

07-自定义城市键盘

Make by:弓_虽_子

```
#import <UIKit/UIKit.h>
```

注意:这里继承的是UITextField

```
@interface CityTextField : UITextField
```

```
@end
```

```
#import "CityTextField.h"
```

省模型

```
#import "ProvincesItem.h"
```

```
@interface CityTextField() <UIPickerViewDataSource, UIPickerViewDelegate>
```

省份数组.里面装的都是ProvincesItem模型.

```
@property (nonatomic, strong) NSMutableArray *provincesdataArray;
```

记录当前选中的是哪一个省份的角标.

```
@property (nonatomic, assign) NSInteger curProIndex;
```

```
@end
```

加载数据

数据内容对应的模型图:

每个字典对应一个模型

每一个省是一个模型

Key	Type	Value
Root	Array	(24 items)
Item 0	Dictionary	(2 items)
cities	Array	(10 items)
name	String	福建
Item 1	Dictionary	(2 items)
cities	Array	(23 items)
Item 0	String	济南
Item 1	String	滨州
Item 2	String	德州
Item 3	String	东营
Item 4	String	肥城
Item 5	String	海阳
Item 6	String	菏泽
Item 7	String	济宁
Item 8	String	莱芜
Item 9	String	莱阳
Item 10	String	聊城
Item 11	String	临沂
Item 12	String	平度
Item 13	String	青岛
Item 14	String	青州
Item 15	String	日照
Item 16	String	泰安
Item 17	String	威海
Item 18	String	潍坊
Item 19	String	烟台
Item 20	String	枣庄
Item 21	String	章丘
Item 22	String	淄博
name	String	山东
Item 2	Dictionary	(2 items)
Item 3	Dictionary	(2 items)
Item 4	Dictionary	(2 items)



普通文件

数组当中保存的都是字典. 我们看到字典就会它转成模型.

所以新建了一个 ProvincesItem模型, ProvincesItem每一个属性,都对应着字典当中的key值。

```
8
9 #import <Foundation/Foundation.h>
10
11 @interface ProvincesItem : NSObject
12
13 //每个省下面的城市
14 @property (nonatomic, strong) NSArray *cities;
15 //省的名称
16 @property (nonatomic, strong) NSString *name;
17 //将一个字典转成模型
18 + (instancetype)provincesItemWithDict:(NSDictionary *)dict;
19
20
21 @end
22
23
24 #import "ProvincesItem.h"
25
26 @implementation ProvincesItem
27
28 +(instancetype)provincesItemWithDict:(NSDictionary *)dict{
29     //创建模型
30     ProvincesItem *item = [[self alloc] init];
31     //通过KVC将字典中的key,给item对象当中的属性进行赋值。
32     [item setValuesForKeysWithDictionary:dict];
33     //返回一个赋值好的模型。
34     return item;
35 }
36
37
38 @end
```

懒加载省份数据

```
-(NSMutableArray *)provincesdataArray{
    if (_provincesdataArray == nil) {
        //创建数组
        _provincesdataArray = [NSMutableArray array];
        //获取plist文件的路径
        NSString *filePath = [[NSBundle mainBundle]
pathForResource:@"provinces.plist" ofType:nil];
        //从plist文件当中加载数据
        NSArray *tempArray = [NSArray arrayWithContentsOfFile:filePath];
        //遍历数组当中的每一个元素.每一个元素都是一个字典,把字典转成模型
        for (NSDictionary *dict in tempArray) {
            //创建省份模型对象.快速的将字典转成模型
            ProvincesItem *item = [ProvincesItem provincesItemWithDict:dict];
            //把转好的对象保存到数组当中。
            [_provincesdataArray addObject:item];
        }
    }
    return _provincesdataArray;
}

-(void)awakeFromNib{
    //初始化。
    [self setUp];
}
```

```

- (instancetype)initWithFrame:(CGRect)frame{
    if (self = [super initWithFrame:frame]) {
        初始化.
        [self setUp];
    }
    return self;
}

```

初始化.

```

- (void)setUp{
    创建UIPickerView
    UIPickerView *pickV = [[UIPickerView alloc] init];
    设置代理
    pickV.delegate = self;
    设置数据源.
    pickV.dataSource = self;
    设置弹出键盘是一个UIPickerView
    self.inputView = pickV;
}

```

实现代理协议方法

总共多少组

```

- (NSInteger)numberOfComponentsInPickerView:(UIPickerView *)pickerView{
    总共两组
    return 2;
}

```

每一组有多少行

```

- (NSInteger)pickerView:(UIPickerView *)pickerView numberOfRowsInComponent:
(NSInteger)component{
    第一组展示多少行,有省份数组解定,数组当中有多少个省份模型,第一组就有多少行.
    if (component == 0) {
        return self.provincesdataArray.count;
    }else{
        第二组展示多少行,有当前选中的省份决定,当前选中哪个省份.哪个省份下面的城市数组决定.
        所以要先获取当前先中的是哪一个省.
        ProvincesItem *proItem = self.provincesdataArray[self.curProIndex];
        返回当前省份下城市的个数
        return proItem.cities.count;
    }
}

```

返回每一行展示的标题

```

- (NSString *)pickerView:(UIPickerView *)pickerView titleForRow:
(NSInteger)row forComponent:(NSInteger)component{

    如果是第0列,就是省份,返回省份的名称
    if (component == 0) {
        ProvincesItem *proItem = self.provincesdataArray[row];
        return proItem.name;
    }else{
        先取出选中的省份.
        ProvincesItem *proItem = self.provincesdataArray[self.curProIndex];
        返回先中省份下,每一个城市.
        return proItem.cities[row];
    }
}

```

选中某一列的某一行时会调用这个方法

```

- (void)pickerView:(UIPickerView *)pickerView didSelectRow:(NSInteger)row

```

```

inComponent:(NSInteger)component{
    当是第0列的时候.(省份列)
    if(component == 0){
        记录当前选中的省份的角标
        self.curProIndex = row;
        刷新列表.
        [pickerView reloadData];
        让第1列的第0行成为选中状态.(让城市列每次刷新时,都选中第一个)
        [pickerView selectRow:0 inComponent:1 animated:YES];
    }
    把选中的省份,城市显示到文本框当中.
    获取选中的省.
    ProvincesItem *proItem = self.provincesdataArray[self.curProIndex];
    获取省份的名称
    NSString *province = proItem.name;
    获取第一列选中的行号.(城市列)
    NSInteger seleRow = [pickerView selectedRowInComponent:1];
    获取当前选中省下面选中的城市名称.
    NSString *city = proItem.cities[seleRow];
    显示数据
    self.text = [NSString stringWithFormat:@"%s %s",province,city];
}

```