

# Flutter Checklist

## Table of contents

<b>Table of contents</b>	<b>1</b>
<b>Widgets</b>	<b>2</b>
Master Layouts	2
Layouts	2
Visual Layouts	2
User Inputs	3
Show	4
Builders	5
States	5
Navigators	6
<b>Dart</b>	<b>7</b>
Data types	7
Functions	8
Classes	9
Pass Data	9
Null Safety	11
Dart exercises	12
<b>Other</b>	<b>14</b>
Auto const	14
Extensions	14
Shortcuts	15
Start a new project	15
Structure	15
Packages	16
Assets	16
Key concepts	17
Change app name	18
<b>Debug</b>	<b>19</b>
Type 1	19
Type 2	19
<b>Next Steps</b>	<b>20</b>
Firebase	20
Save data locally	20

# Widgets

Check on <https://fluttermapp.com/widgets>

## Master Layouts

- Scaffold
- SafeArea
- AppBar
- FloatingActionButton
- NavigationBar
- BottomNavigationBar
- TabBarView (with TabBar and DefaultTabController)
- PageView
- Navigation Drawer

## Layouts

- SizeBox
- Column
- Row
- Wrap
- Stack
- Center
- Padding
- SingleChildScrollView
- ListView
- GridView
- Spacer
- Expanded
- Flexible
- FittedBox (Work best with Text)
- RotatedBox
- Align
- Positioned

## Visual Layouts

- Container
- Color Colors.blueGrey.withOpacity(0.5)
- ListTile
- GridTile
- GridTileBar

Divider  
VerticalDivider  
Card  
Text  
Image  
Icon  
CircularProgressIndicator  
LinearProgressIndicator  
Badge  
DataTable  
CircleAvatar  
BackdropFilter  
ClipRRect

## User Inputs

ElevatedButton  
FilledButton  
OutlinedButton  
TextButton  
IconButton  
GestureDetector  
InkWell  
Dismissible  
Checkbox (CheckboxListTile)  
Switch (SwitchListTile)  
TextField  
TextFormField  
Form  
DropDownMenu  
DropDownButton  
PopupMenuButton  
RadioButton  
Slider  
Chip  
ChoiceChip  
ToggleButton  
Visibility  
ExpansionTile  
CloseButton

## Show

showBottomSheet  
showDatePicker  
showDateRangePicker  
showDialog  
showLicensePage  
showModalBottomSheet  
showSearch  
showTimePicker  
showCupertinoDialog  
showCupertinoModalPopup  
showSnackBar (ScaffoldMessenger)  
showMaterialBanner (ScaffoldMessenger)

```
showDialog(  
  context: context,  
  builder: (context) {  
    return const AlertDialog(  
      content: Text('Dialog'),  
      actions: [  
        CloseButton(),  
      ],  
    );  
  },  
);
```

```
ScaffoldMessenger.of(context).showSnackBar(  
  SnackBar(  
    content:  
      Text('Snackbar'),  
    behavior: SnackBarBehavior.floating,  
  ),  
);
```

## Builders

FutureBuilder

StreamBuilder

Builder

LayoutBuilder (MediaQuery.of(context).size)

OrientationBuilder (MediaQuery.of(context).orientation)

ValueListenableBuilder (with ValueNotifier)

```
ValueNotifier<bool> isDarkNotifier = ValueNotifier(false);
```

```
FloatingActionButton(  
  onPressed: () {  
    isDarkNotifier.value = !isDarkNotifier.value;  
  },  
  child: ValueListenableBuilder(  
    valueListenable: isDarkNotifier,  
    builder: (context, isDark, child) {  
      if (isDark) {  
        return const Icon(Icons.light_mode);  
      } else {  
        return const Icon(Icons.dark_mode);  
      }  
    },  
  ),  
)
```

## States

Stateless (Can't refresh the screen)

Stateful (Can refresh the screen)

SetState (Refresh the screen)

```
setState(() {  
  //Do something  
});
```

## Navigators

Pop

PushReplacement

Push

```
Navigator.push(  
    context,  
    MaterialPageRoute(  
        builder: (context) {  
            return const HomePage();  
        },  
    ),  
);
```

# Dart

## Data types

### Strings

```
String value = 'Hello';
```

- .substring
- .split
- .toLowerCase
- .toUpperCase

### Int

```
int value = 5;
```

- .isEven
- .isOdd
- .toDouble
- .toString
- .toStringAsFixed

### Double

```
double value = 5.0;
```

- .round
- .toInt
- .toString
- .toStringAsFixed

### Bool

```
bool value = true;
```

### Lists

```
List<bool> values = [true, false, true, false];
```

- .add
- .length
- .generate

### Maps

```
Map<String, bool> values = {  
  'value1': true,  
  'value2': false,  
  'value3': true,  
};
```

- .addAll

## Functions

### Looping

```
for (var i = 0; i < 5; i++) {  
    print(i);  
}
```

```
int i = 0;  
while (i < 10) {  
    i = i++;  
    doThis();  
}
```

### Arguments

```
void doSomething() {  
    doThis(stringValue: 'Hello');  
    doThis(stringValue: 'Hello', intValue: 5);  
    doThat('Hello', null);  
    doThat('Hello', 5);  
}  
  
void doThis({required String stringValue, int? intValue}) {  
    print(stringValue);  
    print(intValue);  
}  
  
void doThat(String stringValue, int? intValue) {  
    print(stringValue);  
    print(intValue);  
}
```

### Conditions

```
void doSomething() {  
    if (0 == 1) {  
        doThis();  
    } else {  
        doThat();  
    }  
}
```

```
void doSomething() {  
    0 == 1 ? doThis() : doThat();  
}
```



## Classes

```
class CardClass {  
    CardClass({  
        required this.title,  
        required this.imagePath,  
    });  
    String title;  
    String imagePath;  
}
```

## Pass Data

```
class RandomPage extends StatelessWidget {  
    const RandomPage({  
        super.key,  
        required this.title,  
        this.description,  
        this.isDark = false,  
        required this.function,  
        required this.callback,  
    });  
    final String title;  
    final String? description;  
    final bool isDark;  
    final Function(int) function;  
    final VoidCallback callback;  
  
    @override  
    Widget build(BuildContext context) {  
        return Scaffold(  
            body: Column(  
                children: [  
                    Text(title),  
                    Text(description ?? 'Default'),  
                    Text(isDark ? 'Is dark' : 'Is light'),  
                    TextButton(  
                        onPressed: () {  
                            callback();  
                        },  
                    ),  
                ],  
            ),  
        );  
    }  
}
```

```

        },
        child: const Text('Callback'),
      ),
      TextButton(
        onPressed: () {
          function(5);
        },
        child: const Text('Function'),
      ),
    ],
  ),
);
}
}

```

```

class RandomPage extends StatefulWidget {
  const RandomPage({
    super.key,
    required this.title,
    this.description,
    this.isDark = false,
    required this.function,
    required this.callback,
  });
  final String title;
  final String? description;
  final bool isDark;
  final Function(int) function;
  final VoidCallback callback;

  @override
  State<RandomPage> createState() => _RandomPageState();
}

```

```

class _RandomPageState extends State<RandomPage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Column(
        children: [

```

```

    Text(widget.title),
    Text(widget.description ?? 'Default'),
    Text(widget.isDark ? 'Is dark' : 'Is light'),
    TextButton(
      onPressed: () {
        widget.callback();
      },
      child: const Text('Callback'),
    ),
    TextButton(
      onPressed: () {
        widget.function(5);
      },
      child: const Text('Function'),
    ),
  ],
),
);
}
}

```

## Null Safety

```

// The "?" means "This can be null";
// The "??" means "If it's null, then show the default value"
// The "!" means "I confirm that this value is not null"

String? value1 = 'Hello';
String? value2;
String value3 = 'Hi';

String value4 = value1!;           // value4 = 'Hello'
String value5 = value1 ?? value3; // value5 = 'Hello'

String value6 = value2!;           // value6 = ERROR
String value7 = value2 ?? value3;  // value7 = 'Hi'
String value8 = value2 ?? value2!; // value8 = ERROR

String value9 = value3;           // value9 = 'Hi'

```

## Dart exercises

(Data types)

```
1 void main() {
2   String name = 'Flutter Mapp';
3   String food = 'Pizza';
4   int number = 50;
5
6   print ('My name is $name, I like $food and the best number is $number');
7 }
8
```

Run

Console

My name is Flutter Mapp, I like Pizza and the best number is 50

(Classes, arguments, data types)

```
1 class Person {
2   Person({
3     required this.name,
4     required this.food,
5     required this.number,
6   });
7   String name;
8   String food;
9   int number;
10 }
11
12 void main() {
13   Person person = Person(name: 'Flutter Mapp', food: 'Pizza', number: 5);
14
15   print ('My name is ${person.name},');
16   print ('I like ${person.food},');
17   print ('The best number is ${person.number}.');
18 }
19
```

Run

Console

My name is Flutter Mapp,  
I like Pizza,  
The best number is 5.

Documentation

(Data types)

```
1 void main() {
2   double number = 2.5;
3   bool isAvailable = true;
4   List wordsList = ['Flutter', 'Mapp', 'YouTube'];
5   Map translateMap = {
6     'Monday': 'Lundi',
7     'Friday': 'Vendredi',
8     'Sunday': 'Dimanche',
9   };
10
11   print ('The best decimal number is $number,');
12   print ('Is it available? Answer: $isAvailable');
13   print ('What is the first word in the list? Answer: ${wordsList[0]}');
14   print ('What is the second word in the list? Answer: ${wordsList[1]}');
15   print ('What is Moday in French? Answer: ${translateMap['Monday']}');
16 }
17
```

Run

Console

The best decimal number is 2.5,  
Is it available? Answer: true  
What is the first word in the list? Answer: Flutter  
What is the second word in the list? Answer: Mapp  
What is Moday in French? Answer: Lundi

Documentation

(Function, looping, conditions, data types)

```
1 void main() {
2   List wordsList = ['Flutter', 'Mapp', 'YouTube'];
3   bool? isAvailable;
4   int number = 1;
5
6   if(number>0){
7     isAvailable = true;
8   }else{
9     isAvailable = false;
10  }
11
12  printEveryWords(list: wordsList);
13  print(isAvailable ? 'Available': 'Unavailable');
14 }
15
16 void printEveryWords ({required List list}){
17   for (var i = 0; i < list.length; i++) {
18     print(list[i]);
19   }
20 }
```

Run

Console

Flutter  
Mapp  
YouTube  
Available

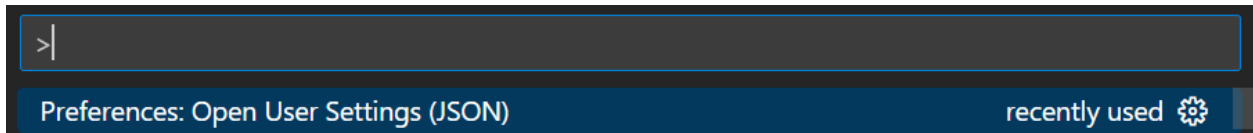
Documentation

# Other

## Auto const

With Visual Studio Code only

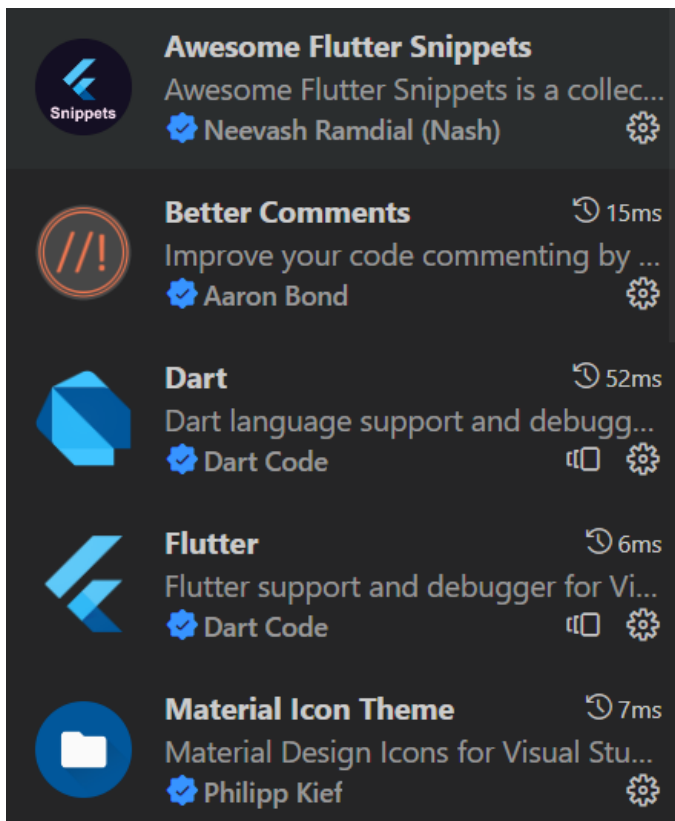
(Ctrl+Shift+P)



Add this to the file:

```
"editor.codeActionsOnSave": {  
  "source.fixAll": true  
},
```

## Extensions



## Shortcuts

Ctrl+b = Minimize/Maximize the Visual Studio Side Bar

Ctrl+j = Minimize/Maximize the Console

Alt+Shift+f = Format Document (Structure the code and make sure to add commas)

Refactor (Right Click) = Wrap with a Widget

Ctrl+space (Command+space for mac) = See all the possible arguments available

Ctrl+s = Save

Ctrl+x = Cut

Ctrl+v = Paste

Ctrl+c = Copy

Ctrl+d = Select the next identical word

Ctrl+l = Select the entire row

## Start a new project

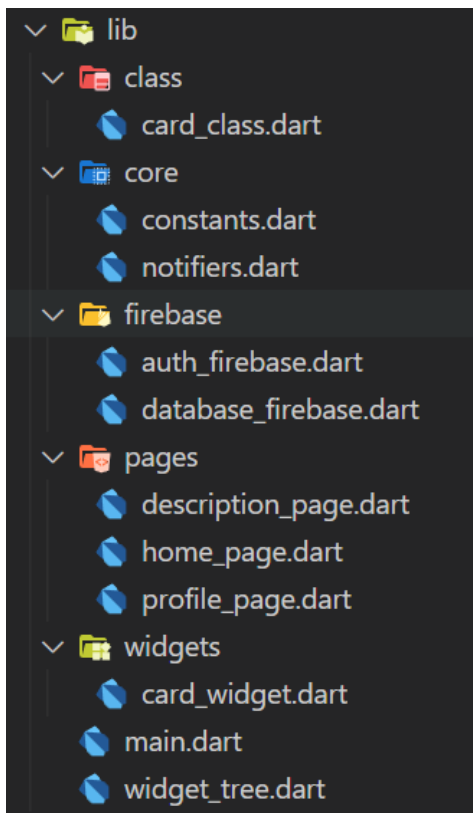
This will automatically set up your application Id, so you will save time in the future

```
flutter create --org com.yourwebsite your_app_name
```

This will not set your application Id

```
flutter create your_app_name
```

## Structure



## Packages

Go on <https://pub.dev/>

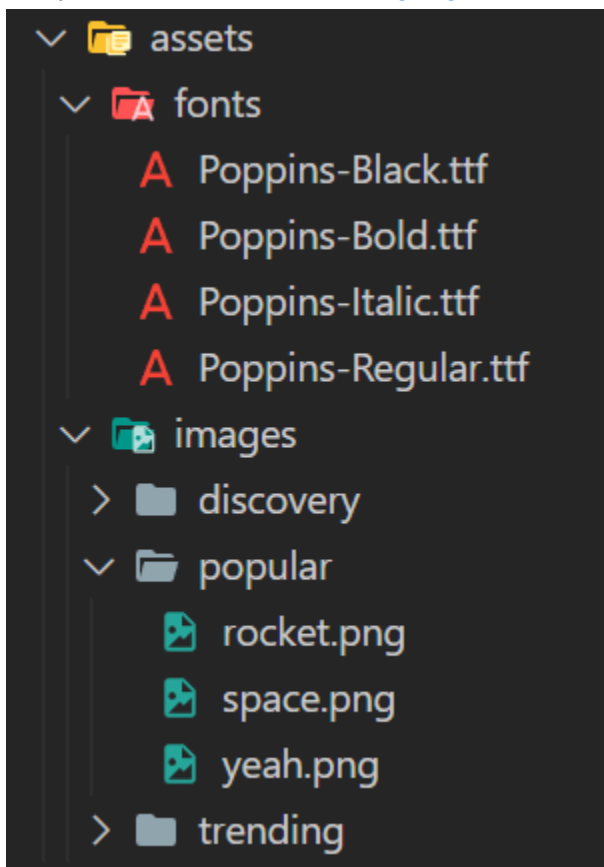
Add in dependencies or dev\_dependencies

```
dependencies:
  flutter:
    sdk: flutter
  cupertino_icons: ^1.0.2

dev_dependencies:
  flutter_test:
    sdk: flutter
  flutter_launcher_icons: ^0.12.0
```

## Assets

Get your fonts on <https://fonts.google.com/>





```
assets:
  - assets/images/discovery/
  - assets/images/popular/
  - assets/images/trending/

fonts:
  - family: Poppins
    fonts:
      - asset: assets/fonts/Poppins-Regular.ttf
      - asset: assets/fonts/Poppins-Bold.ttf
        weight: 700
      - asset: assets/fonts/Poppins-Black.ttf
        weight: 900
      - asset: assets/fonts/Poppins-Italic.ttf
        style: italic
```

## Key concepts

Flutter is Widget inside Widget

- Add a Widget
- Add an argument in the Widget (Ctrl+space) (Command+space for mac)
- Add another Widget in the argument
- Do this forever (Until your app is done)

Widgets **always** start with a **capital letter**.

Arguments **always** start with a **lowercase letter**.

In this example (Read carefully the capital and lowercase letters):

- **Scaffold** is a Widget
- **appBar** is an argument of the **Scaffold** Widget
- **AppBar** is a Widget placed inside the **appBar** argument.
- **title** is an argument of the **AppBar** Widget
- **Text** is a Widget placed inside the **title** argument.

```
return Scaffold(
  appBar: AppBar(
    title: const Text('Mapp blog'),
  ), // AppBar
```

Remember, it's Widget inside Widget!

Use this to see all the possible arguments available (Ctrl+space) (Command+space for mac)

## Change app name

### ▲ Android

552 Open `AndroidManifest.xml` (located at `android/app/src/main`)



```
<application
  android:label="App Name" ...> // Your app name here
```

### iOS

Open `info.plist` (located at `ios/Runner`)

```
<key>CFBundleDisplayName</key>
<string>App Name</string> // Your app name here
```

# Debug

## Type 1

If you have an error like this (With red curly lines):

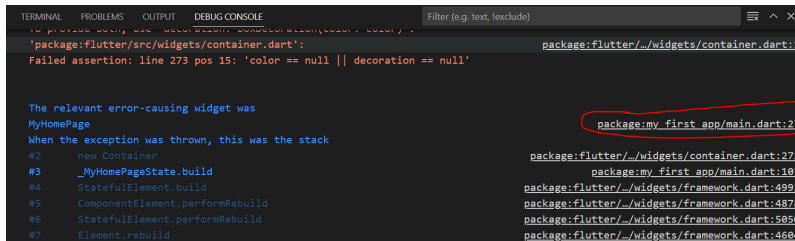
```
ElevatedButton(  
  onPressed: () {},  
)
```

- Put your mouse over the Widget with the red curly line.
- A box will appear, scroll down at the bottom of the box
- Read the error
- This is what you should see:

```
required Widget? child,  
)  
package:flutter/src/material/elevated_button.dart  
Create an ElevatedButton.  
The [autofocus] and [clipBehavior] arguments must not be null.  
Expected to find ' '. dart(expected_token)  
View Problem (Alt+F8) No quick fixes available  
ElevatedButton(  
  onPressed: () {},  
  style: ElevatedButton.styleFrom(  
    ...  
  ),  
)
```

## Type 2

You can also have hidden errors (Sometime the screen will be red)



```
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE  
Filter (e.g. text, exclude)  
'package:flutter/src/widgets/container.dart': package:flutter/src/widgets/container.dart:1  
Failed assertion: line 273 pos 15: 'color == null || decoration == null'  
  
The relevant error-causing widget was  
MyHomePage  
When the exception was thrown, this was the stack  
#2 new Container  
#3 _MyHomePageState.build  
#4 StatefulElement.build  
#5 ComponentElement.performRebuild  
#6 StatefulElement.performRebuild  
#7 Element.rebuild  
package:my_first_app/main.dart:27  
package:flutter/src/widgets/container.dart:273  
package:my_first_app/main.dart:101  
package:flutter/src/widgets/framework.dart:4992  
package:flutter/src/widgets/framework.dart:4878  
package:flutter/src/widgets/framework.dart:5950  
package:flutter/src/widgets/framework.dart:4604
```

- Open the Debug Console in the terminal
- Click the link on the top and it will bring you to your error in the code
- If you can't understand, google search the error

## Next Steps

### Firestore

Authenticate users

Create a database

### Save data locally

Use this: [https://pub.dev/packages/shared\\_preferences](https://pub.dev/packages/shared_preferences)