Week-2

2a) Installation and setup in VS Code.

Updating the extension

Updates to the extensions are shipped on a regular basis. By default, VS Code automatically updates extensions when updates are available.

To install updates yourself:

1. Click **Extensions** in the Side Bar.
2. If the Flutter extension has an available update, click **Update** and then **Reload**.
3. Restart VS Code.

Creating projects
#
There are a couple ways to create a new project.

Creating a new project
#
To create a new Flutter project from the Flutter starter app template:

1. Go to **View** > **Command Palette….**
2. You can also press Ctrl / Cmd + Shift + P.
3. Type flutter.
4. Select the **Flutter: New Project**.
5. Press Enter.
6. Select **Application**.
7. Press Enter.
8. Select a **Project location**.
9. Enter your desired **Project name**.

Opening a project from existing source code

To open an existing Flutter project:

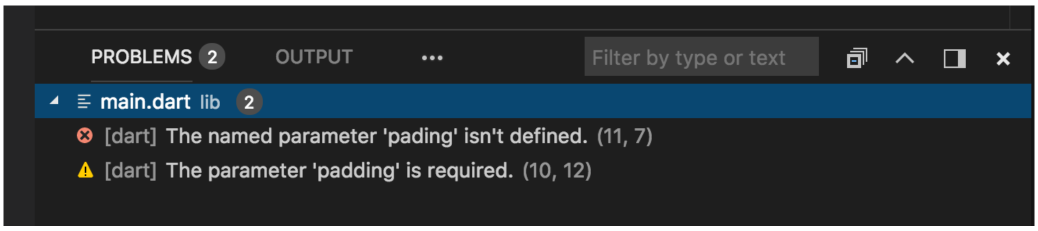
1. Go to **File** > **Open**.
2. You can also press Ctrl / Cmd + O
3. Browse to the directory holding your existing Flutter source code files.
4. Click **Open**

Editing code and viewing issues

The Flutter extension performs code analysis. The code analysis can:

- Highlight language syntax
- Complete code based on rich type analysis
- Navigate to type declarations
- Go to **Go** > **Go to Definition**.

- You can also press F12.
- Find type usages.
- Press Shift + F12.
- View all current source code problems.
    - Go to **View** > **Problems**.
    - You can also press Ctrl / Cmd + Shift + M.
    - The Problems pane displays any analysis issues:



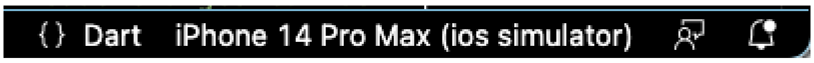
main.dart lib 2
[dart] The named parameter 'pading' isn't defined. (11, 7)
[dart] The parameter 'padding' is required. (10, 12)

Running and debugging

Start debugging by clicking **Run** > **Start Debugging** from the main IDE window, or press F5.

- **Selecting a target device**

When a Flutter project is open in VS Code, you should see a set of Flutter specific entries in the status bar, including a Flutter SDK version and a device name (or the message **No Devices**):



How to perform a hot reload

To hot reload a Flutter app:

1. Run the app from a supported Flutter editor or a terminal window. Either a physical or virtual device can be the target. **Only Flutter apps in debug mode can be hot reloaded or hot restarted.**

2. Modify one of the Dart files in your project. Most types of code changes can be hot reloaded; for a list of changes that require a hot restart, see Special cases.

3. If you're working in an IDE/editor that supports Flutter's IDE tools, select **Save All** (cmd-s/ctrl-s), or click the hot reload button on the toolbar.

4. If you're running the app at the command line using flutter run, enter r in the terminal window.

2b)Explore various Flutter Text Widget.

Text Widget:

```
import 'package:flutter/material.dart';

// function to trigger build process
void main() => runApp(const GeeksforGeeks());

class GeeksforGeeks extends StatelessWidget {
const GeeksforGeeks({Key? key}) : super(key: key);

@override
Widget build(BuildContext context) {
        return MaterialApp(
        home: Scaffold(
                backgroundColor: Colors.lightGreen,
                appBar: AppBar(
                backgroundColor: Colors.green,
                title: const Text("welcome Screen"),
                ), // AppBar
                body: Container(
                child: const Center(
                child: Text("Hello world!!"),
                ), // Center
                ), // Container
        ), // Scaffold
        ); // MaterialApp
}
}
```

Output: Hello World!!