列表滑动卡顿问题

1. 问题：项目开发中，免不了和列表打交道，几乎每个页面都含有列表。列表展示的数据也是多种多样的，各种图片文字各种布局。经常伴随着列表的使用出现的问题就是：列表滑动卡顿现象。有时候会羡慕其他平台开发的 app 列表滑动不会卡，倒把卡顿问题归咎于“ios 系统不给力”。其实，最主要的原因还是在我们开发自己代码问题。

2. 分析：为什么会卡顿呢？很简单，主线程同步操作过多，延迟了滑动过程中 cell 的展示。所以，遇到滑动列表卡顿这种问题，应该先从列表的几个方法里查找，看是否存在过多同步操作。主要列表同步操作集中在 cellForRow，heightForRow 等方法。为什么？因为这两个方法可能存在一些控件高度计算，而计算文本 size 是一项耗时的同步操作，假如在 heightForRow 计算一次 cell 高度后，cellForRow 再次对相应文本进行计算。那么列表滑动卡顿是在所难免的。。。还有什么同步操作会明显卡 UI 呢？这可就多了，比如列表图片的展示，假如直接同步请求，那就完蛋；又比如，动态创建控件，在 cellForRow 里反复执行 removeFromSuperView 啊 addSubview 啥的，那么也会造成卡顿，因为控件创建分配内存等操作都是同步操作！

3. 如何解决：这里就没有什么通用的解决方式了，因为同步操作造成主线程阻塞的情况说也说不清，远不止 2 中所提及的情况。主要还是给大家提几个注意事项把，首先，切记 cellForRow 里不要做反复的文本计算，可以把文本计算放在数据请求结果的 json 转 model 数组的过程中，计算好并把高度作为 model 的一个属性，在 cellForRow 和 heightForRow 直接将对应 model 的高度属性值返回即可；再者，关于卡顿现象，找到同步代码后，很经常一种解决方式就是多线程处理，那么是不是简单的将同步操作放到子线程就万事大吉了呢？当然不是，因为回到主线程的目的主要还是为了 cell 的数据展示，那么问题来了，回到主线程对 cell 的 UI 进行数据更新时，此 cell 彼 cell 吗？还是开启子线程时的那个吗？答案是：不一定～具体原因应该不需要展开了吧，直接上两种解决方案。

   (1) 方案一：

```
NSInvocationOperation *operation = [[NSInvocationOperation alloc]initWithTarget:self
                                    selector:@selector(downloadImage:)
                                    object:indexPath];

NSOperationQueue *queue = [[NSOperationQueue alloc]init];
[queue addOperation:operation];
```

```
-(void)downloadImage:(NSIndexPath *)indexPath{
  @autoreleasepool {
    // 2.计算字符串占据空间大小
    NSString *stringData=[NSString stringWithFormat:@"<style>*{font-size:16px;}</style>%@",[_DataArray[indexPath.row] valueForKey:@"message"]];
    NSAttributedString * attrStr = [[NSAttributedString alloc] initWithData:[stringData dataUsingEncoding:NSUnicodeStringEncoding] options:@{
      NSDocumentTypeDocumentAttribute: NSHTMLTextDocumentType } documentAttributes:nil error:nil];
    NSMutableDictionary *dic= [[NSMutableDictionary alloc]init];
    [dic setObject:attrStr forKey:@"attrStr"];
    [dic setObject:indexPath forKey:@"indexPath"];
    [[NSOperationQueue mainQueue] addOperationWithBlock:^{

      OtherTableViewCell *cell = [_tableView cellForRowAtIndexPath:[dic objectForKey:@"indexPath"]];
      cell.msgLabel.attributedText= [dic objectForKey:@"attrStr"];
      //      NSLog(@"%ld %d  %@",index.row,b,cell);
    }];
  }
}
```

(2) 方案二：

```
- (void)loadCenterViewWithHtmlString:(NSString *)htmlStr imageUrl:(NSString *)imageUrl{
  __weak typeof(self) weakSelf = self;
  dispatch_async(dispatch_get_global_queue(DISPATCH_QUEUE_PRIORITY_DEFAULT, 0), ^{
    @autoreleasepool {
      NSIndexPath *indexpath = weakSelf.indexPath;
      NSString *stringData=[NSString stringWithFormat:@"<style>*{font-size:15px;}</style>%@",htmlStr];
      NSAttributedString *str = [[NSAttributedString alloc] initWithData:[stringData dataUsingEncoding:NSUnicodeStringEncoding] options:@{
        NSDocumentTypeDocumentAttribute: NSHTMLTextDocumentType} documentAttributes:nil error:nil];
      dispatch_async(dispatch_get_main_queue(), ^{
        if(indexpath.row == weakSelf.indexPath.row){
          [weakSelf.contentView setAttributedText:str];
        }
      });
    }
  });
```