### 开发设计规范

1. 代码重构规范
   1. 重构目地:把重复代码抽取出来进行统一维护;把功能相对独立的代码抽取出来,以便于后续修改与维护.
   2. 【格式】重构方法:新建方法,调整参数,调整代码结构,调整注释,测试.
   3. 【建议】重构时间:开发进行一段时间后,出现大量复制粘贴代码,可以考虑开始重构,重构时间过晚会造成因代码逻辑结构复杂程度较高的重构失败.
2. BUG总结与产生原因
   1. 【说明】模拟器点击textField或者textView不显示键盘[解决方法:按下command + shift + k]
   2. 【建议】键盘弹出与收回[解决方法:弹出可选择textfield或textview作为第一响应者;收回可监听return键点击,收回键盘(需实现代理方法)]
   3. 【规则】友盟社会化组件无法客户端分享:[解决方法:检查target->Info->URL Types是否对需要的客户端分享进行设置]
   4. 【规则】图片加载[本地图片加载非png格式需添加图片后缀,SDWebImage加载注意保证URL完整性]
   5. 【规则】unrecognized seletor。错误：这种情况很简单，给一个对象发送了一条它不认识的消息。
   6. 【规则】数组越界导致崩溃.[解决方法:检查引用数组时是否有越界引用行为].
   7. 【规则】TableView&CollectionView数据源方法引用崩溃[检查数据源方法必选方法是否实现]
3. UIImageView显示问题

4.1在默认情况下，如果没有设置UIImageView的尺寸，UIImageView默认会跟图片大小一致，如果设置了尺寸，显示的图片会被压缩或者拉伸以填满整个区域。

4.2【建议】如果不想要图片被拉伸或者压缩的太严重，可以给contentMode设置UIViewContentModeScaleAspectFit。

4.3 【规则】但是上面的设置，还有另外一种情况，就是图片不规则的时候，图片的宽或者高，比frame更小的时候，会出现空白，怎么解决这个问题了，可以给contentMode设置UIViewContentModeScaleAspectFill。但是这样设置还不够，因为在默认情况，图片多出来的部分还是会显示屏幕上，如果不希望超过frame的区域显示在屏幕上需要设置clipsToBounds为YES。  
4.4 【规则】UIImageView和UIView显示的区别：如果将控件添加到UIView中，如果控件的位置超出了UIView的尺寸，就不会显示。但是如果将控件添加到UIImageView，如果控件的位置超出了UIImageView的尺寸，还是会显示的。如果不想让它显示，必须设置clipsToBounds属性。

1. 视图和视图控制(UIView和UIViewController)

4.1 UIView --UIViewController

4.1.1UIView 负责界面的显示

4.1.2UIViewController 负责界面元素及内容的控制和调度

4.1.3每个UIViewController默认都有一个UIView属性,每一个UIView都是一个容器,容纳其他UIView

4.2程序启动过程简述:

读取storyboard.storyboard

->创建ViewController

->根据视图storyboard文件中的描述创建ViewController的UIView对象(无stroryboard使用代码加载)

->将UIView对象显示到手机屏幕上

->监听并响应用户交互

4.3 viewDidLoad方法

在加载完storyboard.storyboard后自动执行

override func viewDidLoad() {

super.viewDidLoad()

}

4.4 UIView的常用属性

superview :获得父视图

subviews :获得所有子视图

4.5 frame.bounds.center

* 1. frame

控件所在矩形框的位置和尺寸（以父控件的左上角为坐标原点）

* 2. bounds

控件所在矩形框的位置和尺寸（以自己左上角为坐标原点，所以bounds的x和y永远为0）w,h

* 3. center

控件中点的位置（以父控件的左上角为坐标原点），通过center属性设置视图位置更加方便

* 注意
  + 用frame和center可以修改UIView的位置
  + 用frame和bounds可以修改UIView的尺寸
  + 通常：修改自身位置大小时使用frame，而子视图（控件）在设置位置时，会相对父视图的bounds进行设置

通过frame.orgin.x 和frame.orgin.y的操作对组件进行位置操作

4.6 tag属性

控件的ID（标记），父控件可以通过tag来找到或区分子控件

【规则】通过tag来实现多个button点击事件指向一个函数,但是注意函数传入的参数要是(UIButton \*)sender

5. **UITextField**

5.1键盘控制:

【建议】需要注意关闭键盘，关闭键盘有两种方法：

textField.endEditing(true)

textField.resignFirstResponder()撤销自身第一响应者的身份

5.2设定键盘类型:

keyboard --> Number Pad数字键盘

secure 将输入的内容变成不可见的.

5.3【建议】给UITextField加回车事件

将UITextFiel的delegate加到对应的ViewController中

ViewController实现<TextFieldDelegate>

写入

func textFieldShouldReturn(textField: UITextField) -> Bool {

return false/true 返回false时忽略(ps:测试的时候返回false和true似乎结果没有变化)

}

5.4【建议】给UITextField后面加删除按钮

-->Clear Button -->Appears while Editing(编辑时出现删除按钮)

5.5【建议】常用属性

用代码创建文本框时，默认是没有边框的，需要自行制定一下

textField.borderStyle = UITextBorderStyle.RoundedRect 边框设置

textField.secureTextEntry = true 密码不可见

6. **UILabel**标签

6.1 label中文字换行,设置属性lines

7. UIButton

7.1 【规则】代码创建并设置属性

set方法设置按钮正常/高亮状态下的文字

set方法设置按钮正常/高亮状态下的文字颜色

set方法设置按钮背景颜色

set方法设置按钮背景图片

7.2 按钮监听事件

Button.addTarget(self, action: "点击方法", forControlEvents: UIControlEvents.TouchDown)

# 8. UISegmentControl

选项卡控件

作用：同一时刻只能选中一个标签

8.1 基本设置

segmentedControl.insertSegmentWithTitle("选项名字", atIndex: int(标签所在选项卡位置), animated: true)

8.2 点击方法

segmentedControl.addTarget(self, action: "方法名", forControlEvents: UIControlEvents.ValueChanged)

# 9. UITableView

9.1

UITableView两种内置样式:

UITableViewStylePlain

UITableViewStyleGrouped

9.2

【规则】需要dateSource 和delegate

dateSource 数据源方法

func tableView(tableView: UITableView, numberOfRowsInSection section: Int) -> Int 每一组有几行数据

func tableView(tableView: UITableView, cellForRowAtIndexPath indexPath: NSIndexPath) -> UITableViewCell 每一个行的内容

optional func numberOfSectionsInTableView(tableView: UITableView) -> Int 一共多少组数据

optional func tableView(tableView: UITableView, titleForHeaderInSection section: Int) -> String? 页头

optional func tableView(tableView: UITableView, titleForFooterInSection section: Int) -> String? 页脚

【规则】delegate 代理协议

optional func tableView(tableView: UITableView, heightForRowAtIndexPath indexPath: NSIndexPath) -> CGFloat 行高

optional func tableView(tableView: UITableView, didSelectRowAtIndexPath indexPath: NSIndexPath) 点击方法

9.3 刷新表格

调用reloadData方法

9.4 【规则】缓存优化

* iOS设备的内存有限，如果用UITableView显示成千上万条数据，就需要成千上万个UITableViewCell对象的话，那将会耗尽iOS设备的内存。要解决该问题，需要重用UITableViewCell对象
* 重用原理：当滚动列表时，部分UITableViewCell会移出窗口，UITableView会将窗口外的UITableViewCell放入一个对象池中，等待重用。当UITableView要求dataSource返回UITableViewCell时，dataSource会先查看这个对象池，如果池中有未使用的UITableViewCell，dataSource会用新的数据配置这个UITableViewCell，然后返回给UITableView，重新显示到窗口中，从而避免创建新对象