TextField () Widget:

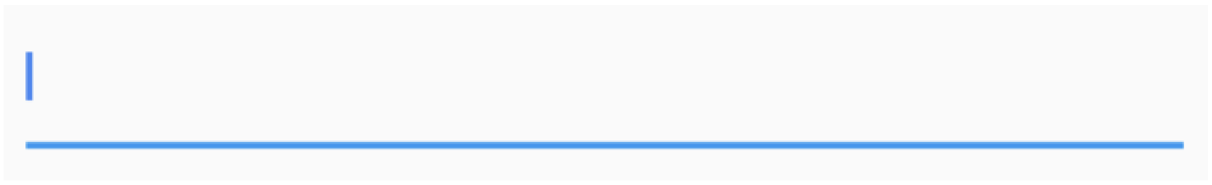
A TextField or TextBox is an **input element** which holds the alphanumeric data, such as name, password, address, etc. It is a GUI control element that enables the user to **enter text information** using a programmable code. It can be of a single-line text field (when only one line of information is required) or multiple-line text field (when more than one line of information is required).

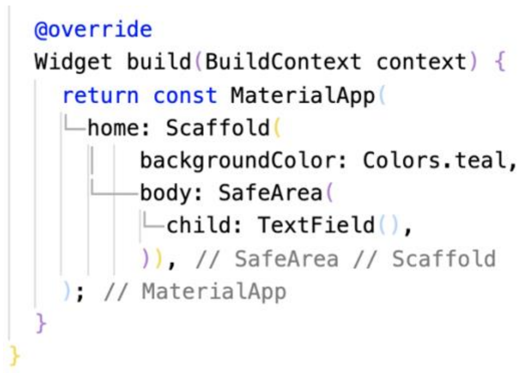
TextField in Flutter is the most commonly used **text input widget** that allows users to collect inputs from the keyboard into an app. We can use the **TextField** widget in building forms, sending messages, creating search experiences, and many more. By default, Flutter decorated the TextField with an underline. We can also add several attributes with TextField, such as label, icon, inline hint text, and error text using an InputDecoration as the decoration. If we want to remove the decoration properties entirely, it is required to set the decoration to **null**.

A TextField widget allows collection of information from the user. The code for a basic TextField is as simple as:

```
TextField()
```

This creates a basic TextField:

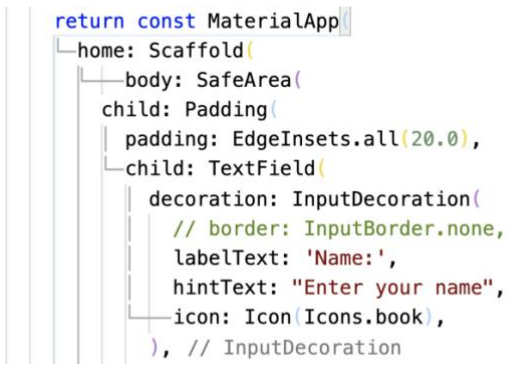




```
@override
Widget build(BuildContext context) {
  return const MaterialApp(
    home: Scaffold(
      backgroundColor: Colors.teal,
      body: SafeArea(
        child: TextField(),
      )), // SafeArea // Scaffold
  ); // MaterialApp
}
}
```

There are various of properties. Some of them are :

- decoration: which takes InputDecoration as values having attributes- border, labelText, hintText, icon, errorText, errorStyle etc.



```
return const MaterialApp(
  home: Scaffold(
    body: SafeArea(
    child: Padding(
      padding: EdgeInsets.all(20.0),
      child: TextField(
        decoration: InputDecoration(
          // border: InputBorder.none,
          labelText: 'Name:',
          hintText: "Enter your name",
          icon: Icon(Icons.book),
        ), // InputDecoration
```

- style: which takes TextStyle as values.

- To autofocus on a TextField when the widget is created, set the autofocus field to true.

```
TextField(
  autofocus: true,
),
```

A TextField allows you to customise the **type of keyboard** that shows up when the TextField is brought into focus. We change the **keyboardType** property for this.

```
TextField(
  keyboardType: TextInputType.number,
),
```

The types are:

- **TextInputType.text** (Normal complete keyboard)

- **TextInputType.number** (A numerical keyboard)

- **TextInputType.emailAddress** (Normal keyboard with an "@")

- **TextInputType.datetime** (Numerical keyboard with a "/" and ":")

- **TextInputType.numberWithOptions** (Numerical keyboard with options to enabled signed and decimal mode)

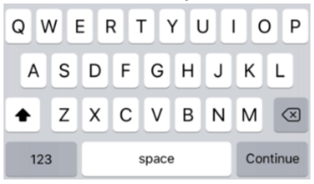
- **TextInputType.multiline** (Optimises for multi-line information)

Changing textInputAction of the TextField lets you change **the action button of the keyboard** itself.

As an example:

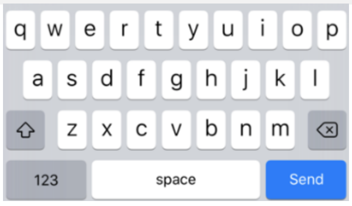
```
TextField(
  textInputAction: TextInputAction.continueAction,
),
```

This causes the "Done" button to be replaced by a "Continue" button:



Or

```
TextField(
  textInputAction: TextInputAction.send,
),
```



Enables or disables autocorrect for the specific TextField. Use the autocorrect field to set this.

```
TextField(
  autocorrect: false,
),
```

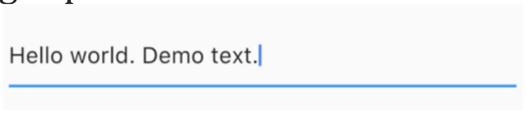
The TextField provides a few options on how to capitalise letters in the input from the user.

```
TextField(
  textCapitalization: TextCapitalization.sentences,
),
```

The types are:

1. **TextCapitalization.sentences**

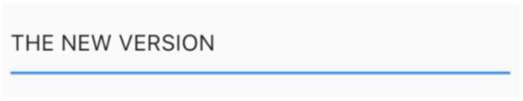
This is the normal type of capitalisation we expect, the first letter of every sentence being capitalised.



Hello world. Demo text.

2. **TextCapitalization.characters**

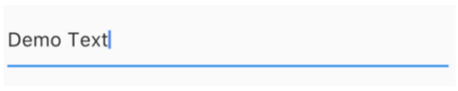
Capitalises all characters in the sentences.



THE NEW VERSION

3. **TextCapitalization.words**

Capitalises the first letter of each word.

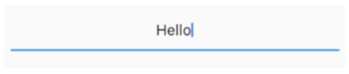

Demo Text

Text alignment inside the TextField

Use the textAlign property to adjust where cursor is inside the TextField.

```
TextField(
  textAlign: TextAlign.center,
),
```

This causes the cursor and text to start in the middle of the TextField.


Hello

This has the usual alignment properties: **start, end, left, right, center, justify**.

Styling the text inside the TextField

We use the **style** property to change how the text inside the TextField looks. Use it to change the color, font size, etc. This is similar to the style property in the Text widget so we will not spend too much time exploring it.

```
TextField(
  style: TextStyle(color: Colors.red, fontWeight: FontWeight.w300),
),
```

Changing the cursor in the TextField

The cursor is customisable directly from the TextField widget.

You are allowed to change the cursor color, width and radius of the corners. For example, here I make a circular red cursor for no apparent reason.

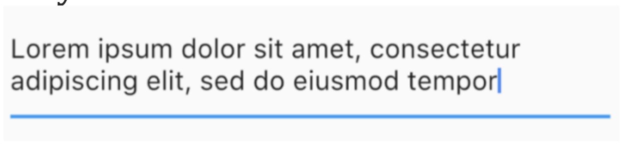
```
TextField(
  cursorColor: Colors.red,
  cursorRadius: Radius.circular(16.0),
  cursorWidth: 16.0,
),
```

Controlling max characters

```
TextField(
  maxLength: 4,
),
```

Making an expandable TextField

Sometimes, we need a TextField that expands when one line is finished. In Flutter it is slightly odd (yet easy) to do. To do this, **we set maxLines to null**, which is 1 by default. Setting to null is not something we're very used to but nevertheless it's easy to do.

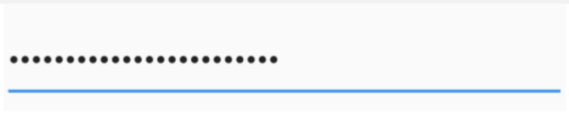

Lorem ipsum dolor sit amet, consectetur
adipiscing elit, sed do eiusmod tempor

Note: Setting the maxLines to a direct value will expand it to that number of lines by default.

```
TextField(
  maxLines: 3,
)
```

Obscuring TextTo obscure text in a TextField, set obscureText to true.

```
TextField(
  obscureText: true,
),
```


••••••••••••••••••••••••

Use “border” to give a border to the TextField

```
TextField(
  decoration: InputDecoration(
    border: OutlineInputBorder()
  )
),
```