UI-Signal Extension

基于{Bee}框架的技术方案

老郭为人民服务

www.Geek-Zoo.com

目的

1. 吸引更多开源技术开发者加入 Bee
2. 吸引更多优秀的开源控件服务于 Bee
3. 通过 Bee UI-Signal 技术本身来统一第三方控件使用规范
4. 提供开源控件展示平台，供网友使用

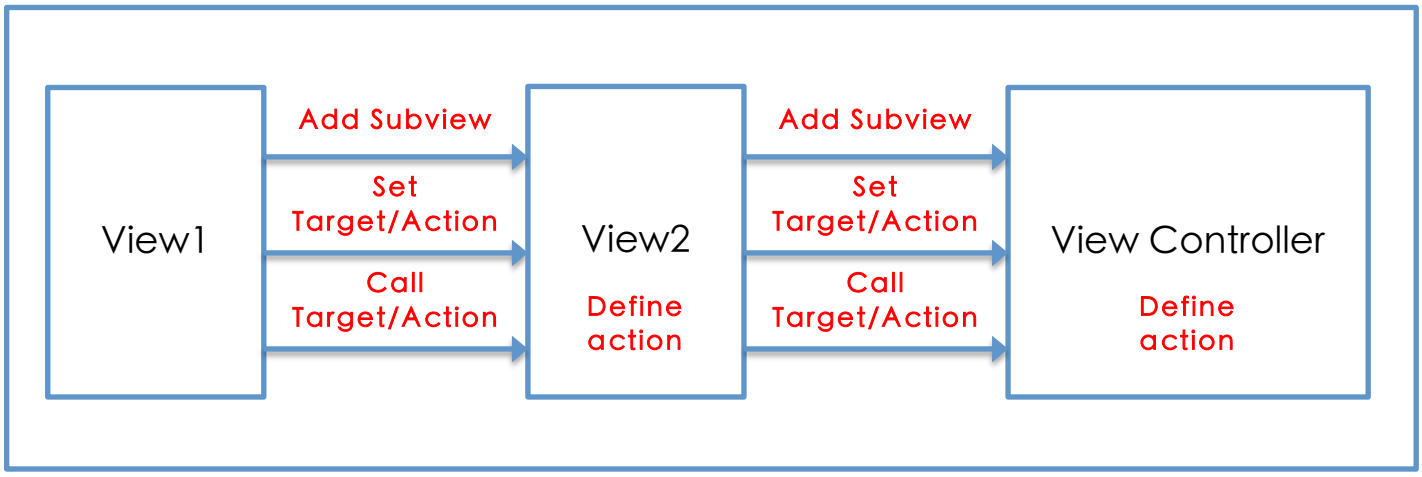
优点

1. 语义明确的事件名称定义
2. 简单高效的事件路由机制

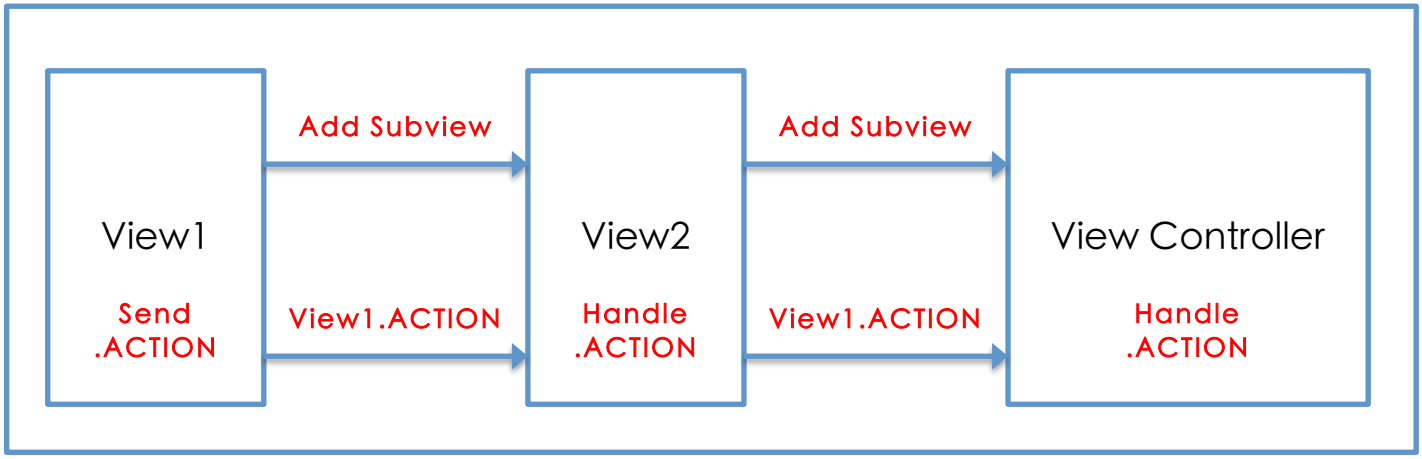
知识点

1. Bee UI-Signal
2. Objective-C Extension

Delegate 实现方式（旧）



UI-Signal 实现方式（新）



相比传统的 Delegate 方式，UI-Signal 事件路由机制有着明显的优势：
1. 带语义的事件命名方式，如：View1.ACTION。
2. 带语义的事件处理方式，如：handleUISignal_View1。
3. 去掉 Target/Action，依赖于 view 层级关系，自底向上传递。

UI-Signal 基础教程 - 定义和使用

1. 声明 Signal

```
@interface View1 : UIView
AS_SIGNAL( ACTION )
@end
```

2. 定义 Signal

```
@implementation View1
DEF_SIGNAL( ACTION )
@end
```

3. 引用 Signal

```
View1.ACTION;
```

4. 发送 Signal

```
@implementation View1
…
{
    [self sendUISignal:self.ACTION];      // 同 View1.ACTION
    [self sendUISignal:View1.ACTION];
    [self sendUISignal:View1.ACTION withObject:nil];
    [self sendUISignal:View1.ACTION withObject:nil from:otherView];
    [self sendUISignal:View2.ACTION];
}
…
@end
```

5. 接受 Signal

```
@implementation ViewController

- (void)handleUISignal:(BeeUISignal *)signal
{
}

@end
```

6. 分类过滤 Signal

```
@implementation ViewController

- (void)handleUISignal:(BeeUISignal *)signal
{
    // 未处理的 Signal 走这里
}
- (void)handleUISignal_View1:(BeeUISignal *)signal
{
    // 从 View1 来的走这里
}
- (void)handleUISignal_View2:(BeeUISignal *)signal
{
    // 从 View2 来的走这里
}

@end
```

7. 使用 Signal

```
@implementation ViewController

- (void)handleUISignal:(BeeUISignal *)signal
{
    if ( [signal is:View2.ACTION] )
    {
          View2 * view2 = signal.source;
          NSObject * object = signal.object;
          // do something with view2
    }
}

- (void)handleUISignal_View1:(BeeUISignal *)signal
{
    if ( [signal is:View1.ACTION] )
    {
          View1 * view1 = signal.source;
          NSObject * object = signal.object;
          // do something with view1
    }

}
@end
```

8. 透传 Signal

```
@implementation ViewController

- (void)handleUISignal:(BeeUISignal *)signal
{
    // TODO something with signal

    [super handleUISignal];
}

@end
```

9. 举个荔枝

```
@interface BeeUIButton : UIButton

AS_SIGNAL( TOUCH_DOWN )        // 按下
AS_SIGNAL( TOUCH_DOWN_REPEAT ) // 长按
AS_SIGNAL( TOUCH_UP_INSIDE )   // 抬起（击中）
AS_SIGNAL( TOUCH_UP_OUTSIDE )  // 抬起（未击中）
AS_SIGNAL( TOUCH_UP_CANCEL )   // 撤销

@end

@implementation BeeUIButton

- (void)didTouchDown
{
    if ( NO == [self testEvent:UIControlEventTouchDown] )
    {
        [self sendUISignal:BeeUIButton.TOUCH_DOWN];
    }
}

@end
```

UI-Signal 高级教程 - 第三方控件集成

1. 原理

   通过 Delegate 重定向，将 Target 及 Action 指向第三方控件自己，达到自实现 Delegate method 的方式，转化为 UISignal 事件。

2. 扩展文件

   假设第三方控件名为 CustomSlider，需要扩展两个文件：
   CustomSlider+BeeUISignal.h
   CustomSlider+BeeUISignal.m

3. 注入 Signal

```
// CustomSlider+BeeUISignal.h

@interface CustomSlider()
AS_SIGNAL( VALUE_CHANGED )
@end

// CustomSlider+BeeUISignal.m

@implementation CustomSlider()
DEF_SIGNAL( VALUE_CHANGED )
@end
```

4. 注入 Constructor

   a. 头文件

```
// CustomSlider+BeeUISignal.h

 @interface CustomSlider()
 - (void)useUISignal;
 @end
```

   b. 源文件 - 方式 1（自代理）

```
// CustomSlider+BeeUISignal.m

@implementation CustomSlider()
- (void)useUISignal
{
      self.delegate = self;
}
-  (void)customSlider:(CustomSlider *)slider changed:(NSNumber *)value
{
      [self sendUISignal:self.VALUE_CHANGED withObject:value];
}
@end
```

c. 源文件 - 方式 2（Agent 代理）

```
// CustomSlider+BeeUISignal.m

@interface CustomSliderAgent<CustomSliderDelegate>
@property CustomSlider * slider;
@end

@implementation CustomSliderAgent
@synthesize slider;
-  (void)customSlider:(CustomSlider *)slider changed:(NSNumber *)value
{
      [slider sendUISignal:CustomSlider.VALUE_CHANGED
withObject:value];
}
@end

@implementation CustomSlider()
- (void)useUISignal
{
      CustomSliderAgent * agent = [[CustomSliderAgent alloc] init]
      agent.slider = self;

      self.delegate = agent;

      objc_setAssociatedObject(
            self,
            "__agent",
            agent,
            OBJC_ASSOCIATION_RETAIN );

      [agent release];
}
@end
```

5. Test

```
…
{
    CustomSlider * slider = [[CustomSlider new] autorelease];
    [slider useUISignal];
    [self.view addSubview:slider];
}

- (void)handleUISignal_CustomSlider:(BeeUISignal *)signal
{
    if ( [signal is:CustomSlider.CHANGED] )
    {
         // do something
    }
}
```