

微信交流群

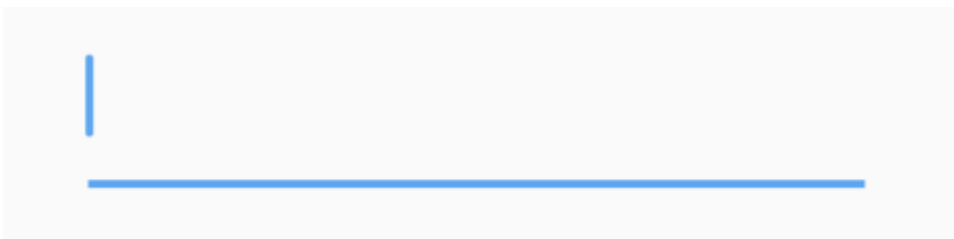


点击购买

TextField 是文本输入组件，即输入框，常用组件之一。基本用法：

```
1 TextField()
```

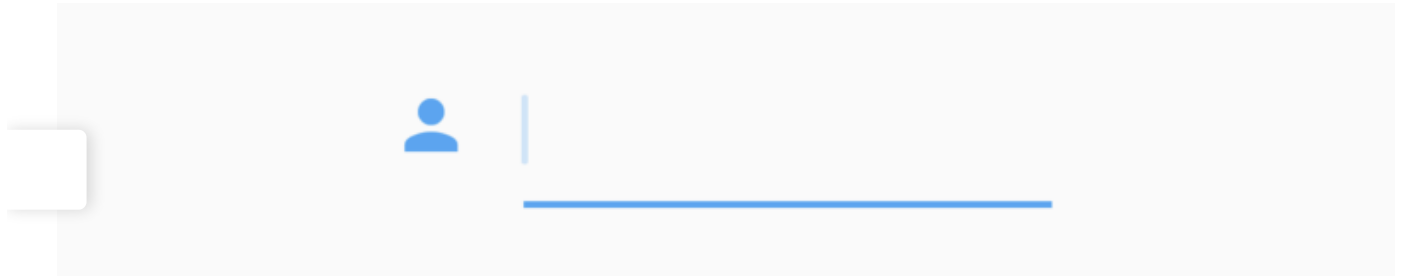
不需要任何参数，一个最简单的文本输入组件就出来了，效果如下：



decoration 是TextField组件的装饰（外观）参数，类型是InputDecoration。

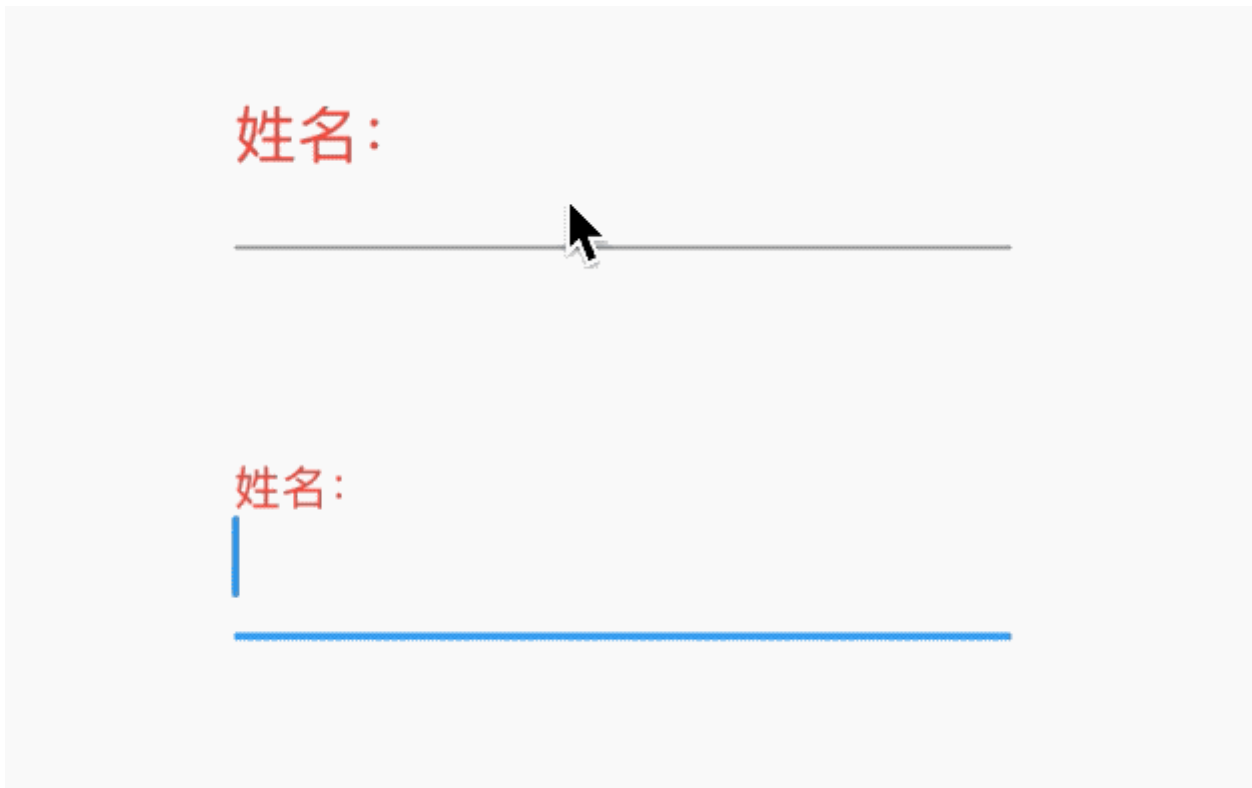
icon 显示在输入框的前面，用法如下：

```
1 TextField(  
2   decoration: InputDecoration(  
3     icon: Icon(Icons.person),  
4   ),  
5 )
```



当输入框是空而且没有焦点时，labelText显示在输入框上边，当获取焦点或者不为空时labelText往上移动一点， `labelStyle` 参数表示文本样式，具体参考 `TextStyle` ， 用法如下：

```
1  TextField(  
2    decoration: InputDecoration(  
3      labelText: '姓名: ',  
4      labelStyle: TextStyle(color: Colors.red)  
5    ),  
6  )
```



`hasFloatingPlaceholder` 参数控制当输入框获取焦点或者不为空时是否还显示 `labelText` ， 默认为true，显示。

`helperText` 显示在输入框的左下部，用于提示用户， `helperStyle` 参数表示文本样式，具体参考 `TextStyle` 用法如下：

```
1 TextField(  
2   decoration: InputDecoration(  
3     helperText: '用户名长度为6-10个字母',  
4     helperStyle: TextStyle(color: Colors.blue),  
5     helperMaxLines: 1  
6   ),  
7 )
```

用户名长度为6-10个字母

<https://blog.csdn.net/mengks1987>

hintText 是当输入框为空时的提示，不为空时不在显示，用法如下：

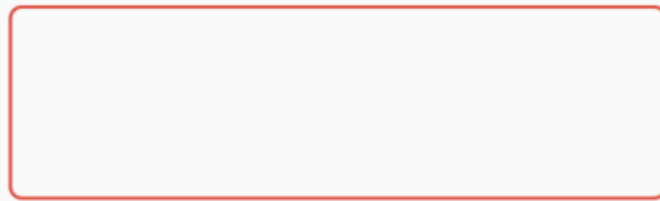
```
1 TextField(  
2   decoration: InputDecoration(  
3     hintText: '请输入用户名',  
4     hintStyle: TextStyle(color: Colors.grey),  
5     hintMaxLines: 1  
6   ),  
7 )
```

请输入用户名

errorText 显示在输入框的左下部，默认字体为红色，用法如下：

```
1 TextField(  
2   decoration: InputDecoration(  
3     errorText: '用户名输入错误',  
4     errorStyle: TextStyle(fontSize: 12),  
5     errorMaxLines: 1,  
6   ),  
7 )
```

```
6     errorBorder: OutlineInputBorder(borderSide: BorderSide(color: Color
7   ),
8 )
```



用户名输入错误

<https://blog.csdn.net/mengks1987>

prefix 系列的组件是输入框前面的部分，用法如下：

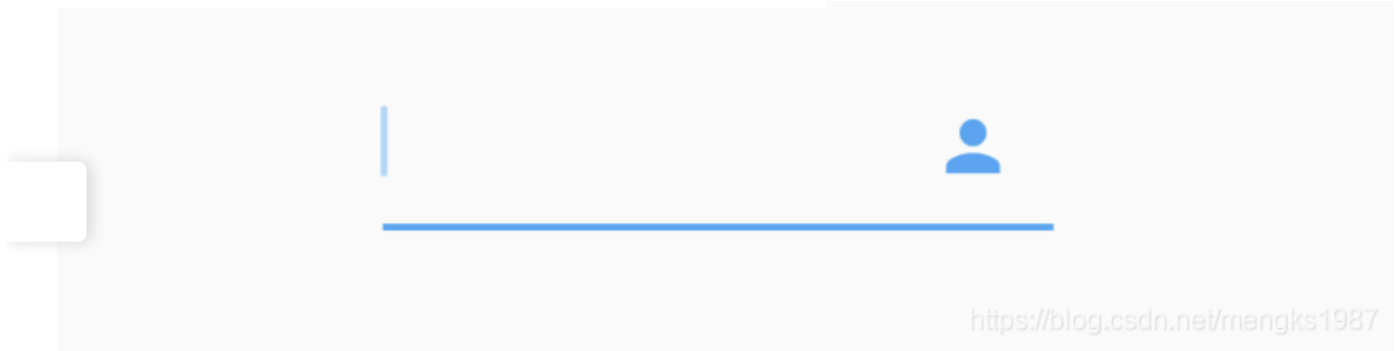
```
1  TextField(
2    decoration: InputDecoration(
3      prefixIcon: Icon(Icons.person)
4    ),
5  )
```



注意prefix和icon的区别，icon是在输入框边框的外部，而prefix在里面。

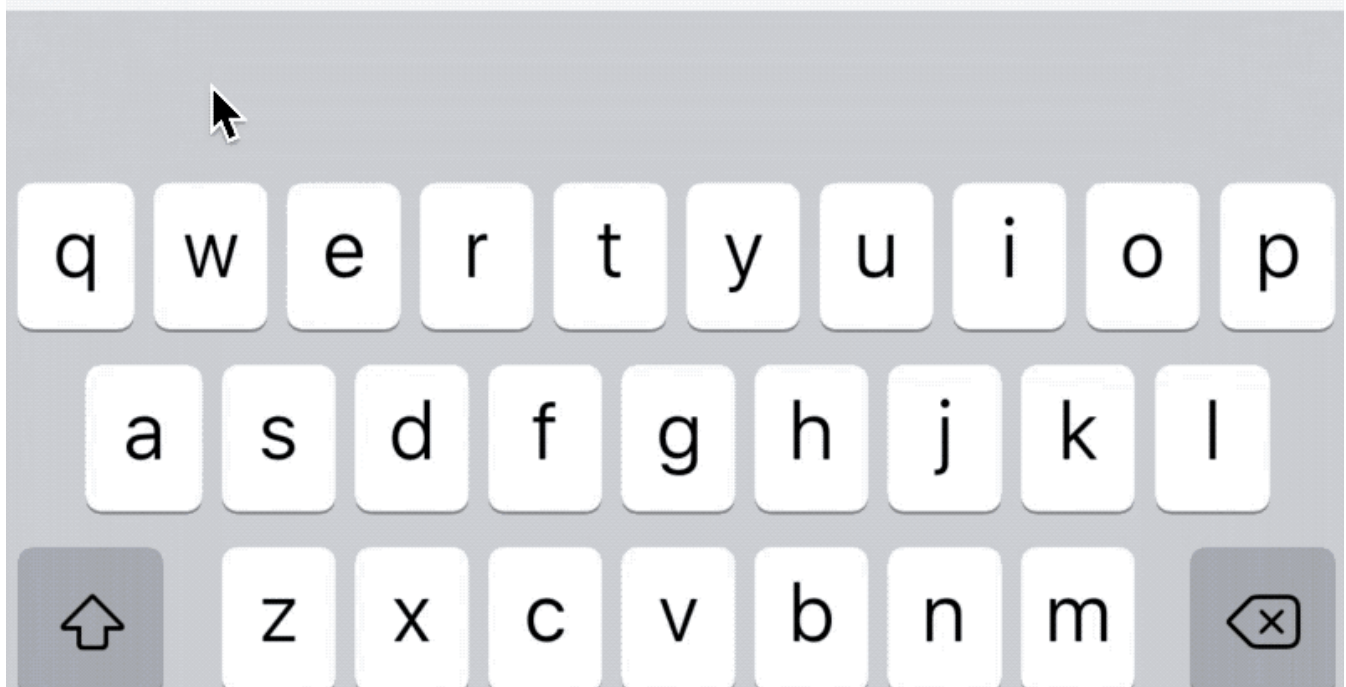
suffix和prefix相反，suffix在输入框的尾部，用法如下：

```
1  TextField(
2    decoration: InputDecoration(
3      suffixIcon: Icon(Icons.person)
4    ),
5  )
```



counter 组件统计输入框文字的个数，counter仅仅是展示效果，不具备自动统计字数的功能，自动统计字数代码如下：

```
1  var _textFieldValue = '';
2  TextField(
3    onChanged: (value){
4      setState(() {
5        _textFieldValue = value;
6      });
7    },
8    decoration: InputDecoration(
9      counterText: '${_textFieldValue.length}/32'
10   ),
11 )
```



`filled` 为true时，输入框将会被 `fillColor` 填充，仿QQ登录输入框代码如下：

```
1  Container(  
2    height: 60,  
3    width: 250,  
4    child: TextField(  
5      decoration: InputDecoration(  
6        fillColor: Color(0x30cccccc),  
7        filled: true,  
8        enabledBorder: OutlineInputBorder(  
9          borderSide: BorderSide(color: Color(0x00FF0000)),  
10         borderRadius: BorderRadius.all(Radius.circular(100))),  
11      hintText: 'QQ号/手机号/邮箱',  
12      focusedBorder: OutlineInputBorder(  
13        borderSide: BorderSide(color: Color(0x00000000)),
```

```
14         borderRadius: BorderRadius.all(Radius.circular(100))),  
15     ),  
16 ),  
17 )
```



QQ号/手机号/邮箱

<https://blog.csdn.net/mengks1987>

controller 是输入框文本编辑的控制器，可以获取TextField的内容、设置TextField的内容，下面将输入的英文变为大写：

```
1   TextEditingController _controller;  
2  
3   @override  
4   void initState() {  
5       super.initState();  
6       _controller = TextEditingController()  
7           ..addListener(() {  
8               //获取输入框的内容，变为大写  
9               _controller.text = _controller.text.toUpperCase();  
10          });  
11   }  
12  
13  
14   @override  
15   Widget build(BuildContext context) {  
16       return TextField(  
17           controller: _controller,  
18       );  
19   }  
20  
21   @override  
22   dispose() {  
23       super.dispose();  
24       _controller.dispose();  
25   }
```

有时输入框后面带有“清除”功能，需要controller来实现。如果需要2个TextField的内容进行同步，只需要给2个TextField设置同一个controller即可实现。

keyboardType 参数控制软键盘的类型，说明如下：

- text：通用键盘。
- multiline：当TextField为多行时（maxLines设置大于1），右下角的为“换行”按钮。
- number：数字键盘。
- phone：手机键盘，比数字键盘多"*"和"#”。
- datetime：在ios上和text一样，在android上出现数字键盘、":"和"-”。
- emailAddress：邮箱键盘，有"@"和"."按钮。
- url：url键盘，有"/"和"."按钮。
- visiblePassword：既有字母又有数字的键盘。

textInputAction 参数控制软键盘右下角的按钮，说明如下：

- none：android上显示返回键，ios不支持。
- unspecified：让操作系统自己决定哪个合适，一般情况下，android显示“完成”或者“返回”。
- done：android显示代表“完成”的按钮，ios显示“Done”（中文：完成）。
- go：android显示表达用户去向目的地的图标，比如向右的箭头，ios显示“Go”（中文：前往）。
- search：android显示表达搜索的按钮，ios显示“Search”（中文：搜索）。
- send：android显示表达发送意思的按钮，比如“纸飞机”按钮，ios显示“Send”（中文：发送）。
- next：android显示表达“前进”的按钮，比如“向右的箭头”，ios显示“Next”（中文：下一项）。
- previous：android显示表达“后退”的按钮，比如“向左的箭头”，ios不支持。
- continueAction：android 不支持，ios仅在ios9.0+显示“Continue”（中文：继续）。
- join：Android和ios显示“Join”（中文：加入）。
- route：android 不支持，ios显示“Route”（中文：路线）。
- emergencyCall：android 不支持，ios显示“Emergency Call”（中文：紧急电话）。
- newline：android显示表达“换行”的按钮，ios显示“换行”。

大家可能发现了，Android上显示的按钮大部分是不确定的，比如 **next** 有的显示向右的箭头，有的显示前进，这是因为各大厂商对Android ROM定制引发的。

textCapitalization 参数是配置键盘是大写还是小写，仅支持键盘模式为 **text** 的模式下忽略此配置，说明如下：

- words: 每一个单词的首字母大写。
- sentences: 每一句话的首字母大写。
- characters: 每个字母都大写
- none: 都小写

这里仅仅是控制软键盘是大写模式还是小写模式，你也可以切换大小写，系统并不会改变输入框内的内容。

`textAlignVertical` 表示垂直方向的对齐方式， `textDirection` 表示文本方向，用法如下：

```
1  TextField(  
2    textAlignVertical: TextAlignVertical.center,  
3    textDirection: TextDirection.rtl,  
4  )
```

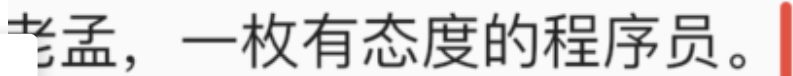
`toolbarOptions` 表示长按时弹出的菜单，有 `copy` 、 `cut` 、 `paste` 、 `selectAll` ，用法如下：

```
1  TextField(  
2    toolbarOptions: ToolbarOptions(  
3      copy: true,  
4      cut: true,  
5      paste: true,  
6      selectAll: true  
7    ),  
8  )
```

`cursor` 表示光标，用法如下：

```
1  TextField(  
2    showCursor: true,  
3    cursorWidth: 3,  
4    cursorRadius: Radius.circular(10),  
5    cursorColor: Colors.red,  
6  )
```

效果如下：



将输入框设置为密码框，只需 `obscureText` 属性设置true即可，用法如下：

```
1  TextField(  
2    obscureText: true,  
3  )
```

通过 `inputFormatters` 可以限制用户输入的内容，比如只想让用户输入字符，设置如下：

```
1  TextField(  
2    inputFormatters: [  
3    WhitelistingTextInputFormatter(RegExp("[a-zA-Z]")),  
4  ],  
5  )
```

这时用户是无法输入数字的。

`onChanged` 是当内容发生变化时回调，`onSubmitted` 是点击回车或者点击软键盘上的完成回调，`onTap` 点击输入框时回调，用法如下：

```
1  TextField(  
2    onChanged: (value){  
3      print('onChanged:$value');  
4    },  
5    onEditingComplete: (){  
6      print('onEditingComplete');  
7    },  
8  
9    onTap: (){  
10     print('onTap');  
11   },  
12  )
```

输入框右下角经常需要字数统计，除了使用上面介绍的方法外，还可以使用 `buildCounter`，建议使用此方法，用法如下：

```
1  TextField(  
2    maxLength: 100,  
3    buildCounter: (  
4      BuildContext context, {  
5        int currentLength,  
6        int maxLength,  
7        bool isFocused,  
8      }) {  
9        return Text(  
10         '$currentLength/$maxLength',  
11       );  
12     },  
13   )
```

动态获取焦点

```
1  FocusScope.of(context).requestFocus(_focusNode);
```

`_focusNode` 为TextField的focusNode：

```
1  _focusNode = FocusNode();  
2  
3  TextField(  
4    focusNode: _focusNode,  
5    ...  
6  )
```

动态失去焦点

```
1  _focusNode.unfocus();
```

喜欢作者



版权所有，禁止私自转发、克隆网站。

[< 2.2 富文本组件-RichText](#)

[2.4 文本组件五大案例 >](#)