iPhone与iPad应用开发课程
精通iOS开发

第三讲 基本UI控件

主讲人：关东升

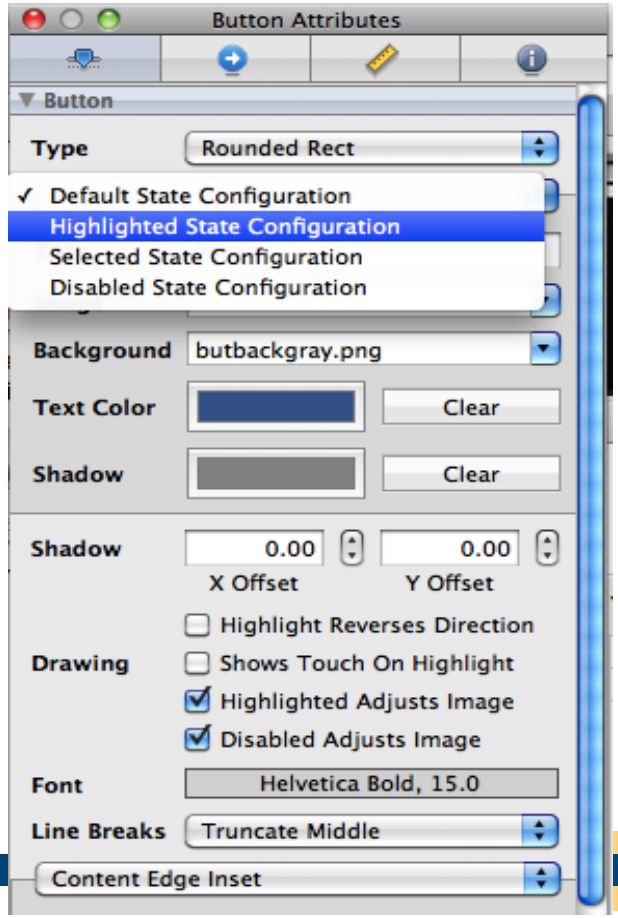
eorient@sina.com

主要知识点

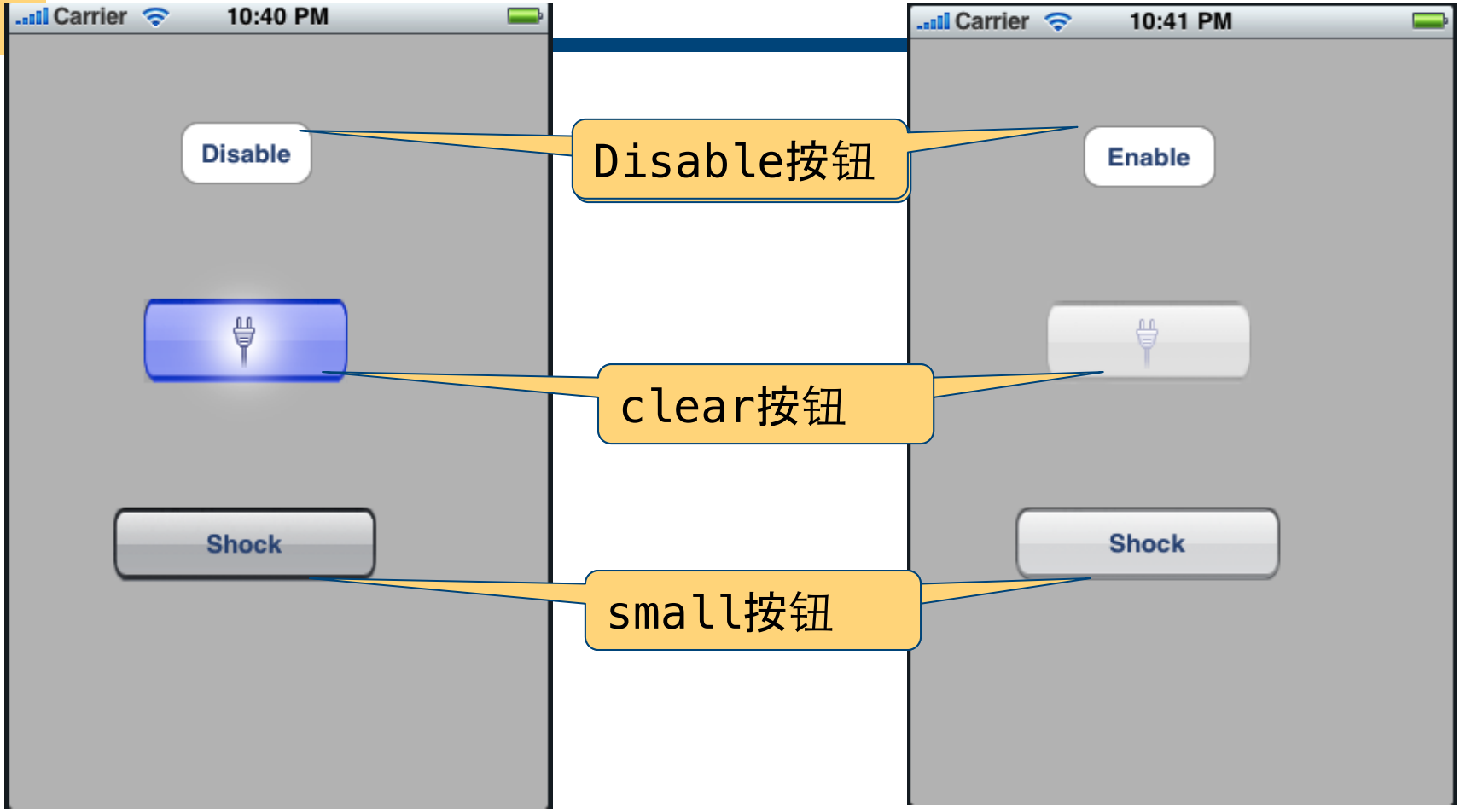
- Button控件
- 开关控件
- 分段控件
- 滑块控件
- WebView

Button控件

- iPhone的Button控件可以做的很绚丽，Button可以有多种状态：
  - Default State
  - Highlighted State
  - Selected State
  - Disabled State



实例代码：ButtonsBackground


Carrier
10:40 PM
Disable
Shock
Carrier
10:41 PM
Enable
Shock
Disable按钮
clear按钮
small按钮

ButtonsBackgroundViewController.h文件

```
@interface ButtonsBackgroundViewController :
     UIViewController {

    IBOutlet UIButton * clearButton;
    IBOutlet UIButton * smallButton;
}

@property (nonatomic, retain) IBOutlet UIButton
* clearButton;
@property (nonatomic, retain) IBOutlet UIButton
* smallButton;

- (IBAction) disableBut: (id) sender;

@end
```

ButtonsBackgroundViewController.m文件

```
@synthesize clearButton;
@synthesize smallButton;

- (IBAction) disableBut: (id) sender {
    if(clearButton.enabled == YES) {
        clearButton.enabled = NO;
        smallButton.enabled = NO;
        [((UIButton *) sender)
            setTitle:@"Enable"forState:UIControlStateNormal];
    } else {
        clearButton.enabled = YES;
        smallButton.enabled = YES;
        [((UIButton *) sender)
            setTitle:@"Disable"forState:UIControlStateNormal];
    }
}

- (void)dealloc {
    [clearButton release];
    [smallButton release];
    [super dealloc];
}
```

说明

- 点击Disable按钮时候，调用disableBut方法，在该方法中实现了clearButton按钮和smallButton按钮“可用”状态和“不可用”状态的切换。在状态发生切换时候还要改变Disable按钮的上面的“标题”和“状态”：
- ```
  [((UIButton *) sender)
  setTitle:@"Enable"forState:UIControlStateNormal];
  ```
- sender是事件源即点击的按钮对象本身。
  ```

Interface Builder设计页面

- 点击Disable按钮的时候，会改变clearButton和smallButton的标题，因此需要连接File’s Owner到两个按钮（clearButton和smallButton）的输出口。
- 为了响应Disable按钮的事件需要，从Disable按钮连接到File’s Owner定义的disableBut事件。

clearButton属性设定

- 属性框，使Shows Touch on Highlight is checked选项被选中。
- Default State Configuration选中时候，设置按钮图片power.png背景图片butbackgray.png。
- Highlighted State Configuration 选中时候，设置按钮图片power.png ,背景图butbackbluegray.png。
- Disabled State Configuration 选中时候，设置按钮图片powerdisabled.png背景图片,butbackgraydisabled.png。

smallButton属性设定

- Default State Configuration选中时候，设置按钮图片butbackgray.png背景图片， 设置title“Shock”。
- Highlighted State Configuration 选中时候，设置图片butbackbluegray.png背景图片，设置title“Shocking”。

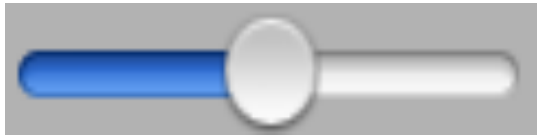
开关控件

- 开关控件（Switch），有些相windows中的checkbox，它只有两种状态，true和false。

- 可以通过该方法改变开关控件的状态。
- -(void) setOn: (BOOL) on animated: (BOOL) animated

滑块控件

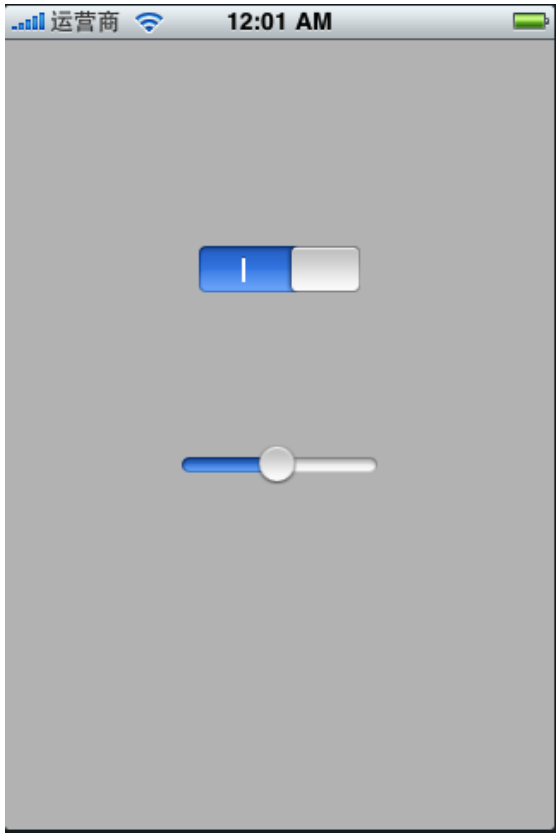
- 滑块控件（Slider），水平放置，可以用手触摸改变它的值。



- 默认情况下它的最小值0最大值1.00，而.50是初始值，我们可以通过下面的方法设定值：
- - (void) setValue:(float) value animated:(BOOL) animated

开关和滑块实例

- 我们通过在页面放在开关和滑块控件了解他们的使用情况。



实例代码：SwitchSlider

SwitchSliderViewController.h

```
@interface SwitchSliderViewController :
UIViewController {
    UISwitch * mySwitch;
}

@property(nonatomic, retain)IBOutlet UISwitch
* mySwitch;

-(IBAction) handleSwitch: (id) sender;
-(IBAction) handleSlider: (id) sender;

@end
```

SwitchSliderViewController.m

```
@implementation SwitchSliderViewController

@synthesize mySwitch;

- (IBAction) handleSwitch: (id) sender {
    if( [((UISwitch *) sender) isOn] == YES){
        NSLog(@"It's on");
    } else {
        NSLog(@"It's off");
    }
}

- (IBAction) handleSlider: (id) sender {
    NSLog(@"value: %f", ((UISlider *)sender).value);
    if( [((UISlider *) sender) value] == ((UISlider *)
sender) .maximumValue) {
        [mySwitch setOn:YES animated:YES];
    }
}

- (void)dealloc {
    [mySwitch release];
    [super dealloc];
}
@end
```

连接输出口和动作事件

- 连接开关控件到的handleSwitch: 动作。
- 连接滑块控件到的handleSlider: 动作。
- 制定开关控件输出口。

分段控件

- 分段控件（Segment），是有2个或更多段构成的组，它相当与独立的按钮。

实例代码：Segment

SegmentViewController.h

- 定义一个动作事件

```
#import <UIKit/UIKit.h>

@interface SegmentViewController :
UIViewController {

}

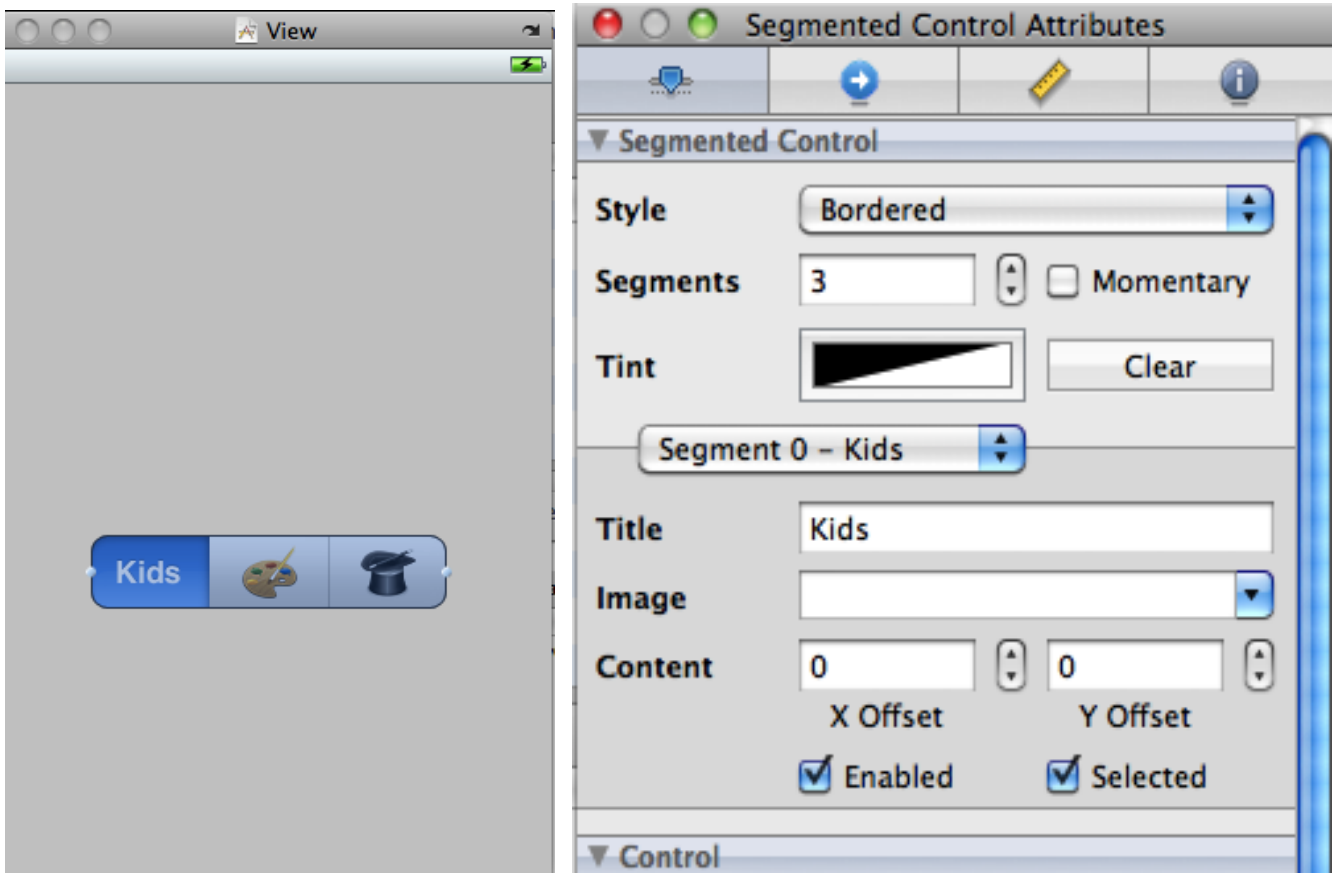
- (IBAction) handleSegment: (id) sender;

@end
```

SegmentViewController.m

```
@implementation SegmentViewController
- (IBAction) handleSegment: (id) sender {
    UISegmentedControl * myseg =
(UISegmentedControl *) sender;
    if(myseg.selectedSegmentIndex == 0) {
        NSLog(@"selected zero index...");
    }
    else if(myseg.selectedSegmentIndex == 1) {
        NSLog(@"selected one index...");
    }
    else {
        NSLog(@"selected two index...");
    }
}
- (void)dealloc {
    [super dealloc];
}
```

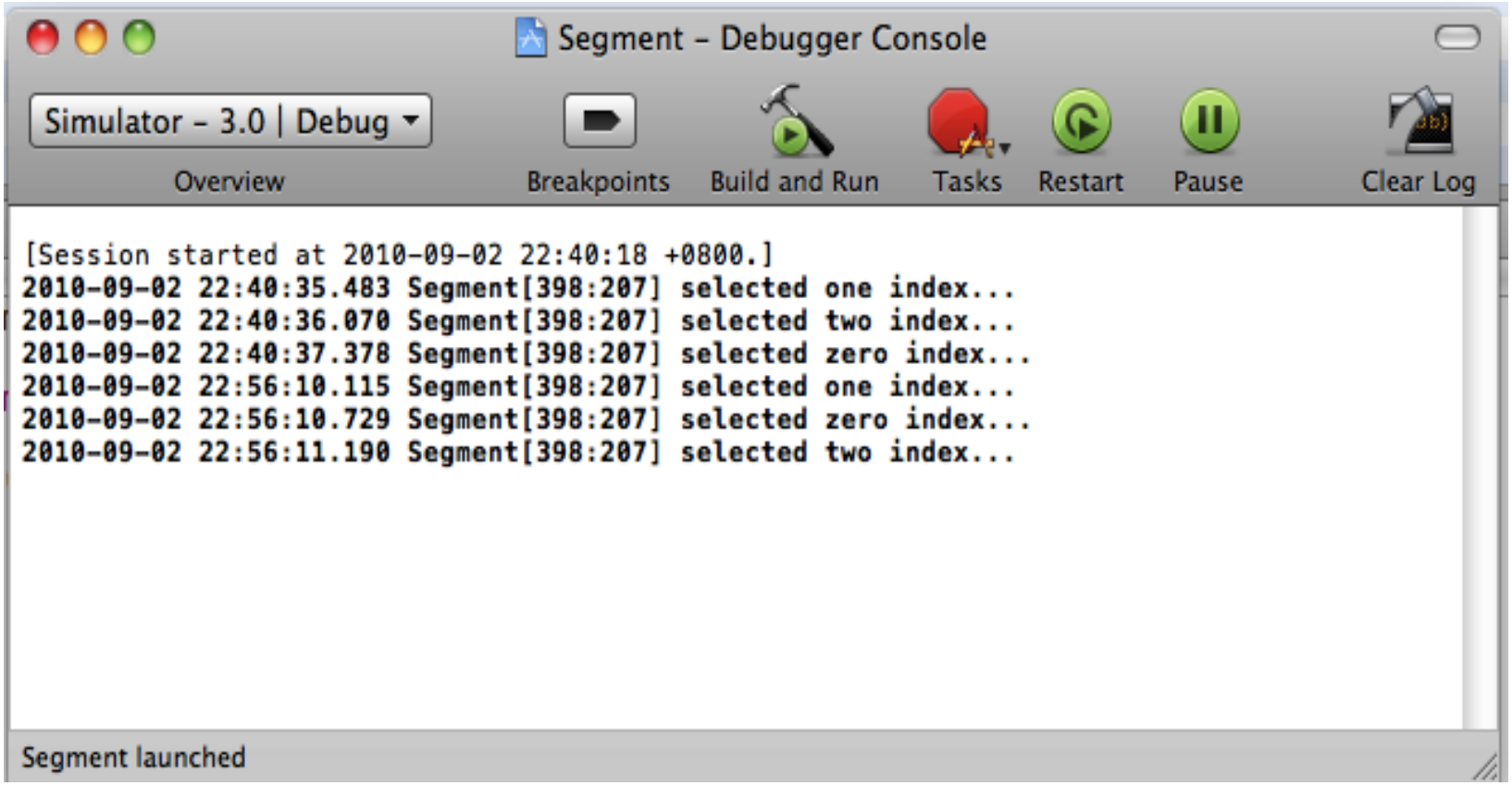
IB设计视图


View
Kids
Segmented Control Attributes
Segmented Control
Style
Bordered
Segments
3
Momentary
Tint
Clear
Segment 0 – Kids
Title
Kids
Image
Content
0
0
X Offset
Y Offset
Enabled
Selected
Control

连接动作事件

- 连接段控件到File’s Owner的handleSegment: 动作。

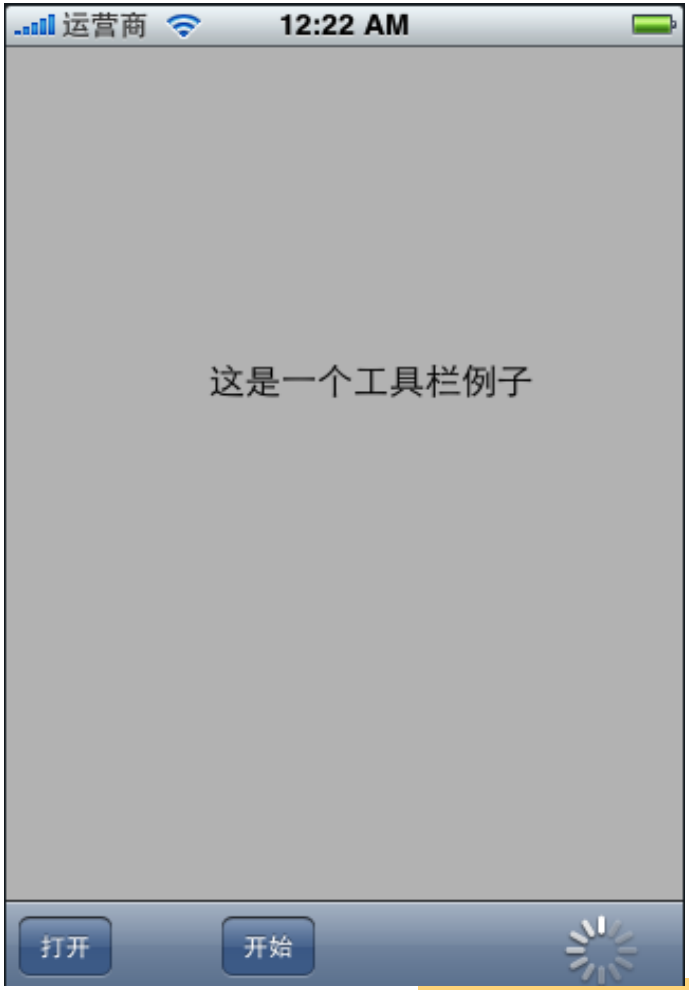
输出结果到控制台


Segment – Debugger Console
Simulator – 3.0 | Debug
Overview
Breakpoints
Build and Run
Tasks
Restart
Pause
Clear Log
[Session started at 2010-09-02 22:40:18 +0800.]
2010-09-02 22:40:35.483 Segment[398:207] selected one index...
2010-09-02 22:40:36.070 Segment[398:207] selected two index...
2010-09-02 22:40:37.378 Segment[398:207] selected zero index...
2010-09-02 22:56:10.115 Segment[398:207] selected one index...
2010-09-02 22:56:10.729 Segment[398:207] selected zero index...
2010-09-02 22:56:11.190 Segment[398:207] selected two index...
Segment launched

工具栏

- 工具栏（UIToolBar），一般是放置在屏幕的底部，在工具栏的内部可以放置多个按钮和控件。

实例代码：ToolBar



ToolBarViewController.h

- 定义两个动作事件

```
@interface ToolBarViewController : UIViewController {

   IBOutlet UIActivityIndicatorView * myActivityView;
}

@property (nonatomic, retain) IBOutlet
     UIActivityIndicatorView * myActivityView;

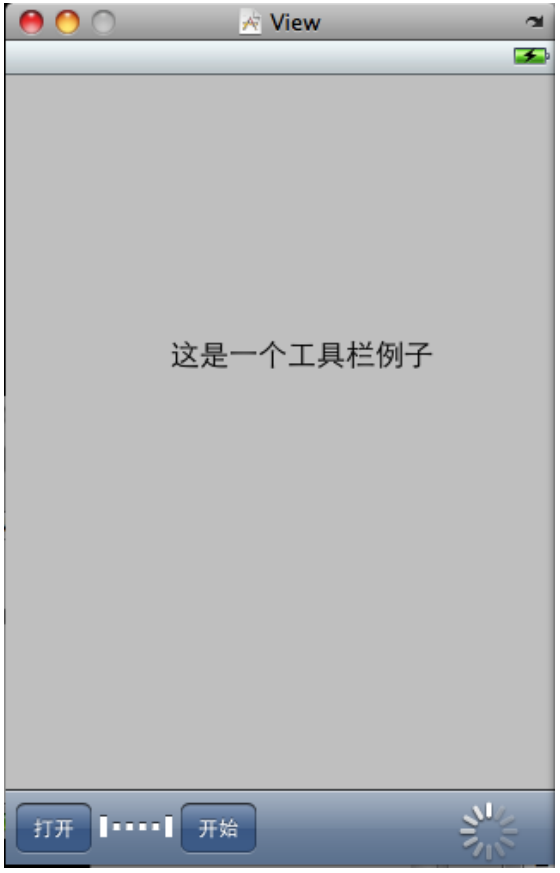
-(IBAction)onClickStartButton: (id)sender;
-(IBAction)onClickOpenButton: (id)sender;

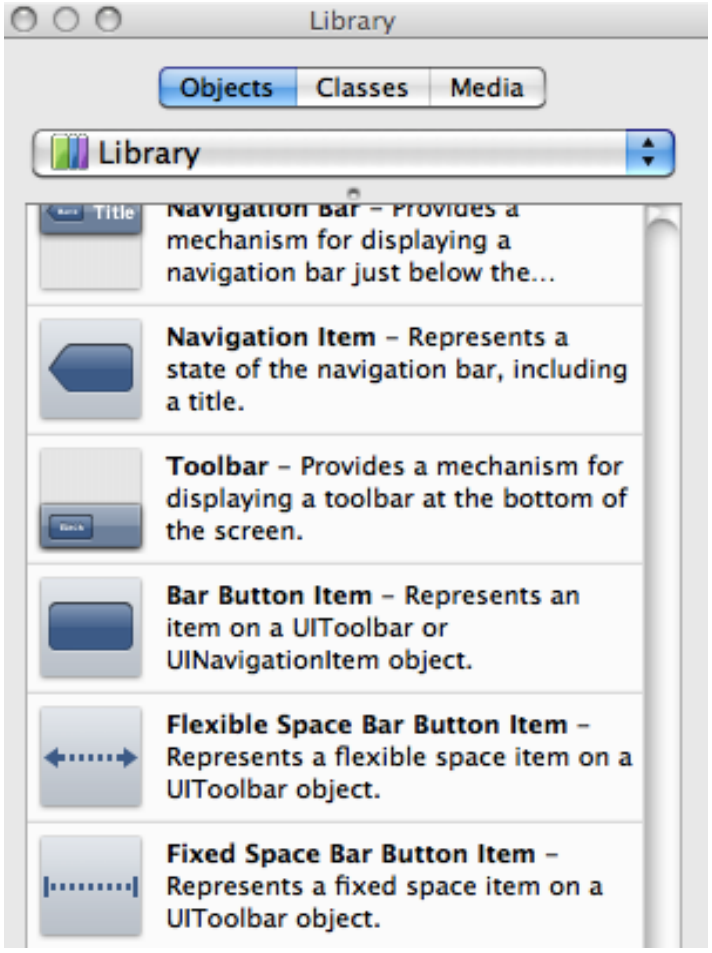
@end
```

ToolBarViewController.m

```
@synthesize myActivityView;
-(IBAction)onClickStartButton: (id)sender {
    if ([myActivityView isAnimating]) {
        [myActivityView stopAnimating];
    } else {
        [myActivityView startAnimating];
    }
}
-(IBAction)onClickOpenButton: (id)sender {
    UIAlertView *alert = [[UIAlertView alloc]
                          initWithTitle:@"提示信息"
                          message:@"您点击了打开按钮”
                           delegate:self
                          cancelButtonTitle:@"Done"
                          otherButtonTitles:nil];
    [alert show];
    [alert release];
}
@end
```

IB设计视图


View
这是一个工具栏例子
打开
开始


Library
Objects
Classes
Media
Library
Navigation Bar – Provides a mechanism for displaying a navigation bar just below the...
Navigation Item – Represents a state of the navigation bar, including a title.
Toolbar – Provides a mechanism for displaying a toolbar at the bottom of the screen.
Bar Button Item – Represents an item on a UIToolbar or UINavigationItem object.
Flexible Space Bar Button Item – Represents a flexible space item on a UIToolbar object.
Fixed Space Bar Button Item – Represents a fixed space item on a UIToolbar object.

连接动作事件

- 连接到打开按钮的onClickOpenButton: 动作。
- 连接到开始按钮的onClickStartButton: 动作。
- 连接到UIActivityIndicatorView输出口。

WebView

- WebView可以帮助我们构建Web的iPhone应用程序。
- 很多网站的iPhone和iPad客户端程序都是使用WebView开发的。
- WebView能够支持HTML5，不支持Flash等。

实例代码：MyWeb

MyWebViewController.h

```
@interface MyWebViewController : UIViewController
      <UIWebViewDelegate> {
   IBOutlet UITextField * myTextField;
   IBOutlet UIWebView * myWebView;
}

@property(nonatomic, retain) UIWebView * myWebView;
@property(nonatomic, retain) UITextField * myTextField;

- (IBAction) changeLocation: (id) sender;

@end
```

说明

- MyWebViewController 需要实现协议 UIWebViewDelegate，它一个委托对象。委托是一种设计模式，在iPhone中主要用于回调事件。在委托中定义了一下方法，实现了该委托的对象，要提供该方法的实现。
- UIWebViewDelegate的方法是 webViewDidFinishLoad:
- 它上在异步情况一个网址，当应答回来后回调该方法。

MyWebViewController.m

```
@implementation MyWebViewController

@synthesize myWebView;
@synthesize myTextField;

- (void) viewDidLoad {
    myWebView.delegate = self;
}

- (void)dealloc {
    myWebView.delegate = nil;
    [myTextField release];
    [myWebView release];
    [super dealloc];
}
```

MyWebViewController.m

```
- (IBAction) changeLocation: (id) sender {
    [myTextField resignFirstResponder];
    NSURL * url = [NSURL URLWithString: myTextField.text];
    NSURLRequest * request = [NSURLRequest
          requestWithURL:url];
    [myWebView loadRequest:request];
}

#pragma mark WebView 委托
#pragma mark --
- (void)webViewDidFinishLoad: (UIWebView *) webView {
    NSLog(@"%@", [webView
       stringByEvaluatingJavaScriptFromString:
         @"document.body.innerHTML"]);
}

@end
```

说明

- 在viewDidLoad 方法中的myWebView.delegate = self是指定为自身。
- webViewDidFinishLoad方法中实现委托回调功能。
- [myWebView stringByEvaluatingJavaScriptFromString: @"document.documentElement.textContent”]
- 是运行一个JavaScript脚本程序，document.body.innerHTML获得页面中的HTML代码。