

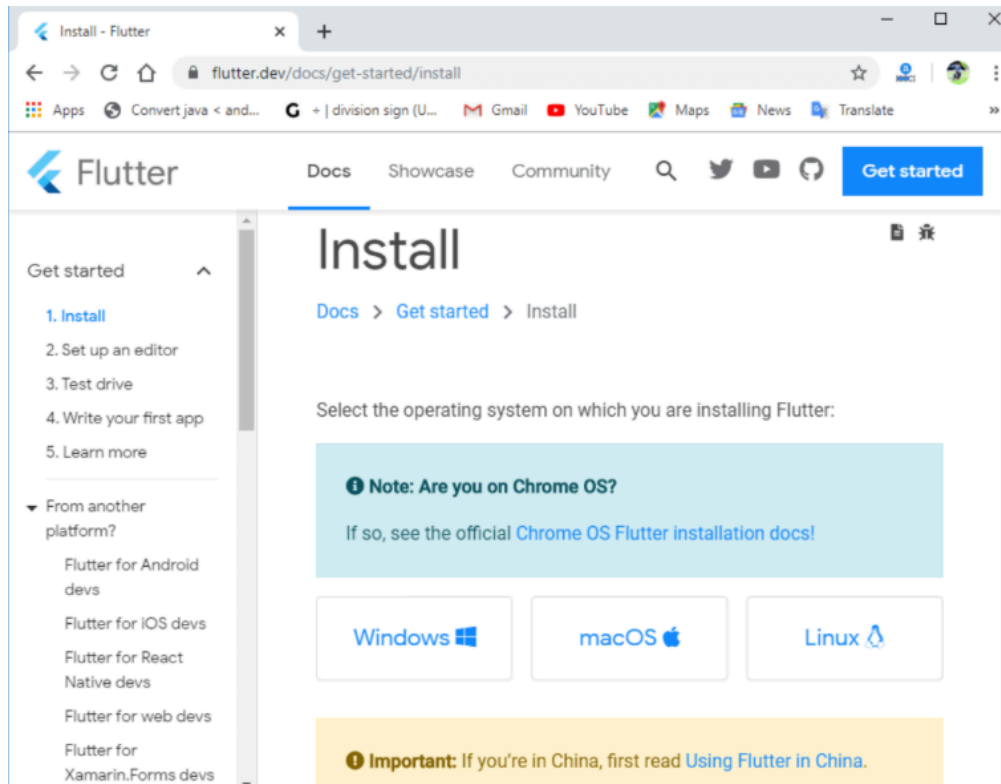
Experiment 1

Aim: Installation and Configuration of Flutter Environment.

Install the Flutter SDK

Step 1: Download the Flutter SDK for Windows

- Visit the official Flutter website at <https://docs.flutter.dev/get-started/install>. You'll see the Flutter installation page.



Step 2: Download the SDK

- Locate and click the Windows icon on the website to access the download link for the latest Flutter SDK.

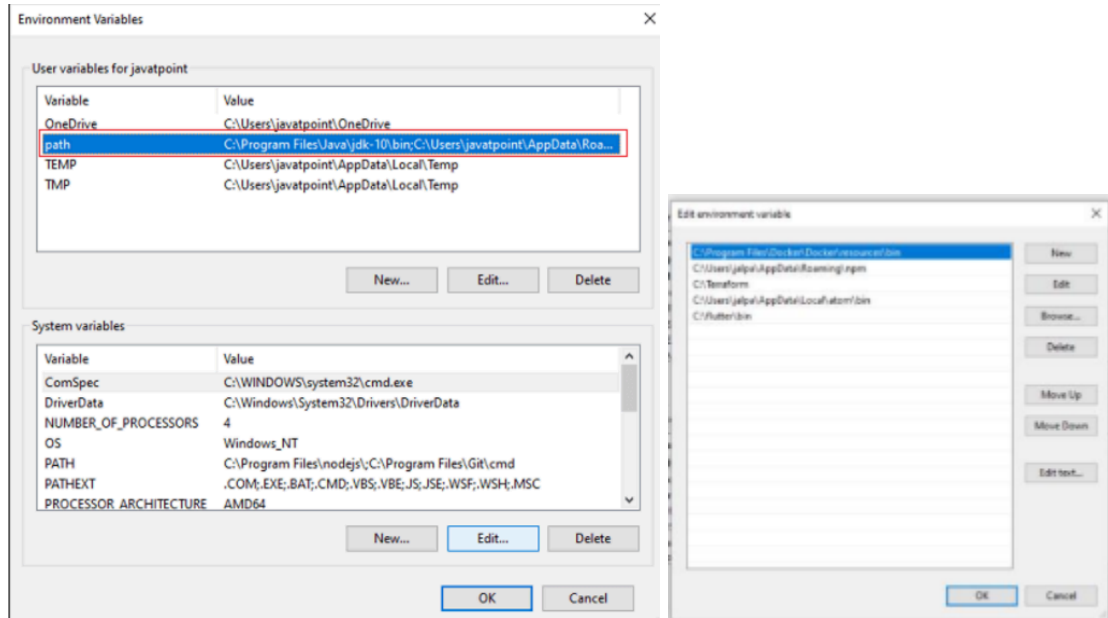
Step 3: Extract the SDK

- Once the download completes, extract the zip file to a preferred location, such as C:\Flutter.

Step 4: Update the System Path

- To run Flutter commands from the Windows Command Prompt, add the Flutter bin directory to your system's PATH environment variable. Follow these steps:

- Step 4.1: Right-click on This PC or My Computer, select Properties, go to the Advanced tab, and click Environment Variables.
- Step 4.2: In the Environment Variables window, find the Path variable under System Variables, select it, and click Edit.
- Step 4.3: Click New, add the path to the Flutter bin folder (e.g., C:\Flutter\bin), then click OK to save all changes.



Step 5: Verify Flutter Installation

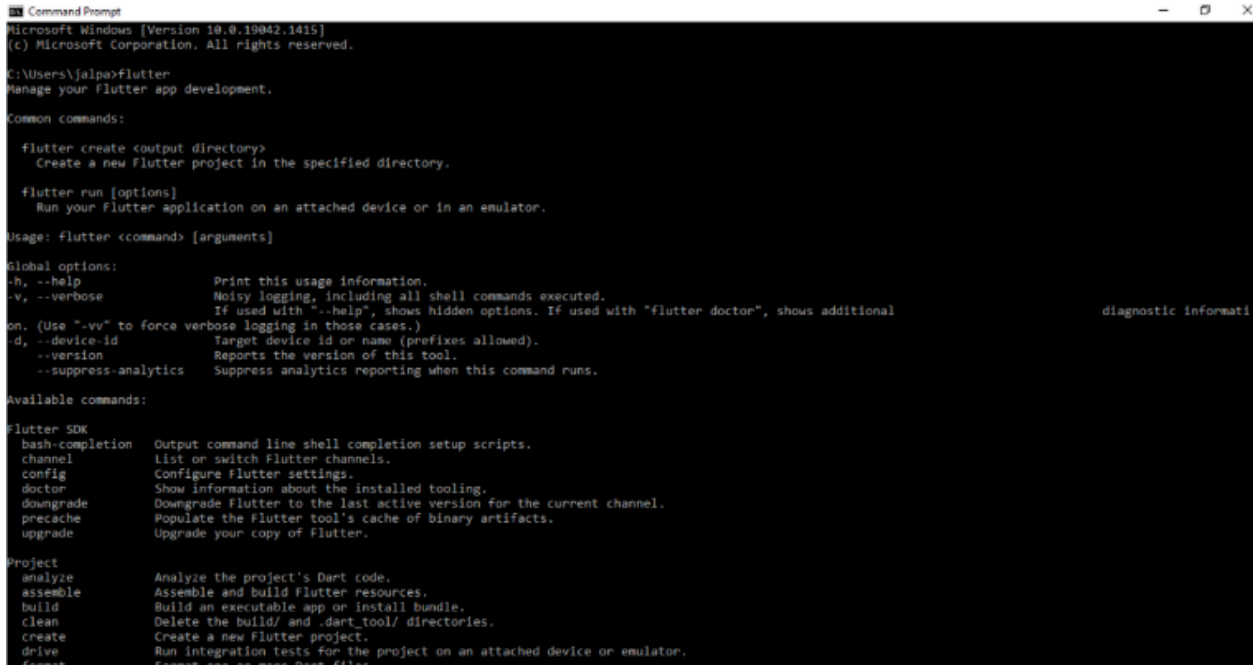
- Open a Command Prompt and run the following command:

flutter

Next, run:

flutter doctor

The flutter doctor command checks your system for Flutter development requirements and generates a report showing the status of your setup.



```

Microsoft Windows [Version 10.0.19042.1415]
(c) Microsoft Corporation. All rights reserved.

C:\Users\jalpa>flutter
Manage your Flutter app development.

Common commands:

Flutter create <output directory>
  Create a new Flutter project in the specified directory.

flutter run [options]
  Run your Flutter application on an attached device or in an emulator.

Usage: flutter <command> [arguments]

Global options:
-h, --help            Print this usage information.
-v, --verbose         Noisy logging, including all shell commands executed.
                     If used with "--help", shows hidden options. If used with "flutter doctor", shows additional diagnostic information.
on. (Use "--vv" to force verbose logging in those cases.)
-d, --device-id       Target device id or name (prefixes allowed).
--version             Reports the version of this tool.
--suppress-analytics  Suppress analytics reporting when this command runs.

Available commands:

Flutter SDK
bash-completion      Output command line shell completion setup scripts.
channel              List or switch Flutter channels.
config               Configure Flutter settings.
doctor               Show information about the installed tooling.
downgrade            Downgrade Flutter to the last active version for the current channel.
precache             Populate the Flutter tool's cache of binary artifacts.
upgrade              Upgrade your copy of Flutter.

Project
analyze              Analyze the project's Dart code.
assemble             Assemble and build Flutter resources.
build                Build an executable app or install bundle.
clean                Delete the build/ and .dart_tool/ directories.
create               Create a new Flutter project.
drive                Run integration tests for the project on an attached device or emulator.
format               Format one or more Dart files.

```

Step 6: Review the Flutter Doctor Report

- After running flutter doctor, you'll see a report detailing any missing tools or configurations needed for Flutter development, as well as the status of installed tools not yet connected to a device.

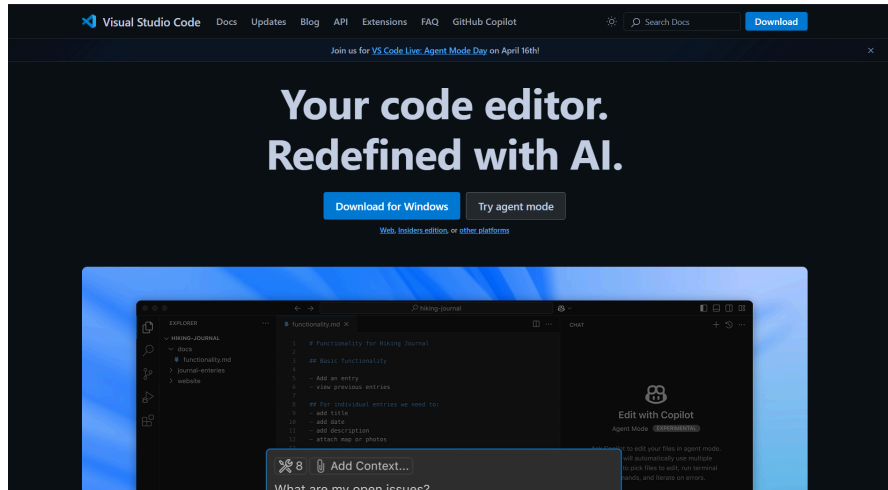
Install Required Tools

Step 7: Install Visual Studio Code and Android SDK

- If flutter doctor indicates that the Android SDK or a compatible IDE is missing, you'll need to install Visual Studio Code (VS Code) and configure the Android SDK manually. Follow these steps:

Step 7.1: Download Visual Studio Code

- Go to the official VS Code website (<https://code.visualstudio.com/>) and download the latest installer for Windows.



Step 7.2: Install VS Code

- Once downloaded, run the .exe file. Follow the installation wizard to complete the setup.

Step 7.3: Install Flutter and Dart Extensions

- Open VS Code, go to the Extensions view (Ctrl+Shift+X), and search for Flutter and Dart. Install both extensions to enable Flutter development support in VS Code.

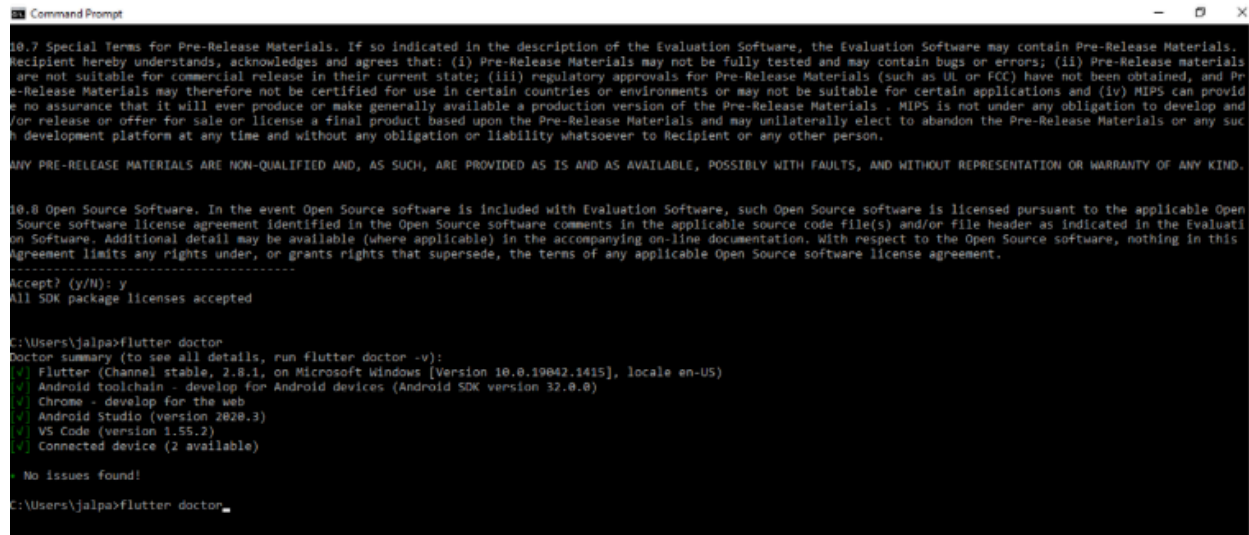
Step 7.4: Install the Android SDK



Step 7.6: Accept Android Licenses

- Run the following commands in the Command Prompt:
flutter doctor
flutter doctor --android-licenses

Follow the prompts to accept all Android SDK licenses.



```
Command Prompt

10.7 Special Terms for Pre-Release Materials. If so indicated in the description of the Evaluation Software, the Evaluation Software may contain Pre-Release Materials.
Recipient hereby understands, acknowledges and agrees that: (i) Pre-Release Materials may not be fully tested and may contain bugs or errors; (ii) Pre-Release materials
are not suitable for commercial release in their current state; (iii) regulatory approvals for Pre-Release Materials (such as UL or FCC) have not been obtained, and Pr
e-Release Materials may therefore not be certified for use in certain countries or environments or may not be suitable for certain applications and (iv) MIPS can provid
e no assurance that it will ever produce or make generally available a production version of the Pre-Release Materials . MIPS is not under any obligation to develop and
/or release or offer for sale or license a final product based upon the Pre-Release Materials and may unilaterally elect to abandon the Pre-Release Materials or any suc
h development platform at any time and without any obligation or liability whatsoever to Recipient or any other person.

ANY PRE-RELEASE MATERIALS ARE NON-QUALIFIED AND, AS SUCH, ARE PROVIDED AS IS AND AS AVAILABLE, POSSIBLY WITH FAULTS, AND WITHOUT REPRESENTATION OR WARRANTY OF ANY KIND.

10.8 Open Source Software. In the event Open Source software is included with Evaluation Software, such Open Source software is licensed pursuant to the applicable Open
Source software license agreement identified in the Open Source software comments in the applicable source code file(s) and/or file header as indicated in the Evaluati
on Software. Additional detail may be available (where applicable) in the accompanying on-line documentation. With respect to the Open Source software, nothing in this
agreement limits any rights under, or grants rights that supersede, the terms of any applicable Open Source software license agreement.

Accept? (y/N): y
All SDK package licenses accepted

C:\Users\jalpa>flutter doctor

Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 2.8.1, on Microsoft Windows [Version 10.0.19042.1415], locale en-US)
[✓] Android toolchain - develop for Android devices (Android SDK version 32.0.0)
[✓] Chrome - develop for the web
[✓] Android Studio (version 2020.3)
[✓] VS Code (version 1.55.2)
[✓] Connected device (2 available)

No issues found!

C:\Users\jalpa>flutter doctor_
```

Step 8: Enable Flutter Web Support and Run on Chrome

- To test your Flutter app in a web browser like Chrome, you need to enable Flutter's web support and configure VS Code to run the app in Chrome. Follow these steps:

Step 8.1: Enable Web Support in Flutter

- Ensure your Flutter SDK is up to date and supports web development. Open a Command Prompt and run:
flutter channel stable
flutter upgrade
- This ensures you're on the stable channel with the latest Flutter version.
- Enable web support by running:
flutter config --enable-web

This command enables the web renderer for Flutter, allowing you to target Chrome as a runtime environment.

Step 8.2: Create or Open a Flutter Project

- If you don't already have a Flutter project, create one by running:
flutter create my_app
cd my_app
- Replace my_app with your desired project name.
- Open the project in VS Code:
- Launch VS Code.
- Go to File > Open Folder and select the project folder (e.g., my_app).

Step 8.3: Configure VS Code to Run on Chrome

- Ensure the Flutter and Dart extensions are installed in VS Code (as described in the previous response, Step 7.3).
- Open the pubspec.yaml file in your project and ensure it includes the Flutter web dependencies. Most Flutter projects created after web support became stable (Flutter 2.0+) include these by default. If not, you don't need to modify anything manually unless you're using specific web-related packages.
- In VS Code, go to the Run and Debug panel (Ctrl+Shift+D).
- Click create a launch.json file (if it doesn't exist) and select Dart & Flutter as the environment.
- Modify the launch.json file in the .vscode folder to include a configuration for web, like this:

```
{
  "version": "0.2.0",
  "configurations": [
    {
      "name": "Run on Chrome",
      "type": "dart",
      "request": "launch",
      "program": "lib/main.dart",
      "deviceId": "web-server",
      "args": ["--web-renderer", "auto"]
    }
  ]
}
```

This sets up VS Code to run your app on Chrome via Flutter's web server.

Step 8.4: Run the App on Chrome

In VS Code, go to the Device Selector (bottom-right corner) and select Chrome (web-javascript) or Web Server from the list of available devices. If Chrome doesn't appear, ensure web support is enabled (Step 8.1) and Chrome is installed.

Open the terminal in VS Code (Ctrl+`) and run:

```
bash
```

```
flutter run
```

Alternatively, press F5 or click Run > Start Debugging with the "Run on Chrome" configuration selected in the Run and Debug panel.

Flutter will build the app for the web and launch Chrome automatically, displaying your app at a URL like http://localhost:port (e.g., <http://localhost:53599>).

Conclusion:

The successful installation and configuration of the Flutter environment, as outlined in this experiment, enable developers to create and test cross-platform applications efficiently. By setting up the Flutter SDK, integrating Visual Studio Code with Flutter and Dart extensions, configuring the Android SDK, and enabling web support to run the app on Chrome, a robust development environment is established. This setup eliminates the need for physical devices or emulators during initial testing, streamlining the development process and allowing for rapid iteration and debugging of Flutter applications directly in a web browser.