从零开始学iOS7开发系列教程-事务管理软件开发实战-Chapter22

欢迎继续我们的学习。

开发环境：
Xcode 5 +iOS 7

如果你觉得上一章的内容太少，非常不过瘾。那么恭喜你，从这一章开始，将进入本系列教程的高潮，一大波代码正在向你袭来。。。

Don't Panic！开个玩笑而已，每一章的内容我们都会控制在一定的字数内，保证可以在45分钟一节课的时间里面看完。如果哪一章的内容花费的时间超过了45分钟，请发送邮件反馈，我在翻译改编全部完成后会根据大家的反馈补充一些疑问解答，并将内容进行一些适当的调整。

最近"互联网思维"这个词很火。什么是互联网思维？雷布斯先生的回答是：专注极致口碑快。互联网的产品讲究迭代开发，iOS开发教程也是如此。首先是版本上必须与时俱进，其次是内容上必须根据用户反馈进行实时更新和调整。
可能当前这个版本的教程还有很多不尽人意之处，对很多问题的解释也不是那么清楚。但没关系，作为一款有生命的产品，后续肯定是需要不断完善和更新的。
最终的期望是是可以超越用户（小白入门学习者）的期望，让本来晦涩难懂的编程教程为大家所接受。

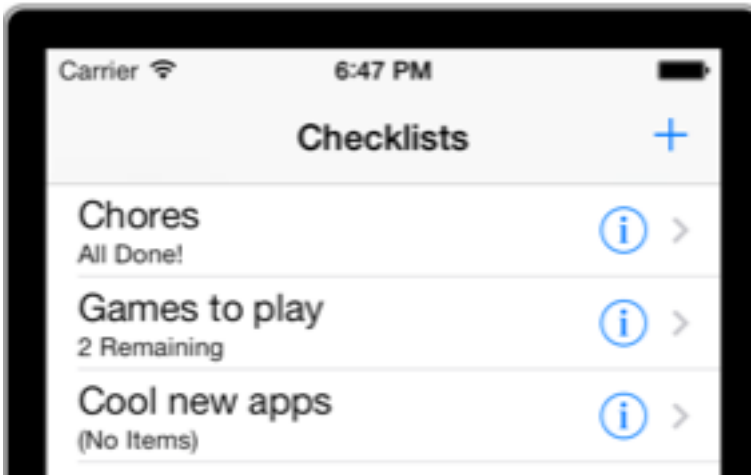
在之前的学习中，我们其实已经成功创建了一个基本可以用的应用了。如果你不嫌丑的话，甚至可以装到GF的手机上显摆下，然后就等着跪搓衣板吧。
用户需要什么样的应用？即便你号称全iOS7扁平化设计，也不能这么糊弄别人吧？要知道，任何一款应用能否吸引用户都是在前三十秒决定的。
当然，正因为此我们需要对icon图标、description描述、screenshot截图这些花点心思。至于如何让用户知道则需要借助各种各样的渠道商和拥有海量用户的社交或工具类应用。

不过，最最重要的还是对用户体验的不懈追求和努力。
在接下来的内容中我们将对应用做一些优化工作，毕竟目标是开发一款可以真正使用的产品。即便可能会被苹果拒绝上架，但起码不要让你的MM看了想吐吧。

好了，废话少说，言归正传。

首先我们要增加一个功能，可以在主界面上显示每个checklist里面还有多少代办事项没有完成。

比如：



如何实现？唯有代码。

打开Xcode,切换到Checklist.h，然后添加一个方法声明：

-(int)countUncheckedItems;

有了这个方法，我们可以询问任何Checklist对象，在它所管辖的范围内有多少ChecklistItem对象还没有自己的勾选标志。该方法返回的是int类型的整数值。

接下来在Checklist.m中添加该方法的实现代码：

首先别忘了在文件顶部添加一行代码导入ChecklistItem.h这个头文件：

#import "ChecklistItem.h"

然后添加方法的实现代码：

```
-(int)countUncheckedItems{

    int count = 0;
    for(ChecklistItem *item in self.items){
        if(!item.checked){
            count+=1;
        }
    }
    return count;

}
```

在上面的方法中，我们使用一个for循环来遍历items数组中的所有ChecklistItem对象。如果item对象的checked属性设置为NO，就可以将本地遍历count的数值加1.当我们遍历完所有的对象后，就可以将最终的count数值返回给方法的调用者。

再次提醒，!操作符就是逻辑非的意思。比如当item.checked是YES的时候，!item.checked的值就是NO。这里可以读作“如果item.checked不是”。

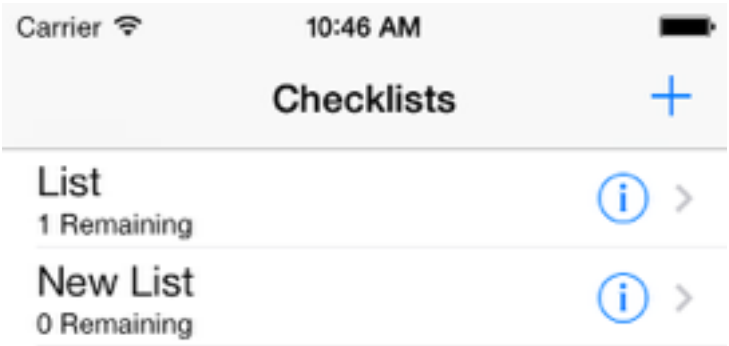
接下来切换到AllListsViewController.m，更改其中的cellForRowAtIndex方法：

```
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath
*)indexPath
{
  static NSString *CellIdentifier = @"Cell";

  UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
  if (cell == nil) {
    cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleSubtitle
reuseIdentifier:CellIdentifier];
```

```
  }

  Checklist *checklist = self.dataModel.lists[indexPath.row];
  cell.textLabel.text = checklist.name;
  cell.accessoryType = UITableViewCellAccessoryDetailDisclosureButton;
  cell.detailTextLabel.text = [NSString stringWithFormat:@"%d Remaining",[checklist
countUncheckedItems]];

  return cell;
}
```

注意在上面的方法中，大部分代码都没变，只有黄色高亮部分发生了变化。首先是UITableViewCell初始化时选择了UITableViewCellStyleSubtitle这种形式，它会在主表情的下面添加一个次级的小一点的标签。我们可以使用cell.detailedTextLabel.text来设置标签的文本内容。

编译运行应用，现在每个checklist都会显示所剩余的代办事项数量了。



问题来了：

如果你试着改动下某个checklist里面的代办事项的勾选状态，会发现主界面所显示的剩余代办事项数量从未发生变化。这是因为我们只是创建了这些table view cell,但却没有更新过他们的标签内容

思考：想想所有可能导致remaining前数字变化的情况。

答案：

1.当用户切换某个代办事项的勾选状态时。如果某个代办事项被勾选，然后count数值就会减1，如果取消了某个代办事项的勾选状态，count数值就会加1。
2.当用户添加了一个新代办事项的时候。初始情况下应该处于未勾选状态，因此每添加一个新事项都会将count数值加1.
3.当用户删除了一个代办事项的时候。此时要做一个判断，如果该代办事项未被勾选，那么久需要将count数值减1，否则就不用管。

记住一点：

无论是编程还是做其它事情，如果你觉得自己无所下手，先停下来在纸上写写画画，看看为了实现这个事情究竟要满足哪些条件。为了实现这些条件又分别需要做哪些事情，直到最终每个小任务都在自己的能力范围之内。

先思考，再动手。
动手完，重新思考。
思考完，继续动手。。。

刚才列出的几点变化都在ChecklistViewController中发生，但这个count数值却显示在AllListsViewController中。这该如何是好?

好吧，我们已经学习了代理的概念，貌似这里可以用代理来解决这个问题。比如我们可以创建一个新的ChecklistViewControllerDelegate协议，让它在以下情况下发送消息：
1.当用户切换了某个代办事项的勾选状态时
2.当用户添加了一个新事项时
3.当用户删除了一个事项时

很好，那么代理对象（AllListsViewController)该如何回应呢？它只需要在这些情况发生的时候在cell的detailTextLabel上显示一个新的文本就好了。

看上去这条路可行。不过世人都说程序猿一向是懒惰的。此法虽善，然则步骤繁琐操作不便，我家那猴子戴上去之后经常头疼脑热。。。

打住打住，这里要介绍一个更简单的方法。因为懒惰的攻城师们永远会选择可以节省代码的方法。

因为懒惰，所以才有创新。
因为懒惰，所有才有进步。
因为懒惰，所有才有编程。
。。。

不过别想跑偏了，懒惰导致创新进步和发明创造往往在一个前提下奏效。那就是减轻体力劳动，而非停止思考。
如果你连想都懒得想，就别在这儿跟风了。
程序猿之懒在于手，而非脑也。

在Xcode中切换到AllListsViewController.m，然后添加一个viewWillAppear方法如下：

```
-(void)viewWillAppear:(BOOL)animated{

    [super viewWillAppear:animated];

    [self.tableView reloadData];

}
```

可别搞错了，这里新添加的是viewWillAppear，不是之前的viewDidAppear。如果你英语足够好的话，应该大致可以明白will和did的区别，一个是将来时，一个是过去时。你可以感受下他们的微妙差别。显然viewWillAppear将在viewDidAppear之前调用。

实际上在iOS的开发过程中，特别是苹果官方提供的API中，有大量的will,Did之类的方法。will方法总是在某件事情发生之前就被触发，而did方法总是在某件事情发生之后才会触发。这就好比杰士邦和避孕药的区别，你懂的。

API（跟我读ei- 屁- 爱)其实是Application Programming Interface（术语党有时候叫它应用编程接口）的缩写。当NB人士言必称iOS API的时候，他们其实值的是苹果官方提供给开发者的和iOS开发相关的所有框架framework，对象，协议和函数。iOS API包含了iOS开发的所有的所有，比如UIKit,Foundation,Core Graphics,甚至还有iOS7中最新的Sprite Kit。你有时候也会听到Facebook API,Google API, 新浪API，腾讯API之类的东东。此时他们的意思是这些公司为了让你给他们的平台开发应用所提供的服务（比如框架，对象，协议和函数，等等）。

这里的viewWillAppear方法要求表视图在显示内容之前先更新它的所有内容，具体来说就是对某个可视的row调用cellForRowAtIndexPath方法。

当我们触碰ChecklistViewController导航栏上的back返回按钮时，AllListsViewContoller界面会再次出现。在此发生前将调用viewWillAppear方法。而因为其中有reloadData方法，可以更新table view cell的所有内容，包括detailTextLabels。

重新加载所有的cell听起来似乎有点任务繁重，不过不要太担心这个。通常情况下All Lists界面中不太可能有过多的行（比如100行就算多的了），因此重新加载的工作其实会很快完成。而且这样可以避免创建另一个代理协议的任务。尽管有时候使用delegate代理是最好的解决方法，但有时候我们只需要重新加载整个表视图就好了。

编译运行应用，看看现在是否可以更新count数值了。

练习：

当所有的代办事项都完成时将标签设置为All Done!

答案：

在AllListsViewController.m中更改cellForRowAtIndexPath方法中的代码如下：

```
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath
*)indexPath
{
  static NSString *CellIdentifier = @"Cell";

  UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
  if (cell == nil) {
    cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleSubtitle
reuseIdentifier:CellIdentifier];
  }

  Checklist *checklist = self.dataModel.lists[indexPath.row];

  cell.textLabel.text = checklist.name;
  cell.accessoryType = UITableViewCellAccessoryDetailDisclosureButton;

    int count = [checklist countUncheckedItems];
    if(count ==0){
        cell.detailTextLabel.text =@"全部搞定收工！";
    }else{
  cell.detailTextLabel.text = [NSString stringWithFormat:@"%d Remaining",[checklist
countUncheckedItems]];
    }

  return cell;
}
```

注意黄色高亮部分就是所做的更改，相信你应该可以看懂了。

再次练习：
当某个checklist中没有代办事项时将标签文本更改为No Items

答案：

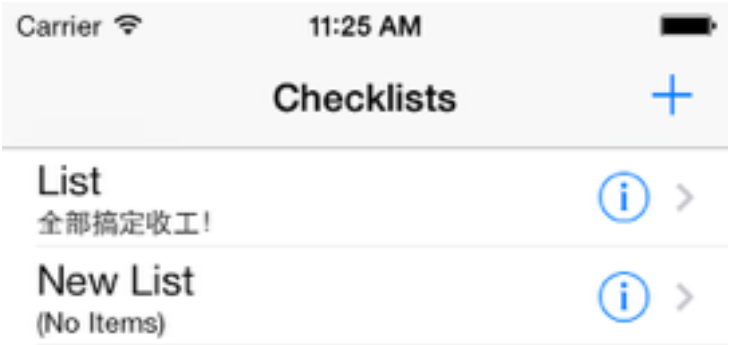
在AllListsViewController.m中更改cellForRowAtIndexPath方法中的代码如下：

```
- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath
*)indexPath
{
  static NSString *CellIdentifier = @"Cell";

  UITableViewCell *cell = [tableView dequeueReusableCellWithIdentifier:CellIdentifier];
  if (cell == nil) {
    cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleSubtitle
reuseIdentifier:CellIdentifier];
  }

  Checklist *checklist = self.dataModel.lists[indexPath.row];

  cell.textLabel.text = checklist.name;
```

```
  cell.accessoryType = UITableViewCellAccessoryDetailDisclosureButton;
    int count = [checklist countUncheckedItems];
    if([checklist.items count]==0){
        cell.detailTextLabel.text = @"(No Items)";
    }else if(count ==0){
        cell.detailTextLabel.text =@"全部搞定收工！";
    }else{
  cell.detailTextLabel.text = [NSString stringWithFormat:@"%d Remaining",[checklist
countUncheckedItems]];
    }
  return cell;
}
```

还是黄色高亮部分，不过这里仅仅查看countUncheckedItems的值还不够。我们还需要通过[checklist.items count]查看某个checklist中的所有代办事项的数量。
编译运行应用，看看这些小细节的变化！
在移动应用（游戏）产品开发的过程中，这种小细节的改进至关重要，它们可以让你的应用和游戏更有趣，更人性化，更具交互性。

别看这小小的文本标签，就可以给某些屌丝和强迫症患者们一些发自内心的满足感和成就感。



好了，为了让大家不至于吃撑了，今天的教程就先到这里了。
老习惯，派发福利时间。

马年快到了，大家希望是马上有钱呢，还是希望马上有房呢?

还是希望马上有美女?

听闻我们的青春偶像周董是希望马上有老婆的，于是八卦搜图，看看传说中的周夫人是神马样子?大家感受下。