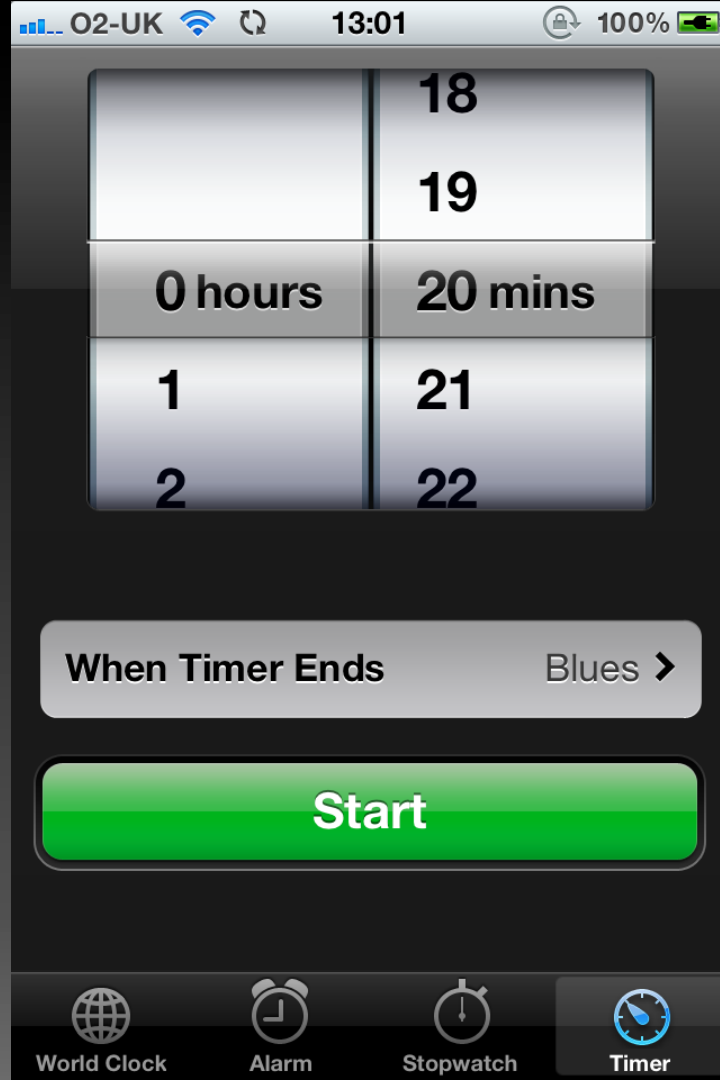
UIControl Preparing and Sending Action Messages

- `sendAction : to : forEvent:`
- `sendActionsForControlEvents:`
- `addTarget : action : forControlEvents:`
- `removeTarget : action : forControlEvents:`
- `actionsForTarget : forControlEvents:`
- `allTargets`
- `allControlEvents`

UIView Derivatives

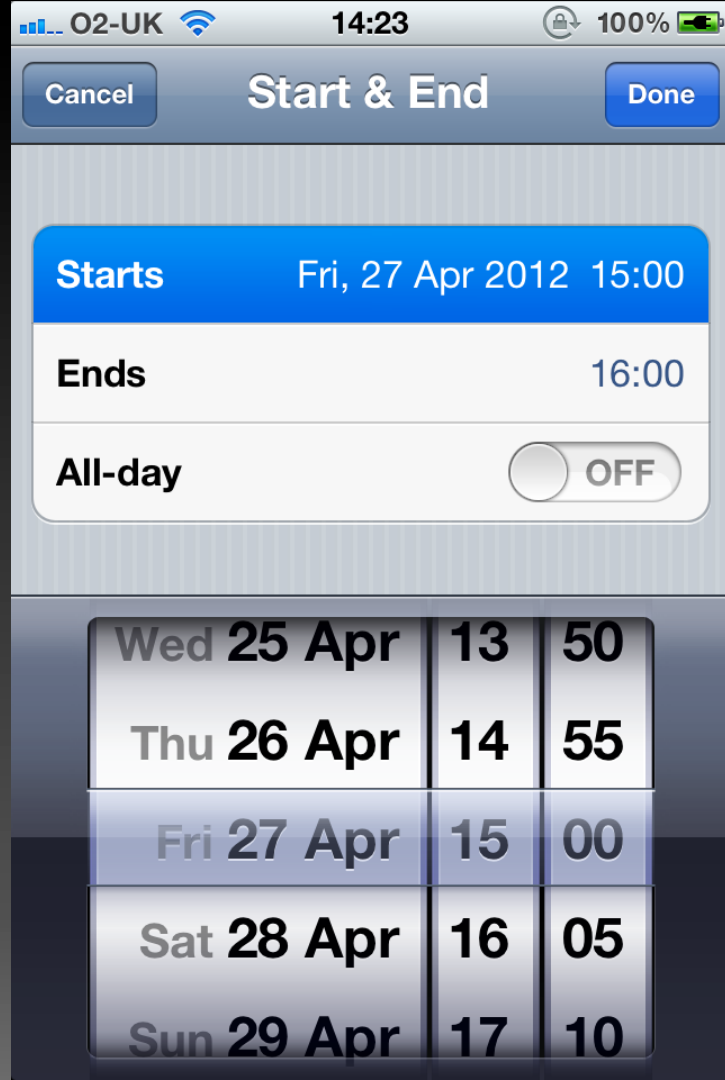


UIPickerView



- ✧ `numberOfComponentsInPickerView`: how many components to render.
- ✧ `pickerView:numberOfRowsInComponent`: how many rows to render
- ✧ `(NSString *)pickerView:(UIPickerView *)pickerView titleForRow:(NSInteger)rowforComponent:(NSInteger)component`: displaying the row for the component.

UIDatePicker

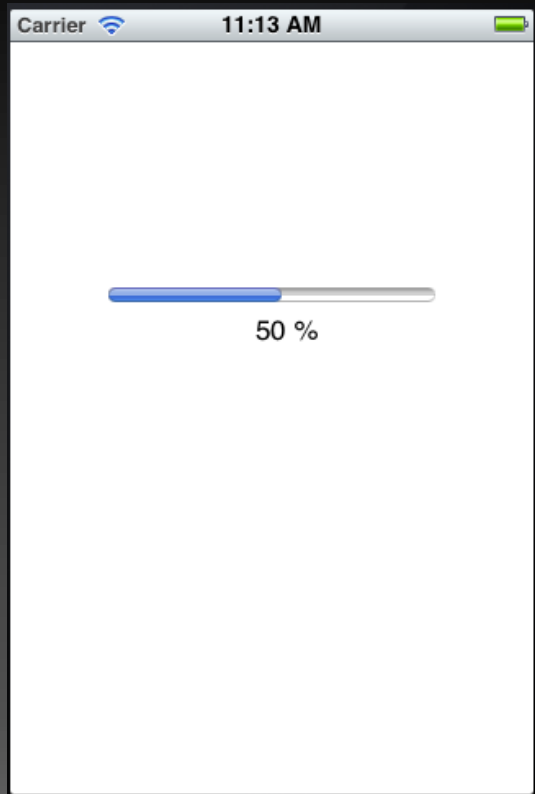


UIDatePicker does not inherit from UIPickerView, but it manages a custom picker-view object as a subview.

```
UIKIT_CLASS_AVAILABLE(2_0)
@interface UIDatePicker : UIControl <NSCoding>
{
    @private
    UIPickerView *_pickerView;
}
```



UIProgressView Demo



✧ scrollViewDidScroll

✧ scrollViewWillBeginDecelerating

✧ scrollViewDidEndDecelerating

✧ scrollViewDidEndDragging:willDecelerate



UIScrollView Demo