**iOS开发UI篇—UITableview控件简单介绍**

**一、基本介绍**

在众多移动应⽤用中,能看到各式各样的表格数据 。

在iOS中,要实现表格数据展示,最常用的做法就是使用UITableView，UITableView继承自UIScrollView,因此支持垂直滚动,⽽且性能极佳 。

UITableview有分组和不分组两种样式，可以在storyboard或者是用代码设置。



二、数据展示

UITableView需要⼀一个数据源(dataSource)来显示数据 UITableView会向数据源查询一共有多少行数据以及每⼀行显示什么数据等

没有设置数据源的UITableView只是个空壳

凡是遵守UITableViewDataSource协议的OC对象,都可以是UITableView的数据源

展示数据的过程：

（1）调用数据源的下面⽅法得知⼀一共有多少组数据 - (NSInteger)numberOfSectionsInTableView:(UITableView \*)tableView;

（2）调用数据源的下面⽅法得知每一组有多少行数据 - (NSInteger)tableView:(UITableView \*)tableView numberOfRowsInSection:(NSInteger)section;

（3）调⽤数据源的下⾯⽅法得知每⼀⾏显示什么内容

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

三、代码示例

（1）能基本展示的“垃圾”代码

[](javascript:void(0);)

1 #import "NJViewController.h"

2

3 @interface NJViewController ()<UITableViewDataSource>

4 @property (weak, nonatomic) IBOutlet UITableView \*tableView;

5

6 @end

7

8 @implementation NJViewController

9

10 - (void)viewDidLoad

11 {

12 [super viewDidLoad];

13 // 设置tableView的数据源

14 self.tableView.dataSource = self;

15

16 }

17

18 #pragma mark - UITableViewDataSource

19 /\*\*

20 \* 1.告诉tableview一共有多少组

21 \*/

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

23 {

24 NSLog(@"numberOfSectionsInTableView");

25 return 2;

26 }

27 /\*\*

28 \* 2.第section组有多少行

29 \*/

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

31 {

32 NSLog(@"numberOfRowsInSection %d", section);

33 if (0 == section) {

34 // 第0组有多少行

35 return 2;

36 }else

37 {

38 // 第1组有多少行

39 return 3;

40 }

41 }

42 /\*\*

43 \* 3.告知系统每一行显示什么内容

44 \*/

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

46 {

47 NSLog(@"cellForRowAtIndexPath %d %d", indexPath.section, indexPath.row);

48 // indexPath.section; // 第几组

49 // indexPath.row; // 第几行

50 // 1.创建cell

51 UITableViewCell \*cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault reuseIdentifier:nil];

52

53 // 2.设置数据

54 // cell.textLabel.text = @"汽车";

55 // 判断是第几组的第几行

56 if (0 == indexPath.section) { // 第0组

57 if (0 == indexPath.row) // 第0组第0行

58 {

59 cell.textLabel.text = @"奥迪";

60 }else if (1 == indexPath.row) // 第0组第1行

61 {

62 cell.textLabel.text = @"宝马";

63 }

64

65 }else if (1 == indexPath.section) // 第1组

66 {

67 if (0 == indexPath.row) { // 第0组第0行

68 cell.textLabel.text = @"本田";

69 }else if (1 == indexPath.row) // 第0组第1行

70 {

71 cell.textLabel.text = @"丰田";

72 }else if (2 == indexPath.row) // 第0组第2行

73 {

74 cell.textLabel.text = @"马自达";

75 }

76 }

77

78 // 3.返回要显示的控件

79 return cell;

80

81 }

82 /\*\*

83 \* 第section组头部显示什么标题

84 \*

85 \*/

86 - (NSString \*)tableView:(UITableView \*)tableView titleForHeaderInSection:(NSInteger)section

87 {

88 // return @"标题";

89 if (0 == section) {

90 return @"德系品牌";

91 }else

92 {

93 return @"日韩品牌";

94 }

95 }

96 /\*\*

97 \* 第section组底部显示什么标题

98 \*

99 \*/

100 - (NSString \*)tableView:(UITableView \*)tableView titleForFooterInSection:(NSInteger)section

101 {

102 if (0 == section) {

103 return @"高端大气上档次";

104 }else

105 {

106 return @"还不错";

107 }

108 }

109 @end

[](javascript:void(0);)

实现效果：



（2）让代码的数据独立

新建一个模型

[](javascript:void(0);)

1 #import <Foundation/Foundation.h>

2

3 @interface NJCarGroup : NSObject

4 /\*\*

5 \* 标题

6 \*/

7 @property (nonatomic, copy) NSString \*title;

8 /\*\*

9 \* 描述

10 \*/

11 @property (nonatomic, copy) NSString \*desc;

12 /\*\*

13 \* 当前组所有行的数据

14 \*/

15 @property (nonatomic, strong) NSArray \*cars;

16

17 @end

[](javascript:void(0);)

[](javascript:void(0);)

1 #import "NJViewController.h"

2 #import "NJCarGroup.h"

3

4 @interface NJViewController ()<UITableViewDataSource>

5 @property (weak, nonatomic) IBOutlet UITableView \*tableView;

6 /\*\*

7 \* 保存所有组的数据(其中每一元素都是一个模型对象)

8 \*/

9 @property (nonatomic, strong) NSArray \*carGroups;

10 @end

11

12

13 @implementation NJViewController

14

15

16 #pragma mark - 懒加载

17 - (NSArray \*)carGroups

18 {

19 if (\_carGroups == nil) {

20 // 1.创建模型

21 NJCarGroup \*cg1 = [[NJCarGroup alloc] init];

22 cg1.title = @"德系品牌";

23 cg1.desc = @"高端大气上档次";

24 cg1.cars = @[@"奥迪", @"宝马"];

25

26 NJCarGroup \*cg2 = [[NJCarGroup alloc] init];

27 cg2.title = @"日韩品牌";

28 cg2.desc = @"还不错";

29 cg2.cars = @[@"本田", @"丰田", @"小田田"];

30

31 NJCarGroup \*cg3 = [[NJCarGroup alloc] init];

32 cg3.title = @"欧美品牌";

33 cg3.desc = @"价格昂贵";

34 cg3.cars = @[@"劳斯莱斯", @"布加迪", @"小米"];

35 // 2.将模型添加到数组中

36 \_carGroups = @[cg1, cg2, cg3];

37 }

38 // 3.返回数组

39 return \_carGroups;

40 }

41

42 - (void)viewDidLoad

43 {

44 [super viewDidLoad];

45 // 设置tableView的数据源

46 self.tableView.dataSource = self;

47

48 }

49

50 #pragma mark - UITableViewDataSource

51 /\*\*

52 \* 1.告诉tableview一共有多少组

53 \*/

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

55 {

56 NSLog(@"numberOfSectionsInTableView");

57 return self.carGroups.count;

58 }

59 /\*\*

60 \* 2.第section组有多少行

61 \*/

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

63 {

64 NSLog(@"numberOfRowsInSection %d", section);

65 // 1.取出对应的组模型

66 NJCarGroup \*g = self.carGroups[section];

67 // 2.返回对应组的行数

68 return g.cars.count;

69 }

70 /\*\*

71 \* 3.告知系统每一行显示什么内容

72 \*/

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

74 {

75 NSLog(@"cellForRowAtIndexPath %d %d", indexPath.section, indexPath.row);

76 // indexPath.section; // 第几组

77 // indexPath.row; // 第几行

78 // 1.创建cell

79 UITableViewCell \*cell = [[UITableViewCell alloc] initWithStyle:UITableViewCellStyleDefault reuseIdentifier:nil];

80

81 // 2.设置数据

82 // cell.textLabel.text = @"嗨喽";

83 // 2.1取出对应组的模型

84 NJCarGroup \*g = self.carGroups[indexPath.section];

85 // 2.2取出对应行的数据

86 NSString \*name = g.cars[indexPath.row];

87 // 2.3设置cell要显示的数据

88 cell.textLabel.text = name;

89 // 3.返回要显示的控件

90 return cell;

91

92 }

93 /\*\*

94 \* 第section组头部显示什么标题

95 \*

96 \*/

97 - (NSString \*)tableView:(UITableView \*)tableView titleForHeaderInSection:(NSInteger)section

98 {

99 // return @"标题";

100 // 1.取出对应的组模型

101 NJCarGroup \*g = self.carGroups[section];

102 return g.title;

103 }

104 /\*\*

105 \* 第section组底部显示什么标题

106 \*

107 \*/

108 - (NSString \*)tableView:(UITableView \*)tableView titleForFooterInSection:(NSInteger)section

109 {

110 // return @"标题";

111 // 1.取出对应的组模型

112 NJCarGroup \*g = self.carGroups[section];

113 return g.desc;

114 }

115 @end

[](javascript:void(0);)

实现效果：



提示：请自行体会把数据独立出来单独处理，作为数据模型的好处。另外，把什么抽成一个模型，一定要弄清楚。

四、补充点

contentView下默认有3个⼦视图

第2个是UILabel(通过UITableViewCell的textLabel和detailTextLabel属性访问)

第3个是UIImageView(通过UITableViewCell的imageView属性访问)

 UITableViewCell还有⼀个UITableViewCellStyle属性,⽤于决定使用contentView的哪些子视图,以及这些子视图在contentView中的位置

