动态计算UITableViewCell高度

UILabel in UITableViewCell

Auto Layout

- UILabel的属性Lines设为了0表示显示多行。Auto Layout约束一定要建立完完整
- UITableView使用C1.xib中自定义的Cell,那么我们需要向UITableView进行注册

```
UINib *cellNib = [UINib nibWithNibName:@"C1" bundle:nil];
[self.tableView registerNib:cellNib forCellReuseIdentifier:@"C1"];
```

- 数据源方法

```
- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:
(NSInteger)section
{
    // Return the number of rows in the section.
    return self.tableData.count;
}

- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    C1 *cell = [self.tableView dequeueReusableCellWithIdentifier:@"C1"];
    cell.t.text = [self.tableData objectAtIndex:indexPath.row];
    return cell;
}
```

- 需要用到一个新的API systemLayoutSizeFittingSize:来计算UITableViewCell所占空间高度。Cell的高度是在- (CGFloat)tableView:(UITableView)tableView heightForRowAtIndexPath:(NSIndexPath)indexPath这个UITableViewDelegate的方法里面传给UITableView的。
- 这里有一个需要特别注意的问题，也就是效率问题。UITableView是一次性计算完所有Cell的高度，如果有1W个Cell，那么- (CGFloat)tableView:(UITableView)tableView heightForRowAtIndexPath:(NSIndexPath)indexPath就会触发1W次，然后才显示内容。不过在iOS7以后，提供了一个新方法可以避免这1W次调用，它就是- (CGFloat)tableView:(UITableView)tableView estimatedHeightForRowAtIndexPath: (NSIndexPath)indexPath。要求返回一个Cell的估计值，实现了这个方法，那只有显示的Cell才会触发计算高度的protocol. 由于systemLayoutSizeFittingSize需要cell的一个实例才能计算，所以这儿用一个成员变量存一个Cell的实列，这样就不需要每次计算

Cell高度的时候去动态生成一个Cell实例，这样即方便也高效也少用内存，可谓一举三得。

- 声明一个存计算Cell高度的实例变量

```
@property (nonatomic, strong) UITableViewCell *prototypeCell;
```

- 初始化prototypeCell

```
self.prototypeCell  = [self.tableView
dequeueReusableCellWithIdentifier:@"C1"];
```

- 计算Cell高度的实现

```
- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:
(NSIndexPath *)indexPath {
    C1 *cell = (C1 *)self.prototypeCell;
    cell.t.text = [self.tableData objectAtIndex:indexPath.row];
    CGSize size = [cell.contentView
systemLayoutSizeFittingSize:UILayoutFittingCompressedSize];
    NSLog(@"h=%f", size.height + 1);
    return 1  + size.height;
}
```

- UITableViewCell的高度要比它的contentView要高1,也就是它的分隔线的高度

Manual Layout

```
- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    C3 *cell = [self.tableView dequeueReusableCellWithIdentifier:@"C3"];
    cell.t.text = [self.tableData objectAtIndex:indexPath.row];
    [cell.t sizeToFit];
    return cell;
}

- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:
(NSIndexPath *)indexPath {
    C3 *cell = (C3 *)self.prototypeCell;
    NSString *str = [self.tableData objectAtIndex:indexPath.row];
    cell.t.text = str;
    CGSize s = [str calculateSize:CGSizeMake(cell.t.frame.size.width,
FLT_MAX) font:cell.t.font];
    CGFloat defaultHeight = cell.contentView.frame.size.height;
    CGFloat height = s.height > defaultHeight ? s.height : defaultHeight;
    NSLog(@"h=%f", height);
    return 1  + height;
}
// NSString的类别
- (CGSize)calculateSize:(CGSize)size font:(UIFont *)font {
    CGSize expectedLabelSize = CGSizeZero;

    if ([[[UIDevice currentDevice] systemVersion] floatValue] >= 7) {
        NSMutableParagraphStyle *paragraphStyle = [[NSMutableParagraphStyle
alloc] init];
        paragraphStyle.lineBreakMode = NSLineBreakByWordWrapping;
        NSDictionary *attributes = @{NSFontAttributeName:font,
NSParagraphStyleAttributeName:paragraphStyle.copy};

        expectedLabelSize = [self boundingRectWithSize:size
options:NSStringDrawingUsesLineFragmentOrigin attributes:attributes
context:nil].size;
    }
    else {
        expectedLabelSize = [self sizeWithFont:font
                                      constrainedToSize:size

lineBreakMode:NSLineBreakByWordWrapping];
    }

    return CGSizeMake(ceil(expectedLabelSize.width),
ceil(expectedLabelSize.height));
}
```

UITextView in UITableViewCell

Auto Layout

- 计算高度的代码

```
- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:
(NSIndexPath *)indexPath {
    C2 *cell = (C2 *)self.prototypeCell;
    cell.t.text = [self.tableData objectAtIndex:indexPath.row];
    CGSize size = [cell.contentView
systemLayoutSizeFittingSize:UILayoutFittingCompressedSize];
    CGSize textViewSize = [cell.t
sizeThatFits:CGSizeMake(cell.t.frame.size.width, FLT_MAX)];
    CGFloat h = size.height + textViewSize.height;
    h = h > 89 ? h : 89;  //89是图片显示的最低高度， 见xib
    NSLog(@"h=%f", h);
    return 1 + h;
}
```

> 说明：*通过sizeThatFits:计算的UITextView的高度(这是计算UITextView内容全部显示时的方法)，然后加上systemLayoutSizeFittingSize:返回的高度。为什么要这样呢？ 因为UITextView内容的高度不会影响systemLayoutSizeFittingSize计算。*

> *此图中距顶的约束是10, 距底的约束8, 距左边约束是87,距右边的约束是13, 那么systemLayoutSizeFittingSize:返回的CGSize为height等于19, size等于100. 它UITextView的frame是不影响systemLayoutSizeFittingSize:的计算。所以，需要加上textViewSize.height*

Manual Layout

```
- (UITableViewCell *)tableView:(UITableView *)tableView
cellForRowAtIndexPath:(NSIndexPath *)indexPath
{
    C4 *cell = [self.tableView dequeueReusableCellWithIdentifier:@"C4"];
    cell.t.text = [self.tableData objectAtIndex:indexPath.row];
    [cell.t sizeToFit];
    return cell;
}

- (CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:
(NSIndexPath *)indexPath {
    C4 *cell = (C4 *)self.prototypeCell;
    NSString *str = [self.tableData objectAtIndex:indexPath.row];
    cell.t.text = str;
    CGSize s =  [cell.t sizeThatFits:CGSizeMake(cell.t.frame.size.width,
FLT_MAX)];
    CGFloat defaultHeight = cell.contentView.frame.size.height;
    CGFloat height = s.height > defaultHeight ? s.height : defaultHeight;
    return 1  + height;
}
```