# UITextField属性

## enablesReturnKeyAutomatically

默认为No,如果设置为Yes,文本框中没有输入任何字符的话，右下角的返回按钮是disabled的。

## 1.borderStyle

设置边框样式，只有设置了才会显示边框样式

  text.borderStyle = UITextBorderStyleRoundedRect;

　typedef enum {

    UITextBorderStyleNone,

    UITextBorderStyleLine,

    UITextBorderStyleBezel,

    UITextBorderStyleRoundedRect

  } UITextBorderStyle;

## 2.backgroundColor

设置输入框的背景颜色，此时设置为白色 如果使用了自定义的背景图片边框会被忽略掉

　 text.backgroundColor = [UIColor whiteColor];

## 3.background

设置背景

  text.background = [UIImage imageNamed:@"dd.png"];//UITextField 的背景，注意只有UITextBorderStyleNone的时候改属性有效

设置enable为no时，textfield的背景

  text.disabledBackground = [UIImage imageNamed:@"cc.png"];

## 4.placeholder

当输入框没有内容时， 提示内容为password

  text.placeholder = @"password";

## 5.font

设置输入框内容的字体样式和大小

  text.font = [UIFont fontWithName:@"Arial" size:20.0f];

## 6. textColor

设置字体颜色

  text.textColor = [UIColor redColor];

## 7. clearButtonMode

输入框中是否有个叉号，在什么时候显示，用于一次性删除输入框中的内容

  text.clearButtonMode = UITextFieldViewModeAlways;

typedef enum {

    UITextFieldViewModeNever,　重不出现

    UITextFieldViewModeWhileEditing, 编辑时出现

    UITextFieldViewModeUnlessEditing,　除了编辑外都出现

    UITextFieldViewModeAlways 　一直出现

} UITextFieldViewMode;

## 8. text

输入框中一开始就有的文字

  text.text = @"一开始就在输入框的文字";

## 9. secureTextEntry

每输入一个字符就变成点 用来输入密码时，设置这个属性。

  text.secureTextEntry = YES;

## 10. autocorrectionType

是否纠错

  text.autocorrectionType = UITextAutocorrectionTypeNo;

typedef enum {

    UITextAutocorrectionTypeDefault, 默认

    UITextAutocorrectionTypeNo, 　不自动纠错

    UITextAutocorrectionTypeYes,　自动纠错

} UITextAutocorrectionType;

## 11. clearsOnBeginEditing

再次编辑就清空

  text.clearsOnBeginEditing = YES;

## 12. textAlignment

内容对齐方式

  text.textAlignment = UITextAlignmentLeft;

## 13. contentVerticalAlignment

内容的垂直对齐方式  UITextField继承自UIControl,此类中有一个属性contentVerticalAlignment

  text.contentVerticalAlignment = UIControlContentVerticalAlignmentCenter;

## 14. adjustsFontSizeToFitWidth

设置为YES时文本会自动缩小以适应文本窗口大小.默认是保持原来大小,而让长文本滚动

  textFied.adjustsFontSizeToFitWidth = YES;

//设置自动缩小显示的最小字体大小

  text.minimumFontSize = 20;

## 15. keyboardType

设置键盘的样式

  text.keyboardType = UIKeyboardTypeNumberPad;

typedef enum {

    UIKeyboardTypeDefault,     　默认键盘，支持所有字符

    UIKeyboardTypeASCIICapable,　支持ASCII的默认键盘

    UIKeyboardTypeNumbersAndPunctuation,　标准电话键盘，支持＋＊＃字符

    UIKeyboardTypeURL,            URL键盘，支持.com按钮 只支持URL字符

UIKeyboardTypeNumberPad,            　数字键盘

UIKeyboardTypePhonePad,　 　电话键盘

    UIKeyboardTypeNamePhonePad, 　电话键盘，也支持输入人名

UIKeyboardTypeEmailAddress, 　用于输入电子 邮件地址的键盘

UIKeyboardTypeDecimalPad,   　数字键盘 有数字和小数点

    UIKeyboardTypeTwitter,      　优化的键盘，方便输入@、#字符

    UIKeyboardTypeAlphabet = UIKeyboardTypeASCIICapable,

} UIKeyboardType;

## 16.autocapitalizationType

首字母是否大写

  text.autocapitalizationType = UITextAutocapitalizationTypeNone;

typedef enum {

    UITextAutocapitalizationTypeNone, 不自动大写

    UITextAutocapitalizationTypeWords,　单词首字母大写

    UITextAutocapitalizationTypeSentences,　句子的首字母大写

    UITextAutocapitalizationTypeAllCharacters, 所有字母都大写

} UITextAutocapitalizationType;

## 17. returnKeyType

return键变成什么键

 text.returnKeyType =UIReturnKeyDone;

typedef enum {

    UIReturnKeyDefault, 默认 灰色按钮，标有Return

    UIReturnKeyGo,    　标有Go的蓝色按钮

    UIReturnKeyGoogle,标有Google的蓝色按钮，用语搜索

    UIReturnKeyJoin,标有Join的蓝色按钮

    UIReturnKeyNext,标有Next的蓝色按钮

    UIReturnKeyRoute,标有Route的蓝色按钮

    UIReturnKeySearch,标有Search的蓝色按钮

    UIReturnKeySend,标有Send的蓝色按钮

    UIReturnKeyYahoo,标有Yahoo的蓝色按钮

    UIReturnKeyYahoo,标有Yahoo的蓝色按钮

    UIReturnKeyEmergencyCall, 紧急呼叫按钮

} UIReturnKeyType;

## 18. keyboardAppearance

键盘外观

textView.keyboardAppearance=UIKeyboardAppearanceDefault；

typedef enum {

UIKeyboardAppearanceDefault， 默认外观，浅灰色

UIKeyboardAppearanceAlert，　 　深灰 石墨色

} UIReturnKeyType;

 19. delegate

设置代理 用于实现协议

  text.delegate = self;

## 20.  rightView

最右侧加图片是以下代码　 左侧类似

    UIImageView \*image=[[UIImageView alloc] initWithImage:[UIImage imageNamed:@"right.png"]];

    text.rightView=image;

    text.rightViewMode = UITextFieldViewModeAlways;

typedef enum {

    UITextFieldViewModeNever,

    UITextFieldViewModeWhileEditing,

    UITextFieldViewModeUnlessEditing,

    UITextFieldViewModeAlways

} UITextFieldViewMode;

## 21. editing

是否允许编辑。

## 22.界面重写绘制行为

除了UITextField对象的风格选项，你还可以定制化UITextField对象，为他添加许多不同的重写方法，来改变文本字段的显示行为。这些方法都会返回一个CGRect结构，制定了文本字段每个部件的边界范围。以下方法都可以重写。

– textRectForBounds:　　  　//重写来重置文字区域

– drawTextInRect:　　      　//改变绘文字属性.重写时调用super可以按默认图形属性绘制,若自己完全重写绘制函数，就不用调用super了.

– placeholderRectForBounds:　　//重写来重置占位符区域

– drawPlaceholderInRect:　　//重写改变绘制占位符属性.重写时调用super可以按默认图形属性绘制,若自己完全重写绘制函数，就不用调用super了.

– borderRectForBounds:　　//重写来重置边缘区域

– editingRectForBounds:　　//重写来重置编辑区域

– clearButtonRectForBounds:　　//重写来重置clearButton位置,改变size可能导致button的图片失真

– leftViewRectForBounds:

– rightViewRectForBounds:

## 22.调整键盘

inputView //代替标准的系统键盘

inputAccessoryView //编辑时显示在系统键盘或用户自定义的inputView上面的视图

## 23. delegate

类要遵守UITextFieldDelegate协议

text.delegate = self;　声明text的代理是我，我会去实现把键盘往下收的方法 这个方法在UITextFieldDelegate里所以我们要遵守UITextFieldDelegate这个协议

- (BOOL)textFieldShouldReturn:(UITextField \*)textField

{

    [text resignFirstResponder];    //主要是[receiver resignFirstResponder]在哪调用就能把receiver（text）对应的键盘往下收

return YES;

}

代理方法

- (BOOL)textFieldShouldBeginEditing:(UITextField \*)textField{

//返回一个BOOL值，YES代表允许编辑，NO不允许编辑.

    return YES;

}

- (void)textFieldDidBeginEditing:(UITextField \*)textField{

 //开始编辑时触发，文本字段将成为first responder

}

- (BOOL)textFieldShouldEndEditing:(UITextField \*)textField{

//返回BOOL值，指定是否允许文本字段结束编辑，当编辑结束，文本字段会让出first responder

　 //要想在用户结束编辑时阻止文本字段消失，可以返回NO　，返回NO，点击键盘的返回按钮会无效果。

　 //这对一些文本字段必须始终保持活跃状态的程序很有用，比如即时消息

    return NO;

}

- (void)textFieldDidEndEditing:(UITextField \*)textField;{

}// 上面返回YES后执行;上面返回NO时有可能强制执行(e.g. view removed from window)

- (BOOL)textField:(UITextField\*)textField shouldChangeCharactersInRange:(NSRange)range replacementString:(NSString \*)string{

//当用户使用自动更正功能，把输入的文字修改为推荐的文字时，就会调用这个方法。

//这对于想要加入撤销选项的应用程序特别有用

//可以跟踪字段内所做的最后一次修改，也可以对所有编辑做日志记录,用作审计用途。

//要防止文字被改变可以返回NO

//这个方法的参数中有一个NSRange对象，指明了被改变文字的位置，建议修改的文本也在其中

　   return YES;

}

- (BOOL)textFieldShouldClear:(UITextField \*)textField{

//返回一个BOOL值指明是否允许根据用户请求清除内容

//可以设置在特定条件下才允许清除内容

　   return YES;

}

-(BOOL)textFieldShouldReturn:(UITextField \*)textField{

//返回一个BOOL值，指明是否允许在按下回车键时结束编辑

 //如果允许要调用resignFirstResponder 方法，这回导致结束编辑，而键盘会被收起[textField resignFirstResponder];

//查一下resign这个单词的意思就明白这个方法了

　   return YES;

}

## 24.通知

UITextField派生自UIControl，所以UIControl类中的通知系统在文本字段中也可以使用。除了UIControl类的标准事件，你还可以使用下列UITextField类特有的事件

UITextFieldTextDidBeginEditingNotification

UITextFieldTextDidChangeNotification

UITextFieldTextDidEndEditingNotification

当文本字段退出编辑模式时触发。通知的object属性存储了最终文本。

因为文本字段要使用键盘输入文字，所以下面这些事件发生时，也会发送动作通知

UIKeyboardWillShowNotification 　//键盘显示之前发送

UIKeyboardDidShowNotification  　//键盘显示之后发送

UIKeyboardWillHideNotification 　//键盘隐藏之前发送

UIKeyboardDidHideNotification  　//键盘隐藏之后发送

## 25.Stroyboard中：

1、Text ：设置文本框的默认文本。

2、Placeholder ： 可以在文本框中显示灰色的字，用于提示用户应该在这个文本框输入什么内容。当这个文本框中输入了数据时，用于提示的灰色的字将会自动消失。

3、Background ：

4、Disabled ： 若选中此项，用户将不能更改文本框内容。

5、接下来是三个按钮，用来设置对齐方式。

6、Border Style ： 选择边界风格。

7、Clear Button ： 这是一个下拉菜单，你可以选择清除按钮什么时候出现，所谓清除按钮就是出一个现在文本框右边的小 X ，你可以有以下选择：

    7.1 Never appears ： 从不出现

    7.2 Appears while editing ： 编辑时出现

    7.3 Appears unless editing ：

    7.4 Is always visible ： 总是可见

8、Clear when editing begins ： 若选中此项，则当开始编辑这个文本框时，文本框中之前的内容会被清除掉。比如，你现在这个文本框 A 中输入了 "What" ，之后去编辑文本框 B，若再回来编辑文本框 A ，则其中的 "What" 会被立即清除。

9、Text Color ： 设置文本框中文本的颜色。

10、Font ： 设置文本的字体与字号。

11、Min Font Size ： 设置文本框可以显示的最小字体（不过我感觉没什么用）

12、Adjust To Fit ： 指定当文本框尺寸减小时，文本框中的文本是否也要缩小。选择它，可以使得全部文本都可见，即使文本很长。但是这个选项要跟 Min Font Size 配合使用，文本再缩小，也不会小于设定的 Min Font Size 。

接下来的部分用于设置键盘如何显示。

13、Captitalization ： 设置大写。下拉菜单中有四个选项：

    13.1 None ： 不设置大写

    13.2 Words ： 每个单词首字母大写，这里的单词指的是以空格分开的字符串

    13.3 Sentances ： 每个句子的第一个字母大写，这里的句子是以句号加空格分开的字符串

    13.4 All Characters ： 所以字母大写

14、Correction ： 检查拼写，默认是 YES 。

15、Keyboard ： 选择键盘类型，比如全数字、字母和数字等。

16、Appearance：

17、Return Key ： 选择返回键，可以选择 Search 、 Return 、 Done 等。

18、Auto-enable Return Key ： 如选择此项，则只有至少在文本框输入一个字符后键盘的返回键才有效。

19、Secure ： 当你的文本框用作密码输入框时，可以选择这个选项，此时，字符显示为星号。

1.Alignment Horizontal 水平对齐方式

2.Alignment Vertical 垂直对齐方式

3.用于返回一个BOOL值　输入框是否 Selected(选中) Enabled(可用) Highlighted(高亮)

## 26.UITextField实例一 ： UITextField限制字符 (只为数字)

**ios代码**

1. #define ALPHA @"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz "
2. #define NUMBERS @"0123456789n"
3. #define ALPHANUM @"ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789 "
4. #define NUMBERSPERIOD @"0123456789."

7. - (BOOL)textField:(UITextField \*)textField shouldChangeCharactersInRange:(NSRange)range replacementString:(NSString \*)string
8. {
9. NSCharacterSet \*cs;
10. cs = [[NSCharacterSet characterSetWithCharactersInString:NUMBERSPERIOD ] invertedSet]; //invertedSet 方法是去反字符,把所有的除了数字的字符都找出来
12. NSString \*filtered = [[string componentsSeparatedByCharactersInSet:cs] componentsJoinedByString:@""];  //componentsSeparatedByCharactersInSet 方法是把输入框输入的字符string 根据cs中字符一个一个去除cs字符并分割成单字符并转化为 NSArray, 然后componentsJoinedByString 是把NSArray 的字符通过 ""无间隔连接成一个NSString字符 赋给filtered.就是只剩数字了.

15. BOOL basicTest = [string isEqualToString:filtered];
16. **if**(!basicTest)
18. {
19. UIAlertView\* alert = [[UIAlertView alloc] initWithTitle:@"提示"
20. message:@"请输入数字"
21. delegate:nil
22. cancelButtonTitle:@"确定"
23. otherButtonTitles:nil];
25. [alert show];
26. [alert release];
27. **return** NO;
28. }
30. // Add any predicate testing here
31. **return** basicTest;
32. }

#define NUMBERS @”0123456789n” （这个代表可以输入数字和换行，请注意这个n，如果不写这个，Done按键将不会触发，如果用在SearchBar中，将会不触发Search事件，因为你自己限制不让输入n，好惨，我在项目中才发现的。） 所以，如果你要限制输入英文和数字的话，就可以把这个定义为： #define kAlphaNum @”ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789″。 当然，你还可以在以上方法return之前，做一提示的，比如提示用户只能输入数字之类的。如果你觉得有需要的话。

实例：限制只能输入一定长度的字符

**ios代码**

1. - (BOOL)textField:(UITextField \*)textField shouldChangeCharactersInRange:(NSRange)range replacementString:(NSString \*)string;
2. {　//string就是此时输入的那个字符 textField就是此时正在输入的那个输入框 返回YES就是可以改变输入框的值 NO相反
4. **if** ([string isEqualToString:@"n"])　//按回车可以改变
5. {
6. **return** YES;
7. }
9. NSString \* toBeString = [textField.text stringByReplacingCharactersInRange:range withString:string]; //得到输入框的内容
11. **if** (self.myTextField == textField)　//判断是否时我们想要限定的那个输入框
12. {
13. **if** ([toBeString length] > 20) { //如果输入框内容大于20则弹出警告
14. textField.text = [toBeString substringToIndex:20];
15. UIAlertView \*alert = [[[UIAlertView alloc] initWithTitle:nil message:@"超过最大字数不能输入了" delegate:nil cancelButtonTitle:@"Ok" otherButtonTitles:nil, nil] autorelease];
16. [alert show];
17. **return** NO;
18. }
19. }
20. **return** YES;
21. }

## 27.UITextField实例二：

在UISearchBar中,当输入信息改变时,它就会调用textDidChange函数,但是UITextField没有这个功能,唯一与这个类似的shouldChangeCharactersInRange函数,也是在文件还没有改变前就调用了,而不是在改变后调用,要想实现这个功能,我们可以增加事件监听的方式.先来看看objective-c提供的接口:

**ios代码**

1. // add target/action for particular event. you can call this multiple times and you can specify multiple target/actions for a particular event.
2. 给特殊事件添加目标或者方法，你能够调用这个方法多次，给特殊事件指定很多目标或者方法
3. // passing in nil as the target goes up the responder chain. The action may optionally include the sender and the event in that order
4. 给target传递空，会建立响应链，在这行命令，方法可以选择包含方法发送者和事件两个参数。
5. // the action cannot be NULL.   方法不能为空。
6. - (**void**)addTarget:(id)target action:(SEL)action forControlEvents:(UIControlEvents)controlEvents;

 怎么去使用这个接口呢?主要分为两步,第一步就是在UItextField组件中增加对文件编辑改变时事件的监听,然后再实现监听器监听到事件时,所调用的方法.

**ios代码**

1. //第一步,对组件增加监听器 可以在viewDidLoad 方法中加入 textField 为你自定义输入框的名称
2. [textField addTarget:self action:@selector(textFieldDidChange:) forControlEvents:UIControlEventEditingChanged];
3. ...
4. //第二步,实现回调函数
5. - (**void**) textFieldDidChange:(id) sender {
6. UITextField \*\_field = (UITextField \*)sender;
7. NSLog(@"%@",[\_field text]);
8. }

## 28.UITextField实例三：

UITextField控件处理键盘弹出时遮住输入框的问题

**打开键盘卷动文本字段**

默认情况下打开键盘会遮住下面的view，带来一点点困扰，不过这不是什么大问题，我们使用点小小的手段就可以解决。

首先我们要知道键盘的高度是固定不变的，不过在IOS 5.0 以后键盘的高度貌似不是216了，不过不要紧，我们调整调整就是了：

|  |  |  |
| --- | --- | --- |
|  | iPhone | ipad |
| 竖屏(portrait) | 216 | 264 |
| 横屏(landScape) | 140 | 352 |

我们采取的方法就是在textField(有可能是其他控件)接收到弹出键盘事件时把self.view整体上移216px了(我们就以iPhone竖屏为例了)。

首先我们要设置textField的代理，我们就设为当前控制器了。

**ios代码**

1. textField,delegate=self;

然后我们在当前控制器实现下面两个委托方法：

**ios代码**

1. - (void)textFieldDidBeginEditing:(UITextField \*)textField
2. { //当点触textField内部，开始编辑都会调用这个方法。textField将成为first responder
3. NSTimeInterval animationDuration = 0.30f;
4. CGRect frame = self.view.frame;
5. frame.origin.y -=216;
6. frame.size.height +=216;
7. self.view.frame = frame;
8. [UIView beginAnimations:@"ResizeView" context:nil];
9. [UIView setAnimationDuration:animationDuration];
10. self.view.frame = frame;
11. [UIView commitAnimations];
12. }

**ios代码**

1. - (BOOL)textFieldShouldReturn:(UITextField \*)textField
2. {//当用户按下ruturn，把焦点从textField移开那么键盘就会消失了
3. NSTimeInterval animationDuration = 0.30f;
4. CGRect frame = self.view.frame;
5. frame.origin.y +=216;
6. frame.size. height -=216;
7. self.view.frame = frame;
8. //self.view移回原位置
9. [UIView beginAnimations:@"ResizeView" context:nil];
10. [UIView setAnimationDuration:animationDuration];
11. self.view.frame = frame;
12. [UIView commitAnimations];
13. [textField resignFirstResponder];
14. }