一、实现效果

二、实现代码

1.数据模型部分

 YYQQGroupModel.h文件

[](javascript:void(0);)

1 //

2 // YYQQGroupModel.h

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-5-31.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import <Foundation/Foundation.h>

10

11 @interface YYQQGroupModel : NSObject

12 /\*\*

13 \* 名称属性

14 \*/

15 @property(nonatomic,copy)NSString \*name;

16 /\*\*

17 \* 是否在线

18 \*/

19 @property(nonatomic,copy)NSString \*online;

20 /\*\*

21 \* 好友列表

22 \*/

23 @property(nonatomic,strong)NSArray \*friends;

24

25 //记录当前组是否要打开

26 @property(nonatomic,assign,getter = isOpen)BOOL open;

27

28 -(instancetype)initWithDict:(NSDictionary \*)dict;

29 +(instancetype) qqGroupModelWithDict:(NSDictionary \*)dict;

30 @end

[](javascript:void(0);)

 YYQQGroupModel.m文件

[](javascript:void(0);)

1 //

2 // YYQQGroupModel.m

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-5-31.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import "YYQQGroupModel.h"

10 #import "YYFriendsModel.h"

11

12 @implementation YYQQGroupModel

13 -(instancetype)initWithDict:(NSDictionary \*)dict

14 {

15 if (self=[super init]) {

16 //将字典转换为模型

17 [self setValuesForKeysWithDictionary:dict];

18

19 //定义一个数组来保存转换后的模型

20 NSMutableArray \*models=[NSMutableArray arrayWithCapacity:self.friends.count];

21 for (NSDictionary \*dict in self.friends) {

22 YYFriendsModel \*friends=[YYFriendsModel friendsWithDict:dict];

23 [models addObject:friends];

24 }

25 \_friends=[models copy];

26 }

27 return self;

28 }

29

30 +(instancetype)qqGroupModelWithDict:(NSDictionary \*)dict

31 {

32 return [[self alloc]initWithDict:dict];

33 }

34 @end

[](javascript:void(0);)

YYFriendsModel.h文件

[](javascript:void(0);)

1 //

2 // YYFriendsModel.h

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-5-31.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import <Foundation/Foundation.h>

10

11 @interface YYFriendsModel : NSObject

12 /\*\*

13 \* 每个好友的名称

14 \*/

15 @property(nonatomic,copy)NSString \*name;

16 /\*\*

17 \*每个好友的头像

18 \*/

19 @property(nonatomic,copy)NSString \*icon;

20 /\*\*

21 \* 每个好友的个性签名

22 \*/

23 @property(nonatomic,copy)NSString \*intro;

24 /\*\*

25 \* 该好友是否是vip

26 \*/

27 @property(nonatomic,assign,getter = isVip)BOOL vip;

28

29 -(instancetype)initWithDict:(NSDictionary \*)dict;

30 +(instancetype)friendsWithDict:(NSDictionary \*)dict;

31 @end

[](javascript:void(0);)

YYFriendsModel.m文件

[](javascript:void(0);)

1 //

2 // YYFriendsModel.m

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-5-31.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import "YYFriendsModel.h"

10

11 @implementation YYFriendsModel

12 -(instancetype)initWithDict:(NSDictionary \*)dict

13 {

14 if (self=[super init]) {

15 [self setValuesForKeysWithDictionary:dict];

16 }

17 return self;

18 }

19

20 +(instancetype)friendsWithDict:(NSDictionary \*)dict

21 {

22 return [[self alloc]initWithDict:dict];

23 }

24 @end

[](javascript:void(0);)

2.视图部分

YYfriendCell.h文件

[](javascript:void(0);)

1 //

2 // YYfriendCell.h

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-5-31.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import <UIKit/UIKit.h>

10 @class YYFriendsModel;

11 @interface YYfriendCell : UITableViewCell

12

13 @property(nonatomic,strong)YYFriendsModel \*friends;

14

15 +(instancetype)cellWithTableview:(UITableView \*)tableView;

16 @end

[](javascript:void(0);)

YYfriendCell.m文件

[](javascript:void(0);)

1 //

2 // YYfriendCell.m

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-5-31.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import "YYfriendCell.h"

10 #import "YYFriendsModel.h"

11 //私有扩展

12 @interface YYfriendCell()

13

14

15 @end

16 @implementation YYfriendCell

17

18 +(YYfriendCell \*)cellWithTableview:(UITableView \*)tableView

19 {

20 static NSString \*identifier=@"qq";

21 YYfriendCell \*cell=[tableView dequeueReusableCellWithIdentifier:identifier];

22 if (cell==nil) {

23 //这里使用系统自带的样式

24 cell=[[YYfriendCell alloc]initWithStyle:UITableViewCellStyleSubtitle reuseIdentifier:identifier];

25 NSLog(@"创建一个cell");

26 }

27 return cell;

28 }

29

30 -(void)setFriends:(YYFriendsModel \*)friends

31 {

32 \_friends=friends;

33 //1.设置头像

34 self.imageView.image=[UIImage imageNamed:\_friends.icon];

35 //2.设置昵称

36 self.textLabel.text=\_friends.name;

37

38 //3.设置简介

39 self.detailTextLabel.text=\_friends.intro;

40 //判断是否是会员

41

42 /\*\*

43 \* 这里有个注意点，如果不写else设置为黑色，会怎么样？

44 \*/

45 if (\_friends.isVip) {

46 [self.textLabel setTextColor:[UIColor redColor]];

47 }else

48 {

49 [self.textLabel setTextColor:[UIColor blackColor]];

50 }

51 //调整字体的大小

52 self.textLabel.font=[UIFont systemFontOfSize:15.f];

53 self.detailTextLabel.font=[UIFont systemFontOfSize:10.f];

54 }

55 @end

[](javascript:void(0);)

YYHeaderView.h文件

[](javascript:void(0);)

1 //

2 // YYHeaderView.h

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-6-1.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import <UIKit/UIKit.h>

10

11 @class YYQQGroupModel,YYHeaderView;

12

13 //商量一个协议

14 @protocol YYHeaderViewDelegate <NSObject>

15 -(void)headerViewDidClickHeaderView:(YYHeaderView \*)headerView;

16 @end

17

18 @interface YYHeaderView : UITableViewHeaderFooterView

19

20 @property(nonatomic,strong)YYQQGroupModel \*group;

21 //提供一个类方法，创建一个头部视图

22 +(instancetype)headerWithTableView:(UITableView \*)tableView;

23

24

25 //delegate遵守YYHeaderViewDelegate这个协议，可以使用协议中的方法

26 @property(nonatomic,weak)id<YYHeaderViewDelegate> delegate;

27 @end

[](javascript:void(0);)

YYHeaderView.m文件

[](javascript:void(0);)

1 //

2 // YYHeaderView.m

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-6-1.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import "YYHeaderView.h"

10 #import "YYQQGroupModel.h"

11

12 @interface YYHeaderView()

13 @property(nonatomic,strong)UIButton \*btn;

14 @property(nonatomic,strong)UILabel \*lab;

15 @end

16 @implementation YYHeaderView

17

18

19 //创建一个自定义的头部分组视图

20 +(instancetype)headerWithTableView:(UITableView \*)tableView

21 {

22 static NSString \*indentifier=@"header";

23 //先到缓存池中去取数据

24 YYHeaderView \*headerview=[tableView dequeueReusableCellWithIdentifier:indentifier];

25 //如果没有，则自己创建

26 if (headerview==nil) {

27 headerview=[[YYHeaderView alloc]initWithReuseIdentifier:indentifier];

28 }

29 //返回一个头部视图

30 return headerview;

31 }

32

33 #warning 注意在构造方法中为控件设置的frame是无效的

34 -(id)initWithReuseIdentifier:(NSString \*)reuseIdentifier

35 {

36 //初始化父类中的构造方法

37 if (self=[super initWithReuseIdentifier:reuseIdentifier]) {

38 //创建一个按钮

39 UIButton \*btn=[UIButton buttonWithType:UIButtonTypeCustom];

40 //设置按钮的属性

41 //设置普通状态下按钮的背景图片

42 [btn setBackgroundImage:[UIImage imageNamed:@"buddy\_header\_bg"] forState:UIControlStateNormal];

43 //设置高亮状态下按钮的背景图片

44 [btn setBackgroundImage:[UIImage imageNamed:@"buddy\_header\_bg\_highlighted"] forState:UIControlStateHighlighted];

45

46 //设置按钮上的小三角图片

47 [btn setImage:[UIImage imageNamed:@"buddy\_header\_arrow"] forState:UIControlStateNormal];

48 //设置按钮上信息的对其方式为左对齐

49 btn.contentHorizontalAlignment=UIControlContentHorizontalAlignmentLeft;

50 //设置小三角图片的内边距

51 btn.contentEdgeInsets=UIEdgeInsetsMake(0, 20, 0, 0);

52 //设置按钮上文字距离小三角图片的距离

53 btn.titleEdgeInsets=UIEdgeInsetsMake(0, 20, 0, 0);

54 //设置按钮上分组标题的文本颜色（默认是白色）

55 //[btn setTintColor:[UIColor blackColor]];

56 [btn setTitleColor:[UIColor blackColor] forState:UIControlStateNormal];

57 //添加按钮的点击事件

58 [btn addTarget:self action:@selector(btnOnclick:) forControlEvents:UIControlEventTouchUpInside];

59

60 // 设置btn中的图片不填充整个imageview

61 btn.imageView.contentMode = UIViewContentModeCenter;

62 // 超出范围的图片不要剪切

63 // btn.imageView.clipsToBounds = NO;

64 btn.imageView.layer.masksToBounds = NO;

65

66 //把按钮添加到视图

67 [self addSubview:btn];

68 self.btn=btn;

69

70 //创建一个lab

71 UILabel \*lab=[[UILabel alloc]init];

72 //设置在线人数的对齐方式为右对齐

73 lab.textAlignment=NSTextAlignmentRight;

74 //设置在线人数的文本颜色为灰色

75 lab.textColor=[UIColor grayColor];

76 [self addSubview:lab];

77 self.lab=lab;

78 }

79 return self;

80 }

81

82

83 -(void)btnOnclick:(UIButton \*)btn

84 {

85 NSLog(@"按钮被点击了");

86 //修改模型的isopen属性

87 //1.修改模型数据

88 self.group.open=!self.group.isOpen;

89 //2.刷新表格

90 //（刷新表格的功能由控制器完成，在这里可以设置一个代理），当按钮被点击的时候，就通知代理对表格进行刷新

91 //通知代理

92 if ([self.delegate respondsToSelector:@selector(headerViewDidClickHeaderView:)]) {

93 [self.delegate headerViewDidClickHeaderView:self];

94 }

95 }

96

97 //当控件的frame值改变时，会自动调用该方法，故可以在该方法中设置控件的frame;

98 -(void)layoutSubviews

99 {

100 #warning 一定不要忘记调用父类的方法

101 [super layoutSubviews];

102 //设置按钮的frame和头部视图一样大小

103 self.btn.frame=self.bounds;

104

105 //设置lab的frame

106 CGFloat padding=20;

107 CGFloat labW=50;

108 CGFloat labH=self.frame.size.height;

109 CGFloat labY=0;

110 CGFloat labX=self.frame.size.width-padding-labW;

111 self.lab.frame=CGRectMake(labX, labY, labW, labH);

112 }

113

114 #pragma mark - 当一个控件被添加到其它视图上的时候会调用以下方法

115 // 已经被添加到父视图上的时候会调用

116 - (void)didMoveToSuperview

117 {

118 NSLog(@"已经添加到视图了");

119 // 在这个方法中就快要拿到最新的被添加到tableview上的头部视图修改它的图片

120 if (self.group.isOpen) {

121 //让小三角图片向下旋转

122 self.btn.imageView.transform = CGAffineTransformMakeRotation(M\_PI\_2);

123 }

124 }

125

126 // 即将被添加到父视图上的时候会调用

127 - (void)willMoveToSuperview:(UIView \*)newSuperview

128 {

129 NSLog(@"将要添加到视图了");

130 }

131

132

133 //重写get方法，设置数据

134 -(void)setGroup:(YYQQGroupModel \*)group

135 {

136 \_group=group;

137 //设置分组标题

138

139 //self.btn.titleLabel.text=\_group.name;

140 #warning 请注意在设置按钮的文本时，一定要设置按钮的状态，像上面这样设置不会显示

141 [self.btn setTitle:\_group.name forState:UIControlStateNormal];

142 NSLog(@"%@",self.btn.titleLabel.text);

143 //设置在线人数

144 self.lab.text=[NSString stringWithFormat:@"%@/%d",\_group.online,\_group.friends.count];

145 }

146

147 @end

[](javascript:void(0);)

3.控制器部分

YYViewController.h文件

[](javascript:void(0);)

1 //

2 // YYViewController.h

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-5-31.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import <UIKit/UIKit.h>

10

11 @interface YYViewController : UITableViewController

12

13 @end

[](javascript:void(0);)

YYViewController.m文件

[](javascript:void(0);)

1 //

2 // YYViewController.m

3 // 02-QQ好友列表（基本数据的加载）

4 //

5 // Created by apple on 14-5-31.

6 // Copyright (c) 2014年 itcase. All rights reserved.

7 //

8

9 #import "YYViewController.h"

10 #import "YYQQGroupModel.h"

11 #import "YYfriendCell.h"

12 #import "YYFriendsModel.h"

13 #import "YYHeaderView.h"

14

15 @interface YYViewController ()<YYHeaderViewDelegate>

16 /\*\*

17 \* 用来保存所有的分组数据

18 \*/

19 @property(nonatomic,strong)NSArray \*groupFriends;

20 @end

21

22 @implementation YYViewController

23 #pragma mark-懒加载

24 //1.先拿到数据，实现懒加载

25 -(NSArray \*)groupFriends

26 {

27 if (\_groupFriends==nil) {

28 NSString \*fullpath=[[NSBundle mainBundle]pathForResource:@"friends.plist" ofType:nil];

29 NSArray \*arrayM=[NSArray arrayWithContentsOfFile:fullpath];

30

31 NSMutableArray \*models=[NSMutableArray arrayWithCapacity:arrayM.count];

32 for (NSDictionary \*dict in arrayM) {

33 YYQQGroupModel \*group=[YYQQGroupModel qqGroupModelWithDict:dict];

34 [models addObject:group];

35 }

36 \_groupFriends=[models copy];

37 }

38 return \_groupFriends;

39 }

40

41 - (void)viewDidLoad

42 {

43 [super viewDidLoad];

44 self.tableView.sectionHeaderHeight = 100;

45

46 }

47

48 #pragma mark- 设置数据源

49 //返回多少组

50 -(NSInteger)numberOfSectionsInTableView:(UITableView \*)tableView

51 {

52 return self.groupFriends.count;

53 }

54 //每组返回多少行

55 -(NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section

56 {

57 // //取出对应的组模型

58 YYQQGroupModel \*group=self.groupFriends[section];

59 // //返回对应组中的好友数

60 // return group.friends.count;

61

62 //在这里进行判断，如果该组收拢，那就返回0行，如果该组打开，就返回实际的行数

63 // if (group.isOpen) {

64 // return group.friends.count;

65 // }else

66 // {

67 // return 0;

68 // }

69

70 if (group.isOpen) {

71 // 代表要展开

72 return group.friends.count;

73 }else

74 {

75 // 代表要合拢

76 return 0;

77 }

78 }

79 //每组每行的内容

80 -(UITableViewCell \*)tableView:(UITableView \*)tableView cellForRowAtIndexPath:(NSIndexPath \*)indexPath

81 {

82 //1.创建cell

83 YYfriendCell \*cell=[YYfriendCell cellWithTableview:tableView];

84

85 //2.设置cell

86 YYQQGroupModel \*group=self.groupFriends[indexPath.section];

87 YYFriendsModel \*friends=group.friends[indexPath.row];

88 cell.friends=friends;

89 //3.返回一个cell

90 return cell;

91 }

92

93

94 #pragma mark - 代理方法

95 // 当一个分组标题进入视野的时候就会调用该方法

96 - (UIView \*)tableView:(UITableView \*)tableView viewForHeaderInSection:(NSInteger)section

97 {

98 // // 1.创建头部视图

99 // UIView \*view = [[UIView alloc] init];

100 // view.backgroundColor = [UIColor grayColor];

101 // // 2.返回头部视图

102 // return view;

103

104 //创建自定义的头部视图

105 YYHeaderView \*headerview=[YYHeaderView headerWithTableView:tableView];

106

107 //设置当前控制器为代理

108 headerview.delegate=self;

109 //设置头部视图的数据

110 YYQQGroupModel \*groupmodel=self.groupFriends[section];

111 headerview.group=groupmodel;

112 //返回头部视图

113 return headerview;

114 }

115

116

117 #pragma mark - YYHeaderViewDelegate

118 -(void)headerViewDidClickHeaderView:(YYHeaderView \*)headerView

119 {

120 //重新调用数据源的方法刷新数据

121 [self.tableView reloadData];

122 }

123 //设置分组头部标题的高度

124 -(CGFloat)tableView:(UITableView \*)tableView heightForHeaderInSection:(NSInteger)section

125 {

126 return 30;

127 }

128

129 #pragma mark 隐藏状态栏

130 -(BOOL)prefersStatusBarHidden

131 {

132 return YES;

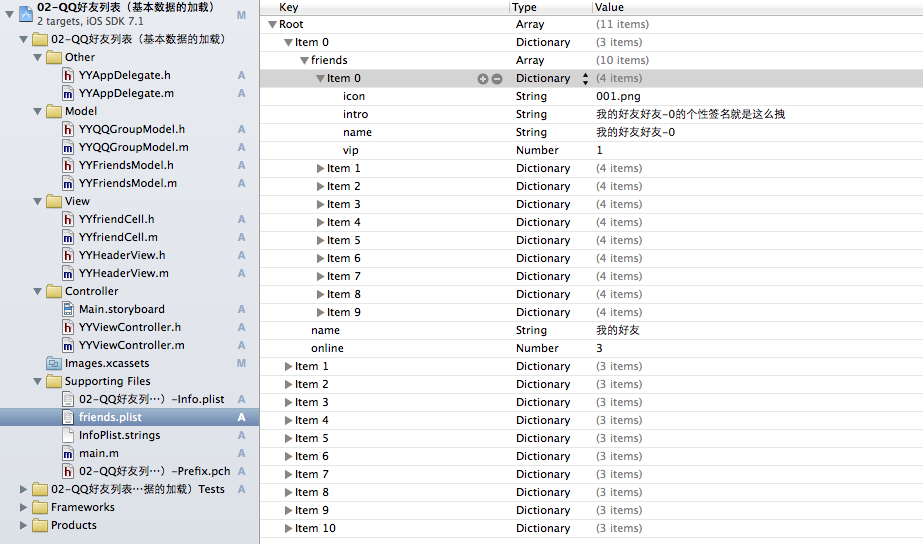
133 }

134 @end

[](javascript:void(0);)

三、代码说明

1.项目文件结构



2.注意点

（1）调整字体的大小：    self.textLabel.font=[UIFont systemFontOfSize:15.f];

（2）-(void)layoutSubviews方法。该方法在控件的frame被改变的时候就会调用，这个方法一般用于调整子控件的位置，注意一定要调用[super layoutSubviews];

（3）但凡在init方法中获取到的frame都是0；

（4）如果控件不显示，有以下一些排错方法

a.frame为空（没有设置frame）

b.hidden是否为YES

c.alpha<=0.1(透明度)

d.没有添加到父控件中

e.查看父控件以上几点

（5）请注意在设置按钮的文本时，一定要设置按钮的状态

正确：[self.btn setTitle:\_group.name forState:UIControlStateNormal];

错误： self.btn.titleLabel.text=\_group.name;

（6）调用构造方法时，一定要先初始化父类的方法，先判断，再进行自己属性的初始化

self=[super initWithReuseIdentifier:reuseIdentifier]

if(self)

{

……

}

（7）当一个控件被添加到其它视图上的时候会调用以下方法

1） 已经被添加到父视图上的时候会调用- (void)didMoveToSuperview

2） 即将被添加到父视图上的时候会调用- (void)willMoveToSuperview:(UIView \*)newSuperview

（8）图片填充知识

   1）设置btn中的图片不填充整个imageview btn.imageView.contentMode = UIViewContentModeCenter;

   2）超出范围的图片不要剪切

  //btn.imageView.clipsToBounds = NO;

  btn.imageView.layer.masksToBounds = NO;

四、补充(代理)

设置代理的几个步骤

（1）如果一个视图中的某个按钮被点击了，这个时候需要去主控制器中刷新数据。有一种做法是，让这个视图拥有控制器这个属性，然后当按钮被点击的时候去利用该属性去做刷新数据的操作。另一种做法是把控制器设置为这个视图的代理，当视图中的某个按钮被点击的时候，通知它的代理（主控制器）去干刷新数据这件事。

（2）要成为代理是由条件的，有以下几个步骤

1）.双方约定一个协议（代理协议，注意命名规范），在视图中自定义一个协议，协议中提供一个方法。

@protocol YYHeaderViewDelegate <NSObject>

-(void)headerViewDidClickHeaderView:(YYHeaderView \*)headerView;

@end

2）.在视图中添加一个id类型的属性变量，任何人只要遵守了约定协议的都可以成为它的代理。

//delegate遵守YYHeaderViewDelegate这个协议，可以使用协议中的方法

@property(nonatomic,weak)id<YYHeaderViewDelegate> delegate;

3）.在控制器中，遵守自定义的代理协议，就可以使用代理提供的方法，在这个方法中对数据进行刷新。

@interface YYViewController ()<YYHeaderViewDelegate>

-(void)headerViewDidClickHeaderView:(YYHeaderView \*)headerView

{

    [self.tableView reloadData];

}

4）.把控制器设置作为按钮点击事件的代理。

headerview.delegate=self;