Department of Information Engineering Technology
The Superior University, Lahore

Mobile Application Development Lab

Experiment No.1
Installation and Execution of Flutter

Prepared for

EISHA NAWAZ

By:
Name: ____________

ID: _______________

Section: ___________

Semester: __________

Total Marks: 10

Obtained Marks: ______________

Signature: __________________

Date: _____________

Experiment No.1

Installation and Execution of Flutter

(Rubrics)

Name: ________________________________ **Roll No:** ____________________

A. PSYCHOMOTOR

Sr. No.	Criteria	Allocated Marks	Unacceptable 0%	Poor 25%	Fair 50%	Good 75%	Excellent 100%	Total Obtained
1	Follow Procedures	1	0	0.25	0.5	0.75	1	
2	Software and Simulations	2	0	0.5	1	1.5	2	
3	Accuracy in Output Results	3	0	0.75	1.5	2.25	3	
Sub Total		**6**	**Sub Total marks Obtained in Psychomotor(P)**					

B. AFFECTIVE

Sr. No.	Criteria	Allocated Marks	Unacceptable 0%	Poor 25%	Fair 50%	Good 75%	Excellent 100%	Total Obtained
1	Respond to Questions	1	0	0.25	0.5	0.75	1	
2	Lab Report	1	0	0.25	0.5	0.75	1	
3	Assigned Task	2	0	0.5	1	1.5	2	
Sub Total		**4**	**Sub Total marks Obtained in Affective (A)**					

Instructor Name: EISHA NAWAZ ____________ **Total Marks (P+A):** ____10________

Instructor Signature:________________________ **Obtained Marks (P+A):** ___________

1.6 Configure a text editor or IDE

You can build apps with Flutter using any text editor or integrated development environment (IDE) combined with Flutter's command-line tools.

Using an IDE with a Flutter extension or plugin provides code completion, syntax highlighting, widget editing assists, debugging, and other features.

Popular options include:

- Visual Studio Code 1.77 or later with the Flutter extension for VS Code.
- Android Studio 2023.3.1 (Jellyfish) or later with the Flutter plugin for IntelliJ.
- IntelliJ IDEA 2023.3 or later with the Flutter plugin for IntelliJ.

1.7 Install the Flutter SDK

To install the Flutter SDK, you can use the VS Code Flutter extension or download and install the Flutter bundle yourself.

Step 1: Download Flutter SDK:

Download the following installation bundle to get the latest stable release of the Flutter SDK

Step 2: Extract the File: Extract the downloaded zip file and move it to the desired location where you want to install Flutter SDK.

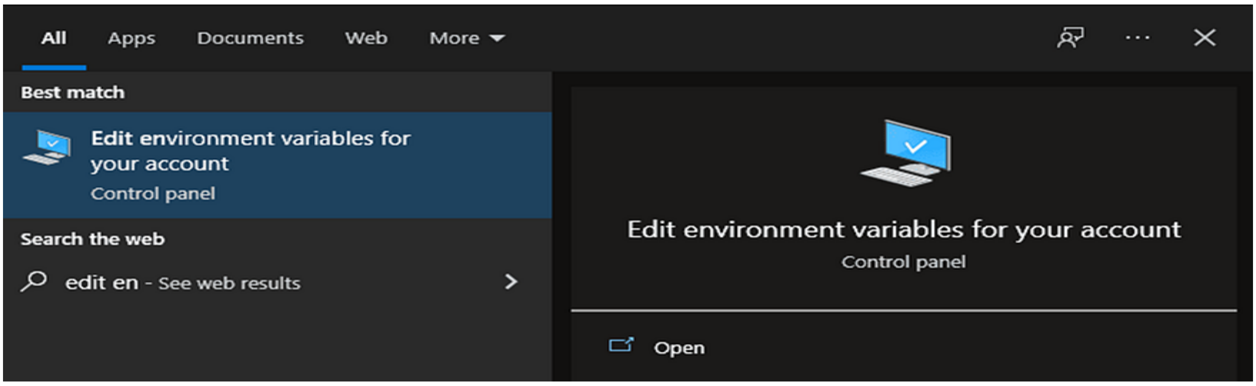
Do not install it in a folder or directory that requires elevated privileges, (such as C:\Program Files\) to ensure the program runs properly. For this tutorial, it will be stored in C:\development\flutter.

You are now ready to run Flutter commands in the Flutter Console.

Step 3: Update Path Variable for Windows PowerShell

If you wish to run Flutter commands in the regular Windows console, take these steps to add Flutter to the PATH environment variable:

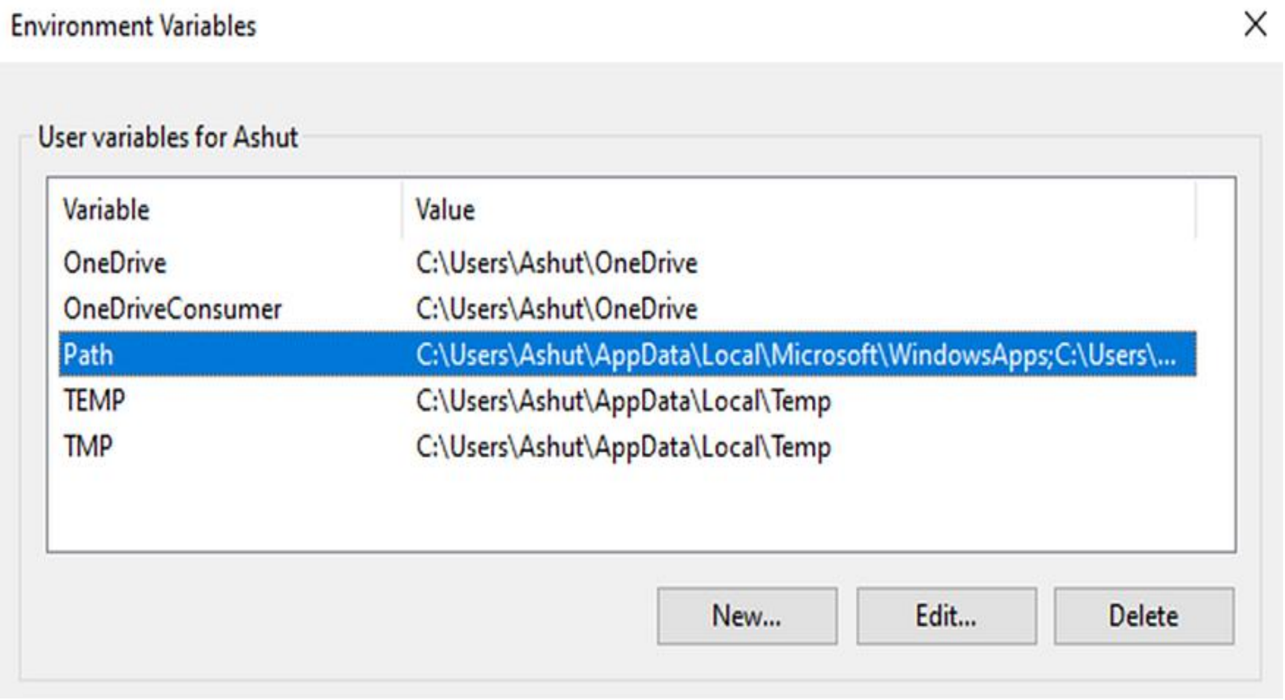
From the Start search bar, enter 'env' and select Edit environment variables for your account.


All
Apps
Documents
Web
More
Best match
Edit environment variables for your account
Control panel
Search the web
edit en - See web results
Edit environment variables for your account
Control panel
Open

Press enter or click to view image in full size

Under User variables check if there is an entry called Path:

If the entry exists, append the full path to flutter\bin using; as a separator from existing values.



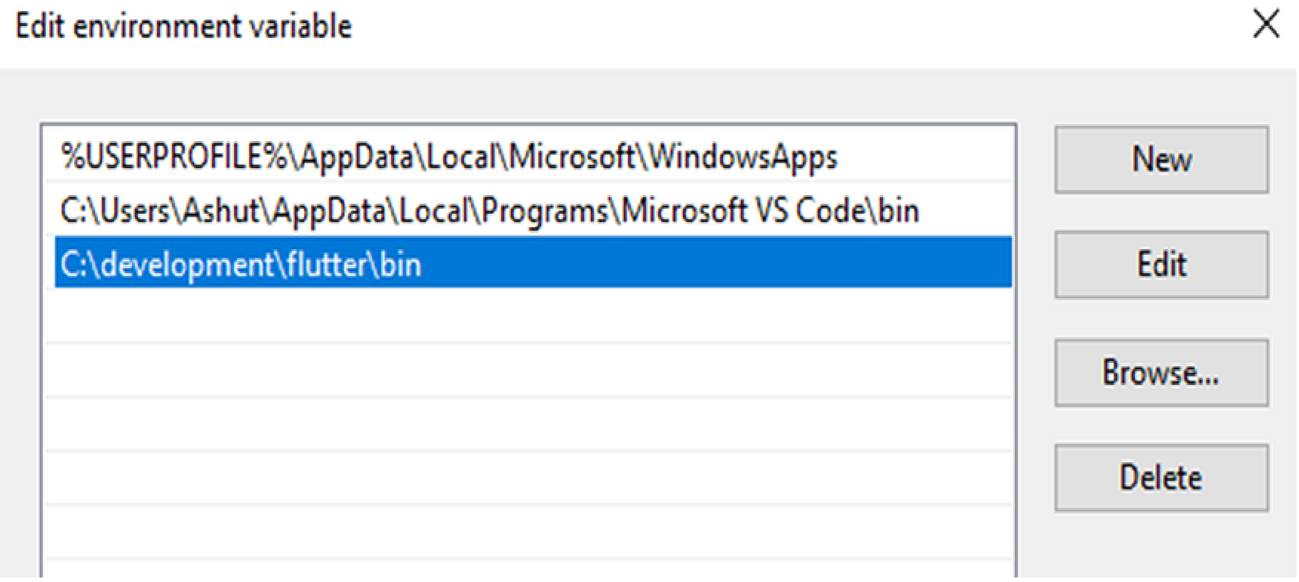
Environment Variables

User variables for Ashut

Variable	Value
OneDrive	C:\Users\Ashut\OneDrive
OneDriveConsumer	C:\Users\Ashut\OneDrive
Path	C:\Users\Ashut\AppData\Local\Microsoft\WindowsApps;C:\Users\...
TEMP	C:\Users\Ashut\AppData\Local\Temp
TMP	C:\Users\Ashut\AppData\Local\Temp

Press enter or click to view image in full size

On the next screen, click New and add the full path to your flutter\bin directory. For this guide, it is shown below. Click OK on both windows to enable running Flutter commands in Windows consoles.



Edit environment variable

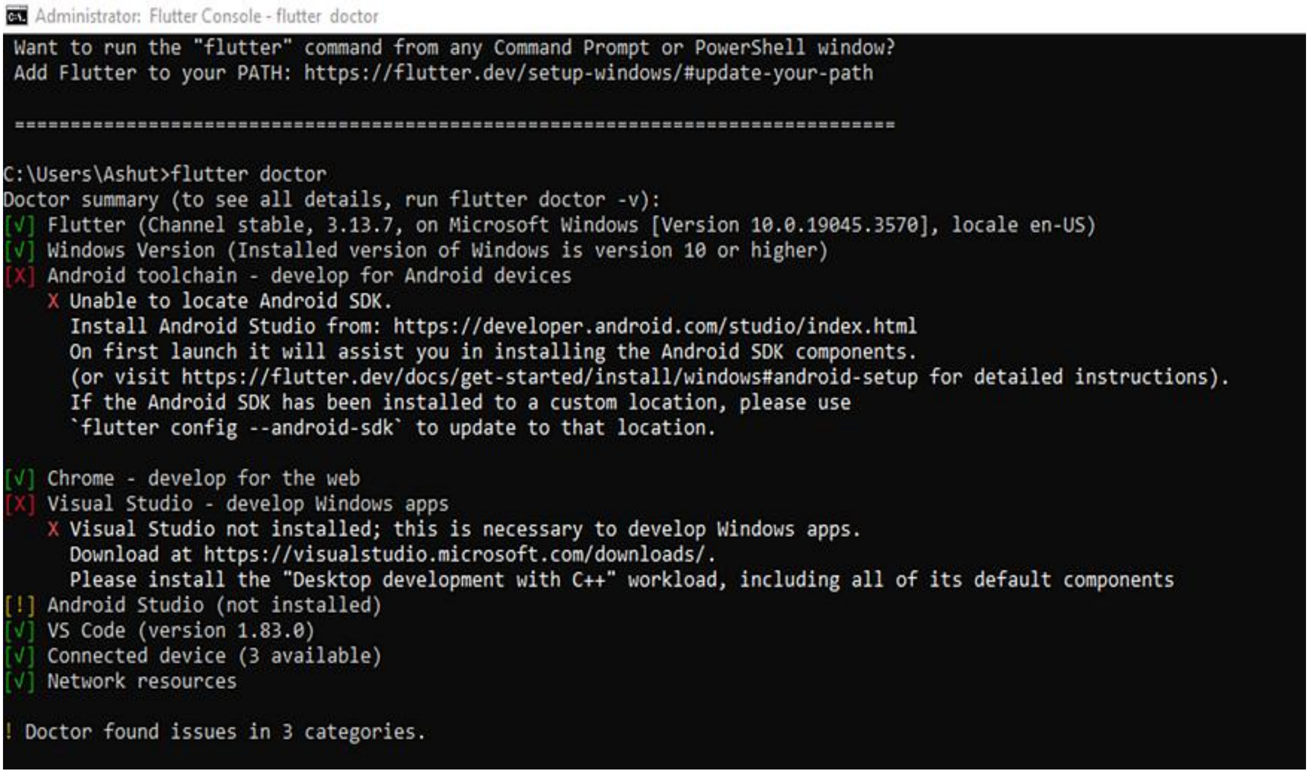
```
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
C:\Users\Ashut\AppData\Local\Programs\Microsoft VS Code\bin
C:\development\flutter\bin
```

Press enter or click to view image in full size

If the entry doesn’t exist, create a new user variable named Path with the full path to flutter\bin as its value.

Step 4: Confirm Installed Tools for Running Flutter

In CMD, run the flutter doctor command to confirm the installed tools along with brief descriptions.



```
 Want to run the "flutter" command from any Command Prompt or PowerShell window?
 Add Flutter to your PATH: https://flutter.dev/setup-windows/#update-your-path

 ================================================================================

C:\Users\Ashut>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.13.7, on Microsoft Windows [Version 10.0.19045.3570], locale en-US)
[√] Windows Version (Installed version of Windows is version 10 or higher)
[X] Android toolchain - develop for Android devices
    X Unable to locate Android SDK.
      Install Android Studio from: https://developer.android.com/studio/index.html
      On first launch it will assist you in installing the Android SDK components.
      (or visit https://flutter.dev/docs/get-started/install/windows#android-setup for detailed instructions).
      If the Android SDK has been installed to a custom location, please use
      `flutter config --android-sdk` to update to that location.

[√] Chrome - develop for the web
[X] Visual Studio - develop Windows apps
    X Visual Studio not installed; this is necessary to develop Windows apps.
      Download at https://visualstudio.microsoft.com/downloads/.
      Please install the "Desktop development with C++" workload, including all of its default components
[!] Android Studio (not installed)
[√] VS Code (version 1.83.0)
[√] Connected device (3 available)
[√] Network resources

! Doctor found issues in 3 categories.
```

Press enter or click to view image in full size

As visible, several components still need to be installed to complete the installation.

2.1. Use VS Code to install Flutter

To install Flutter using these instructions, verify that you have installed Visual Studio Code 1.77 or later and the Flutter extension for VS Code.

2.1.1. Prompt VS Code to install Flutter

- Launch VS Code.
- To open the **Command Palette**, press **Control** + **Shift** + **P**.
- In the **Command Palette**, type flutter.
- Select **Flutter: New Project**.
- VS Code prompts you to locate the Flutter SDK on your computer.
    - If you have the Flutter SDK installed, click **Locate SDK**.
    - If you do not have the Flutter SDK installed, click **Download SDK**.

    - This option sends you the Flutter install page if you have not installed Git for Windows as directed in the development tools prerequisites.
- When prompted Which Flutter template? ignore it. Press Esc. You can create a test project after checking your development setup.

2.1.2. Download the Flutter SDK

When the Select Folder for Flutter SDK dialog displays, choose where you want to install Flutter.

VS Code places you in your user profile to start. Choose a different location.

Consider %USERPROFILE% or C:\dev.

Warning
Don't install Flutter to a directory or path that meets one or both of the following conditions:

The path contains special characters or spaces.
The path requires elevated privileges.
As an example, C:\Program Files fails both conditions.

1. Click Clone Flutter.

While downloading Flutter, VS Code displays this pop-up notification:

```
Downloading the Flutter SDK. This may take a few minutes.
```

This download takes a few minutes. If you suspect that the download has hung, click Cancel then start the installation again.

2. Once it finishes downloading Flutter, the Output panel displays.

```
Checking Dart SDK version...
Downloading Dart SDK from the Flutter engine ...
Expanding downloaded archive...
```

When successful, VS Code displays this pop-up notification:

```
Initializing the Flutter SDK. This may take a few minutes.
```

While initializing, the **Output** panel displays the following:

```
Building flutter tool...
Running pub upgrade...
Resolving dependencies...
Got dependencies.
Downloading Material fonts...
Downloading Gradle Wrapper...
Downloading package sky_engine...
Downloading flutter_patched_sdk tools...
Downloading flutter_patched_sdk_product tools...
Downloading windows-x64 tools...
Downloading windows-x64/font-subset tools...
```

This process also runs flutter doctor -v. At this point in the procedure, ignore this output. Flutter Doctor might show errors that don't apply to this quick start.

When the Flutter install succeeds, VS Code displays this pop-up notification:

```
Do you want to add the Flutter SDK to PATH so it's accessible
```

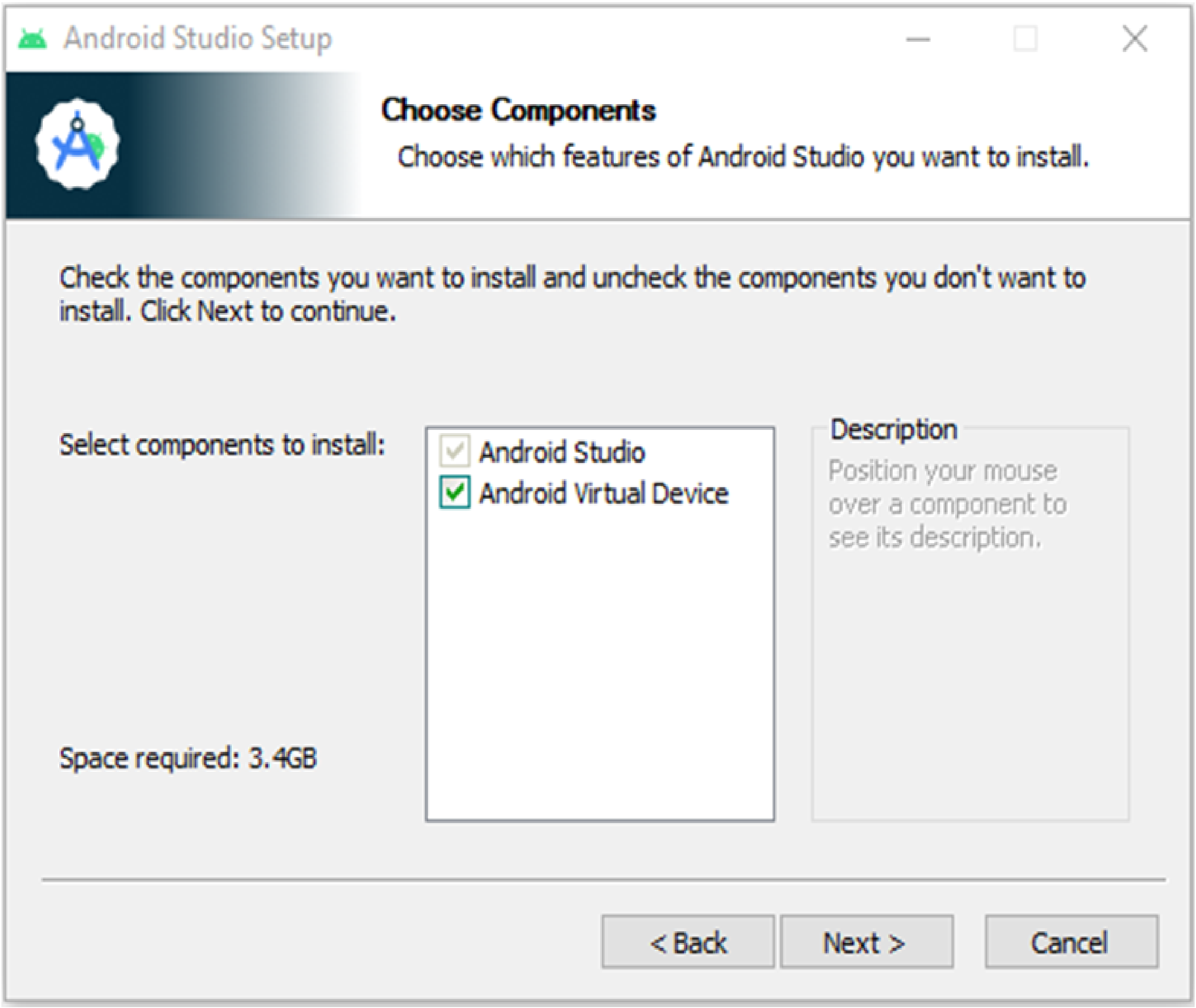
2.2.2. Configure your target Android device

Download Android Studio:

- ***Visit the official Android Studio download page at https://developer.android.com/studio.***
- ***Click on the "Download Android Studio" button.***

Next, proceed by downloading Android Studio. During the setup, unless you have unique requirements, simply click "Next" on all screens to keep the default settings. On the "Choose Components" screen, be sure to select the "Android Virtual Device" option to enable an Android emulator for your app development needs.

Press enter or click to view image in full size



Android Studio Setup

Choose Components

Choose which features of Android Studio you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

☑ Android Studio
☑ Android Virtual Device

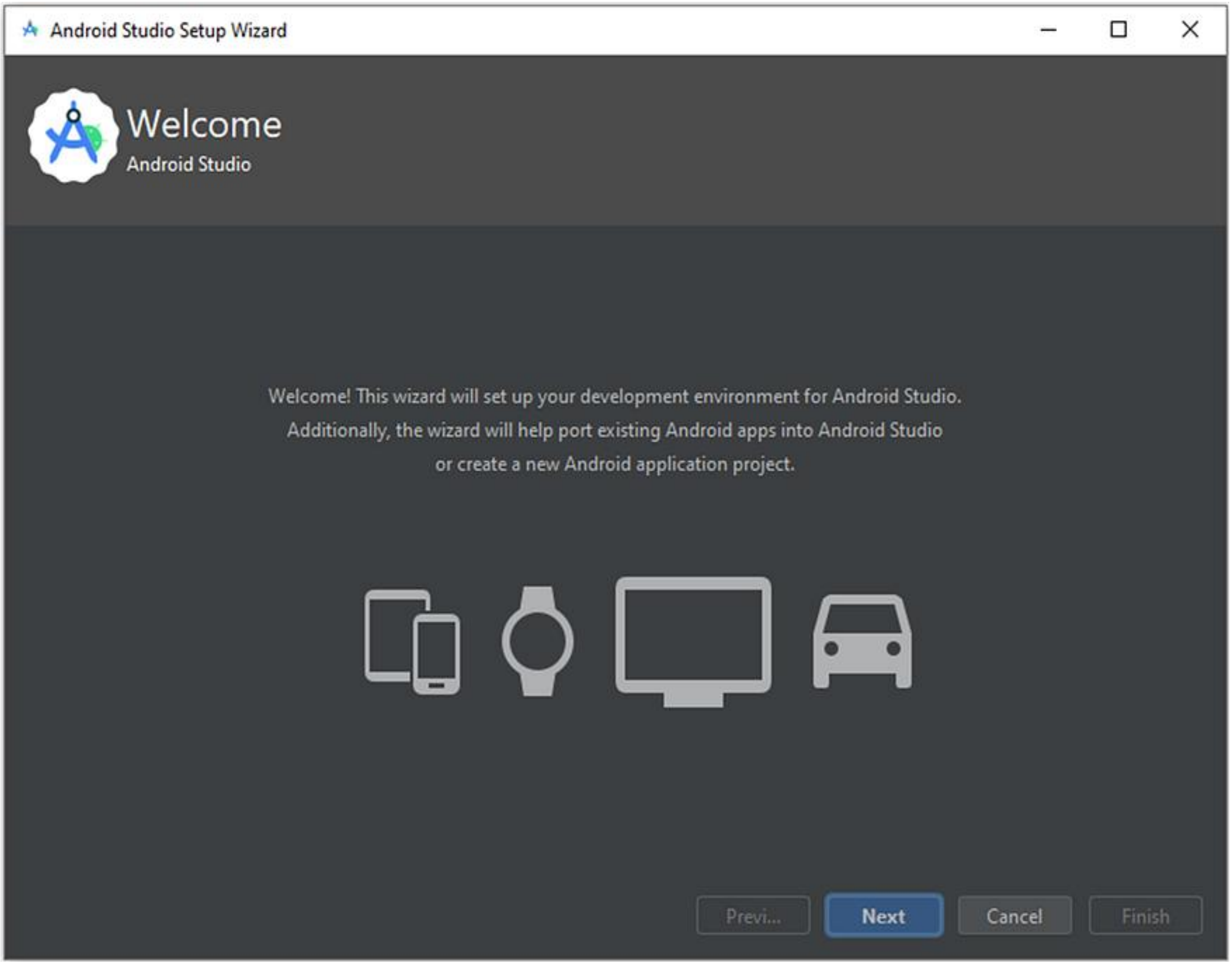
Space required: 3.4GB

Description

Position your mouse over a component to see its description.

Afterward, The Android Studio Setup Wizard will start and you can proceed by clicking Next.

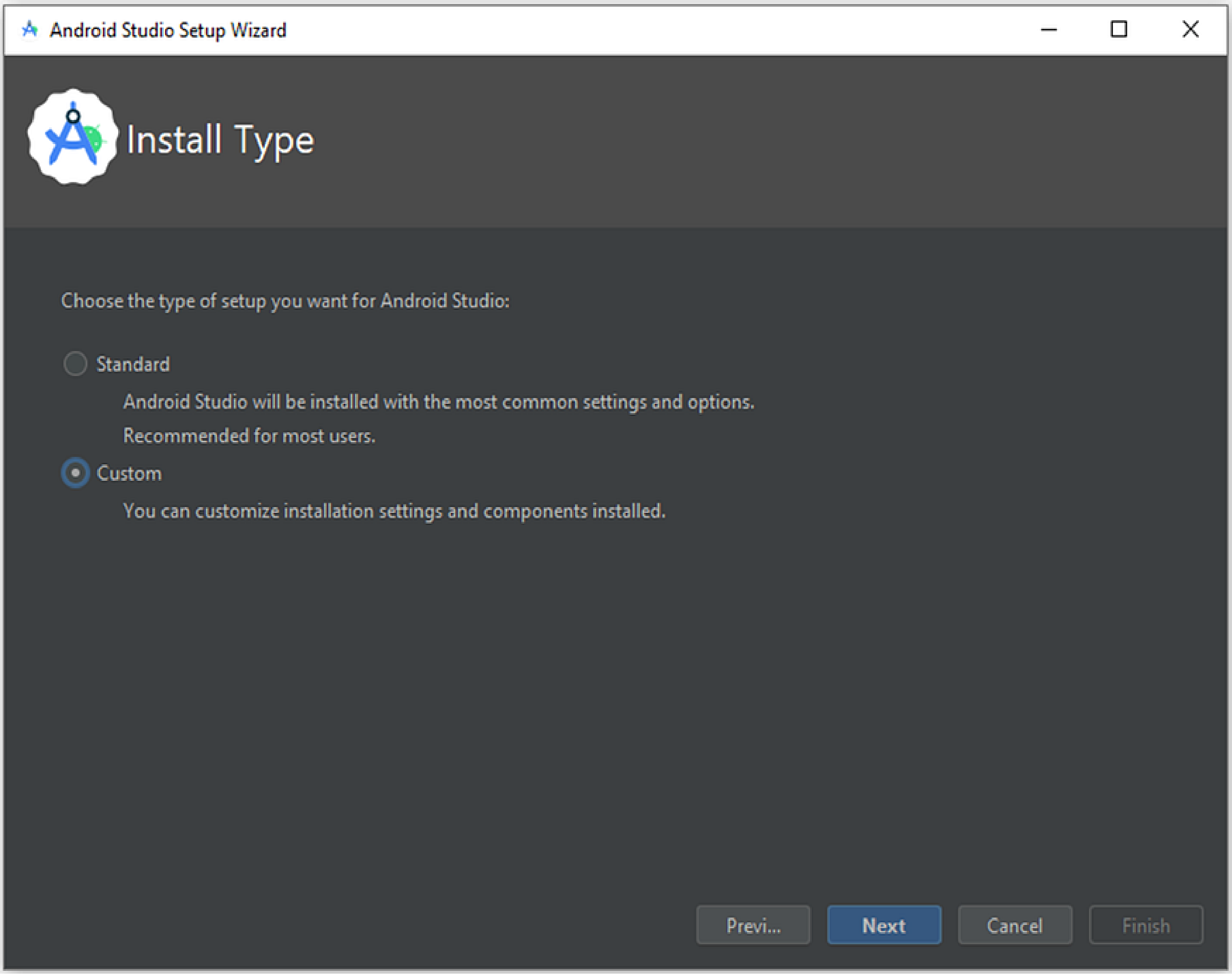
Press enter or click to view image in full size



Welcome! This wizard will set up your development environment for Android Studio. Additionally, the wizard will help port existing Android apps into Android Studio or create a new Android application project.

On the Install Type screen, select Custom and click Next

Press enter or click to view image in full size



Install Type

Choose the type of setup you want for Android Studio:

○ Standard

Android Studio will be installed with the most common settings and options.

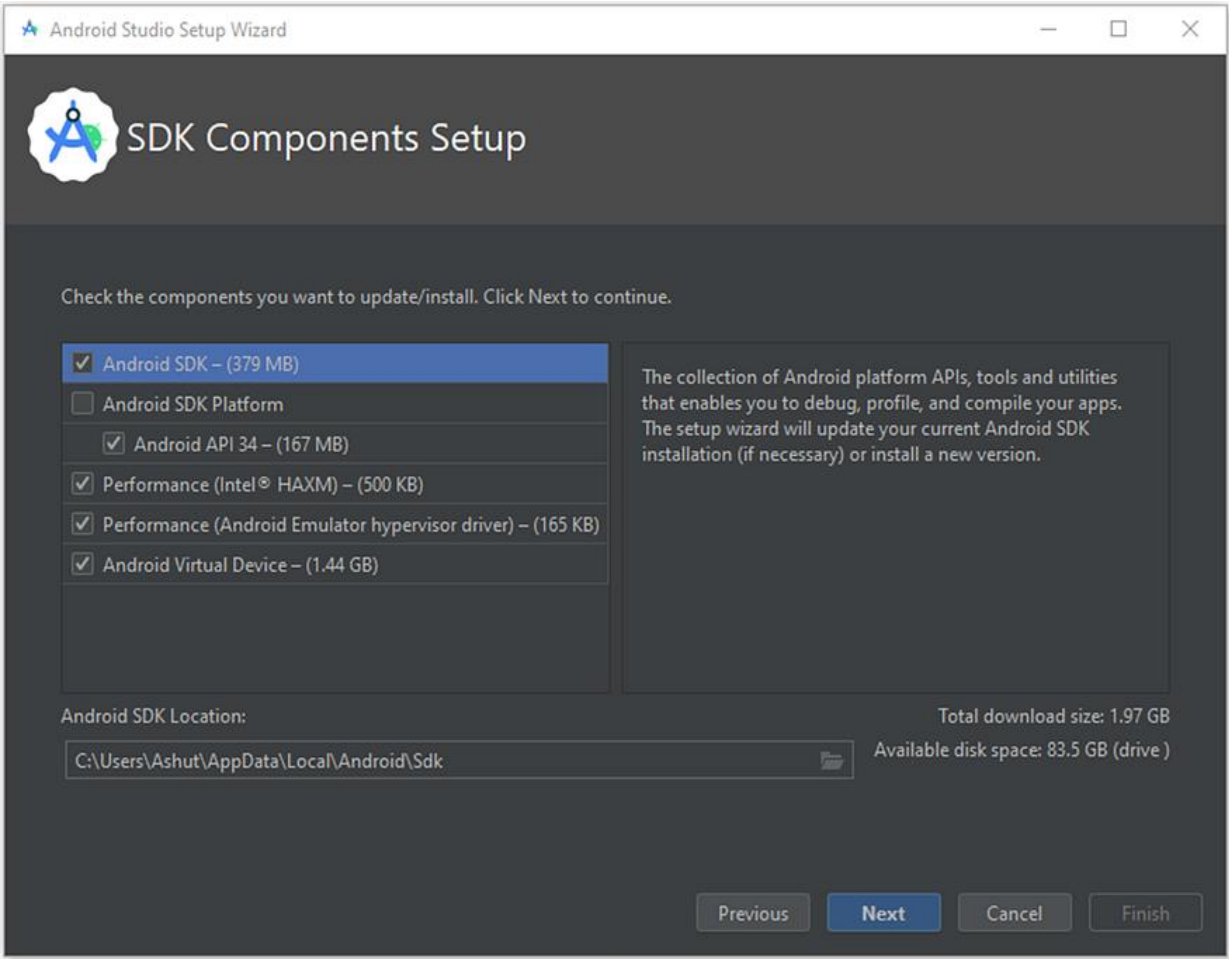
Recommended for most users.

● Custom

You can customize installation settings and components installed.

Select the installation location or leave the default path and click Next.

Press enter or click to view image in full size



SDK Components Setup

Check the components you want to update/install. Click Next to continue.

- ☑ Android SDK – (379 MB)
- □ Android SDK Platform
  - ☑ Android API 34 – (167 MB)
- ☑ Performance (Intel® HAXM) – (500 KB)
- ☑ Performance (Android Emulator hypervisor driver) – (165 KB)
- ☑ Android Virtual Device – (1.44 GB)

The collection of Android platform APIs, tools and utilities that enables you to debug, profile, and compile your apps. The setup wizard will update your current Android SDK installation (if necessary) or install a new version.

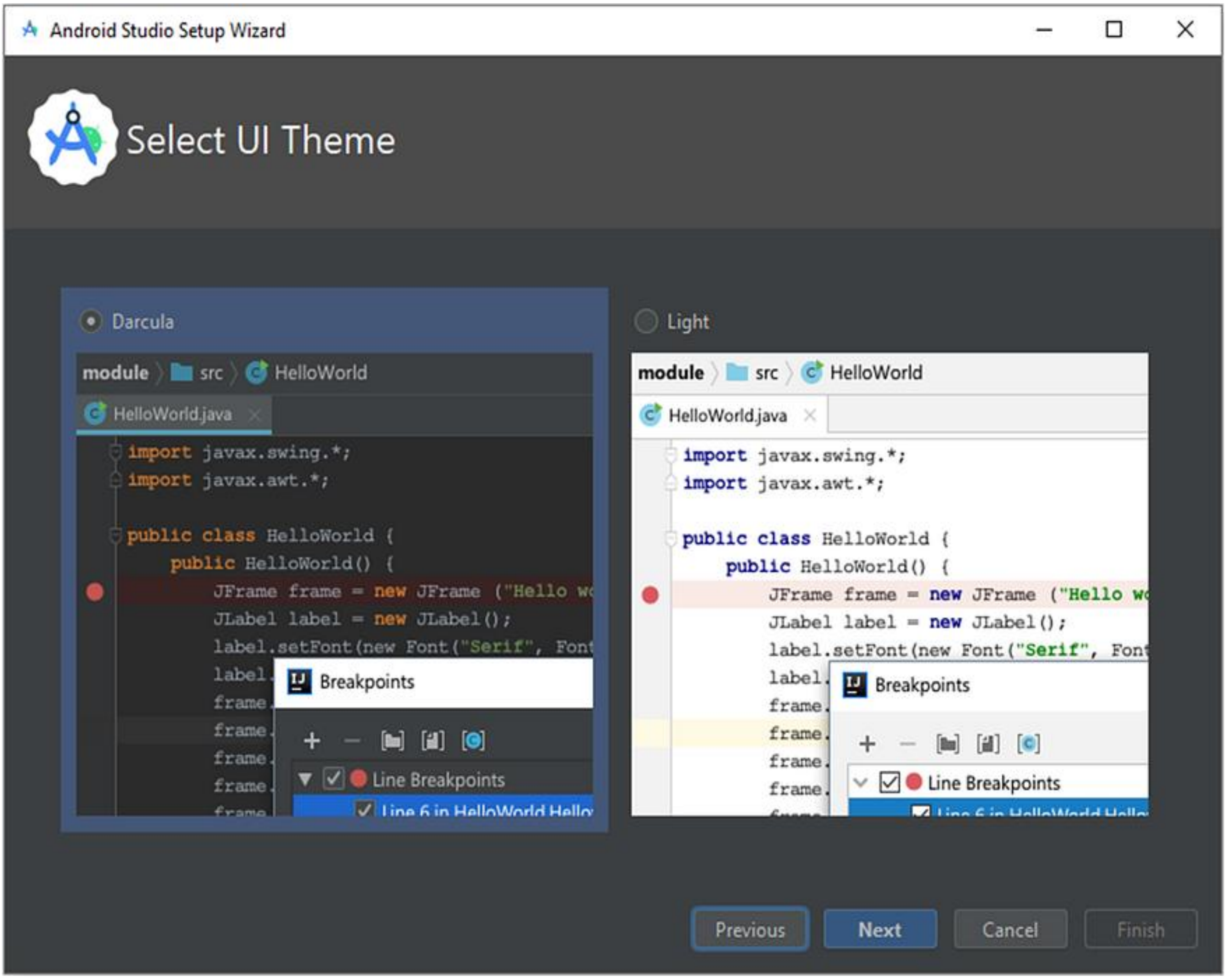
Android SDK Location: C:\Users\Ashut\AppData\Local\Android\Sdk

Total download size: 1.97 GB

Available disk space: 83.5 GB (drive)

Select your UI theme and click Next.

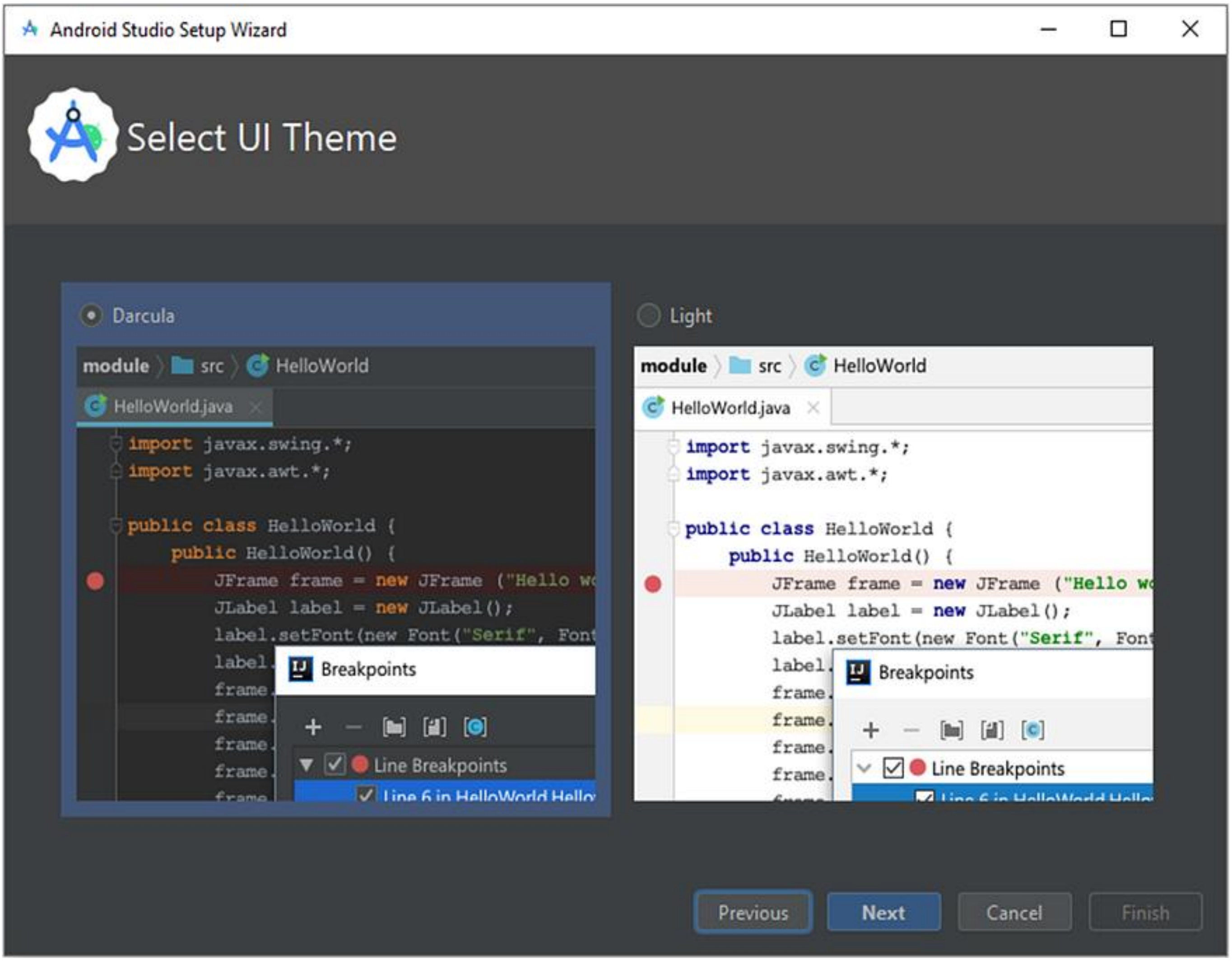
Press enter or click to view image in full size



```
import javax.swing.*;
import javax.awt.*;

public class HelloWorld {
    public HelloWorld() {
        JFrame frame = new JFrame ("Hello wo
        JLabel label = new JLabel();
        label.setFont(new Font("Serif", Font
        label.
        frame.
        frame.
        frame.
        frame.
```

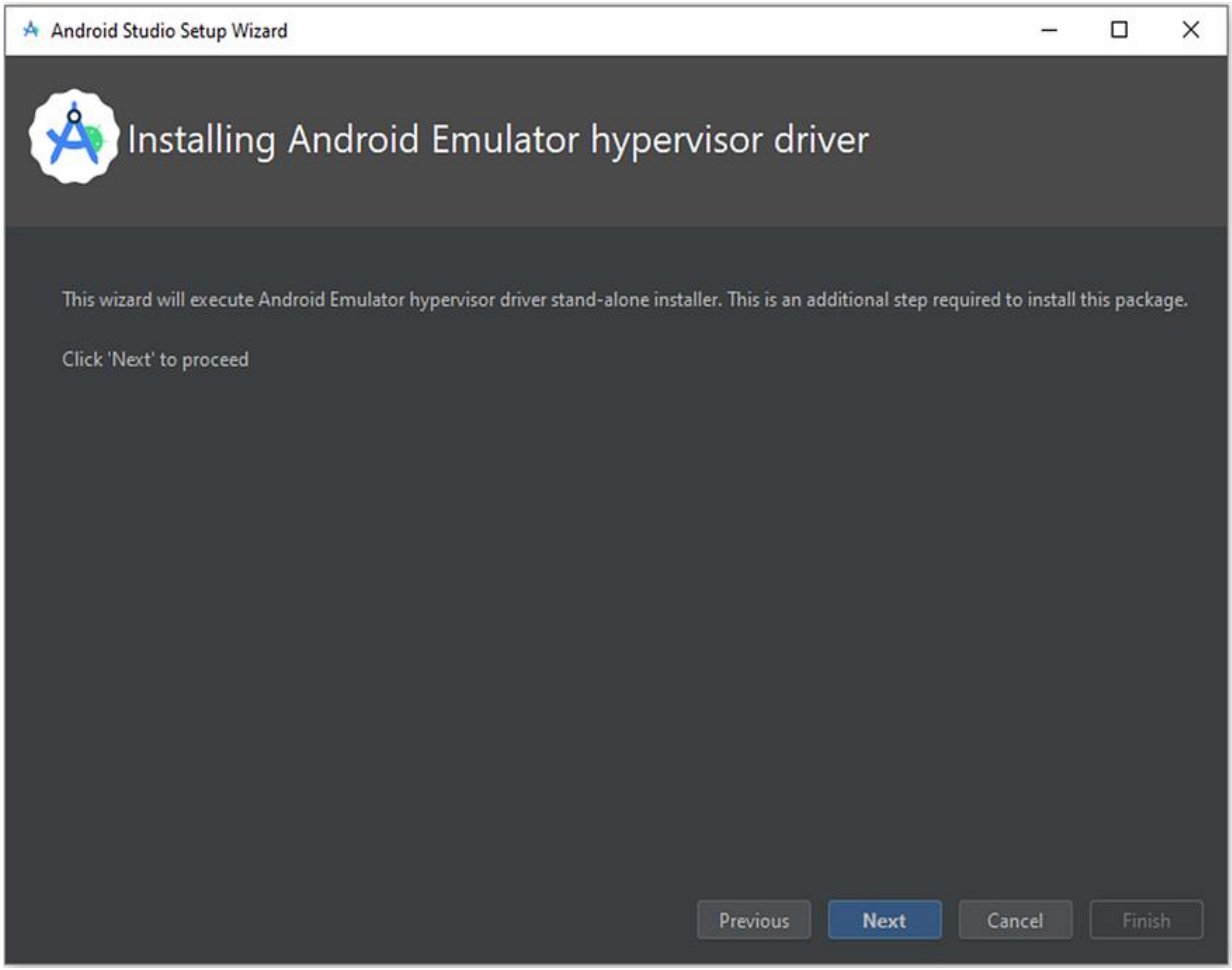
Press enter or click to view image in full size



```
import javax.swing.*;
import javax.awt.*;

public class HelloWorld {
    public HelloWorld() {
        JFrame frame = new JFrame ("Hello wo
        JLabel label = new JLabel();
        label.setFont(new Font("Serif", Font
        label.
        frame.
        frame.
        frame.
        frame.
```

Press enter or click to view image in full size



Android Studio Setup Wizard

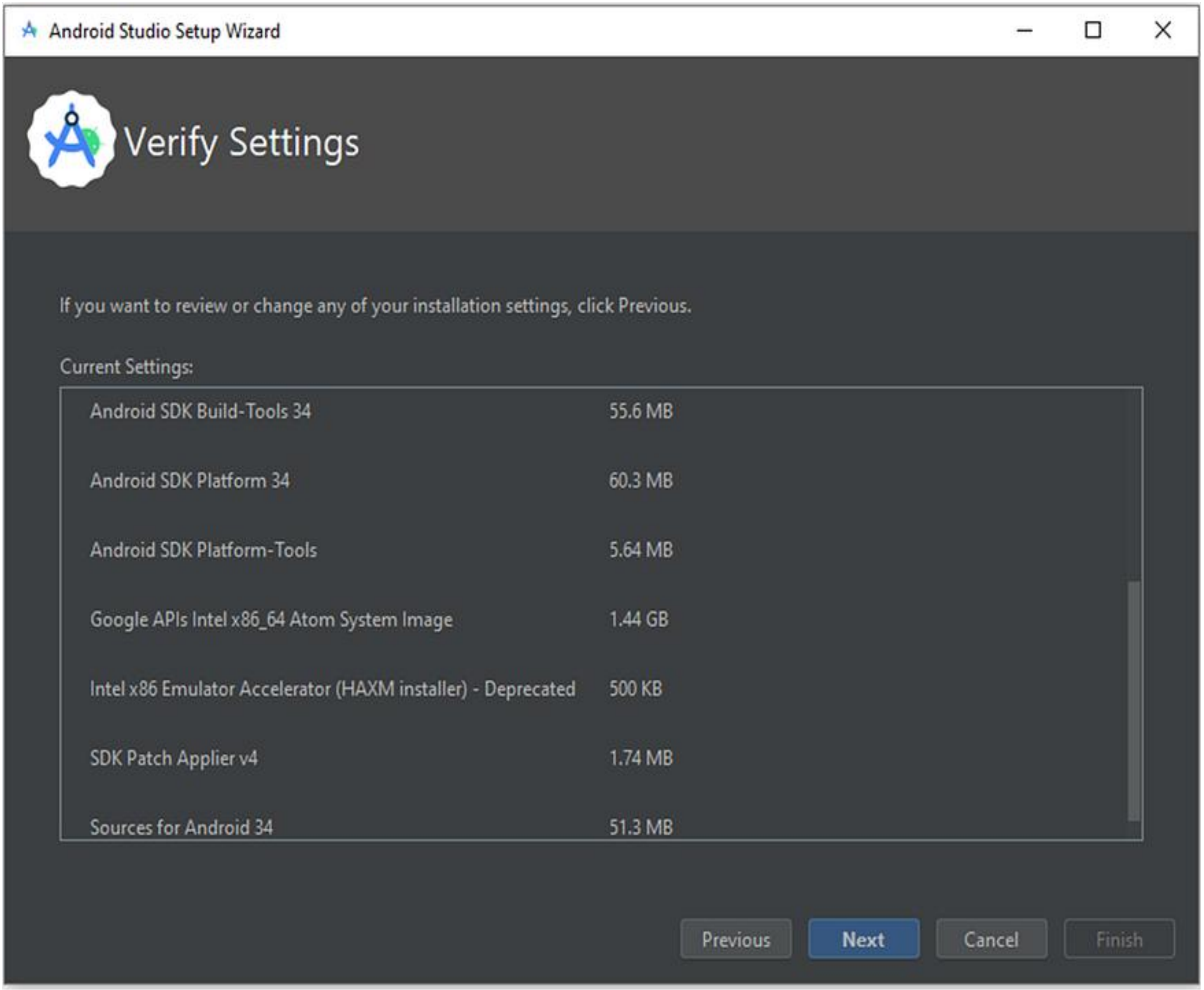
Installing Android Emulator hypervisor driver

This wizard will execute Android Emulator hypervisor driver stand-alone installer. This is an additional step required to install this package.

Click 'Next' to proceed

Verify the selections and click Next.

Press enter or click to view image in full size



Verify Settings

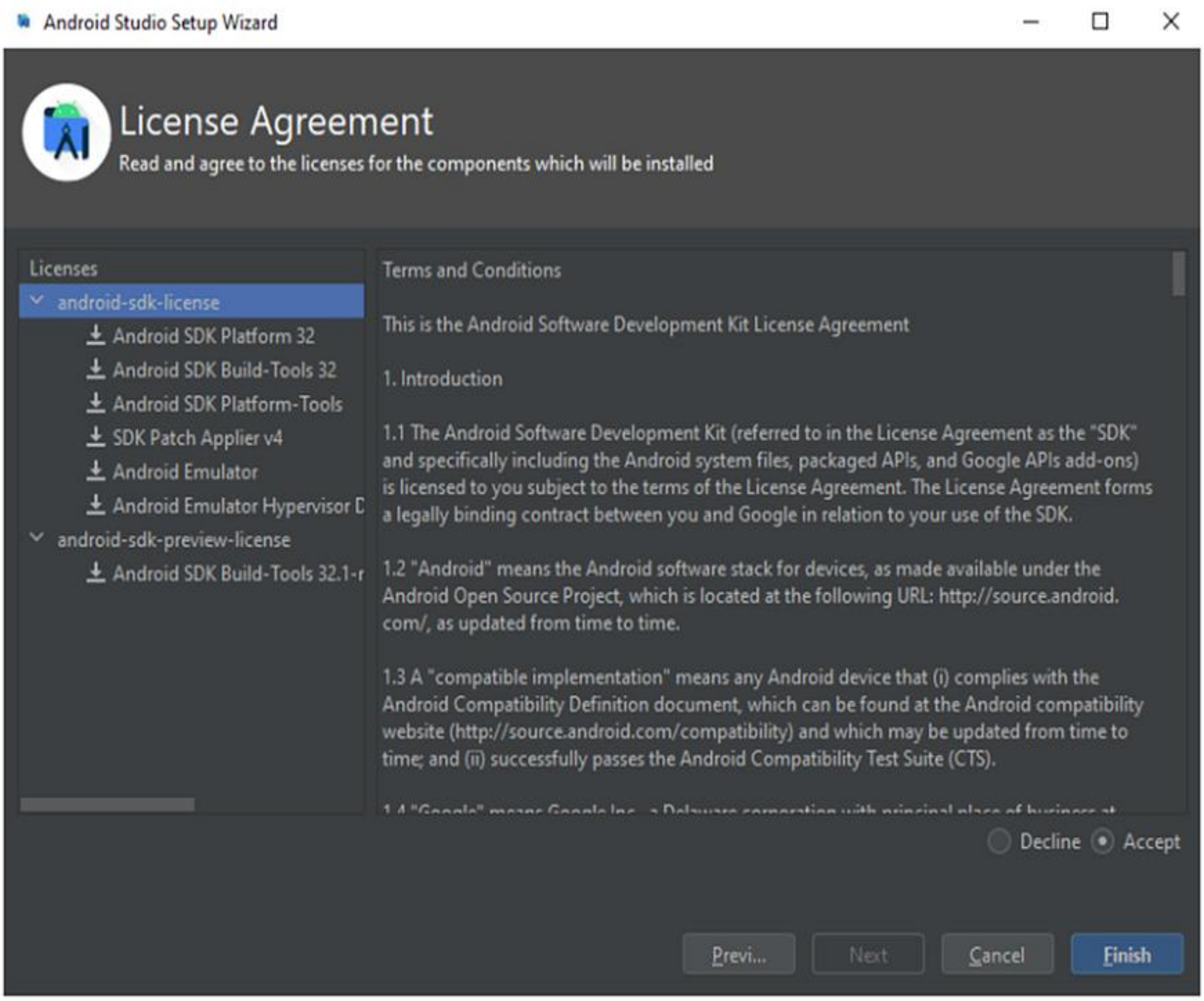
If you want to review or change any of your installation settings, click Previous.

Current Settings:

Android SDK Build-Tools 34	55.6 MB
Android SDK Platform 34	60.3 MB
Android SDK Platform-Tools	5.64 MB
Google APIs Intel x86_64 Atom System Image	1.44 GB
Intel x86 Emulator Accelerator (HAXM installer) - Deprecated	500 KB
SDK Patch Applier v4	1.74 MB
Sources for Android 34	51.3 MB

On the next screen, accept the License Agreement and click Finish.

Press enter or click to view image in full size


Android Studio Setup Wizard
Licenses
android-sdk-license
Android SDK Platform 32
Android SDK Build-Tools 32
Android SDK Platform-Tools
SDK Patch Applier v4
Android Emulator
Android Emulator Hypervisor D
android-sdk-preview-license
Android SDK Build-Tools 32.1-r
Previ...
Next
Cancel
Finish

License Agreement

Read and agree to the licenses for the components which will be installed

Terms and Conditions

This is the Android Software Development Kit License Agreement

1. Introduction

1.1 The Android Software Development Kit (referred to in the License Agreement as the "SDK" and specifically including the Android system files, packaged APIs, and Google APIs add-ons) is licensed to you subject to the terms of the License Agreement. The License Agreement forms a legally binding contract between you and Google in relation to your use of the SDK.

1.2 "Android" means the Android software stack for devices, as made available under the Android Open Source Project, which is located at the following URL: http://source.android.com/, as updated from time to time.

1.3 A "compatible implementation" means any Android device that (i) complies with the Android Compatibility Definition document, which can be found at the Android compatibility website (http://source.android.com/compatibility) and which may be updated from time to time; and (ii) successfully passes the Android Compatibility Test Suite (CTS).

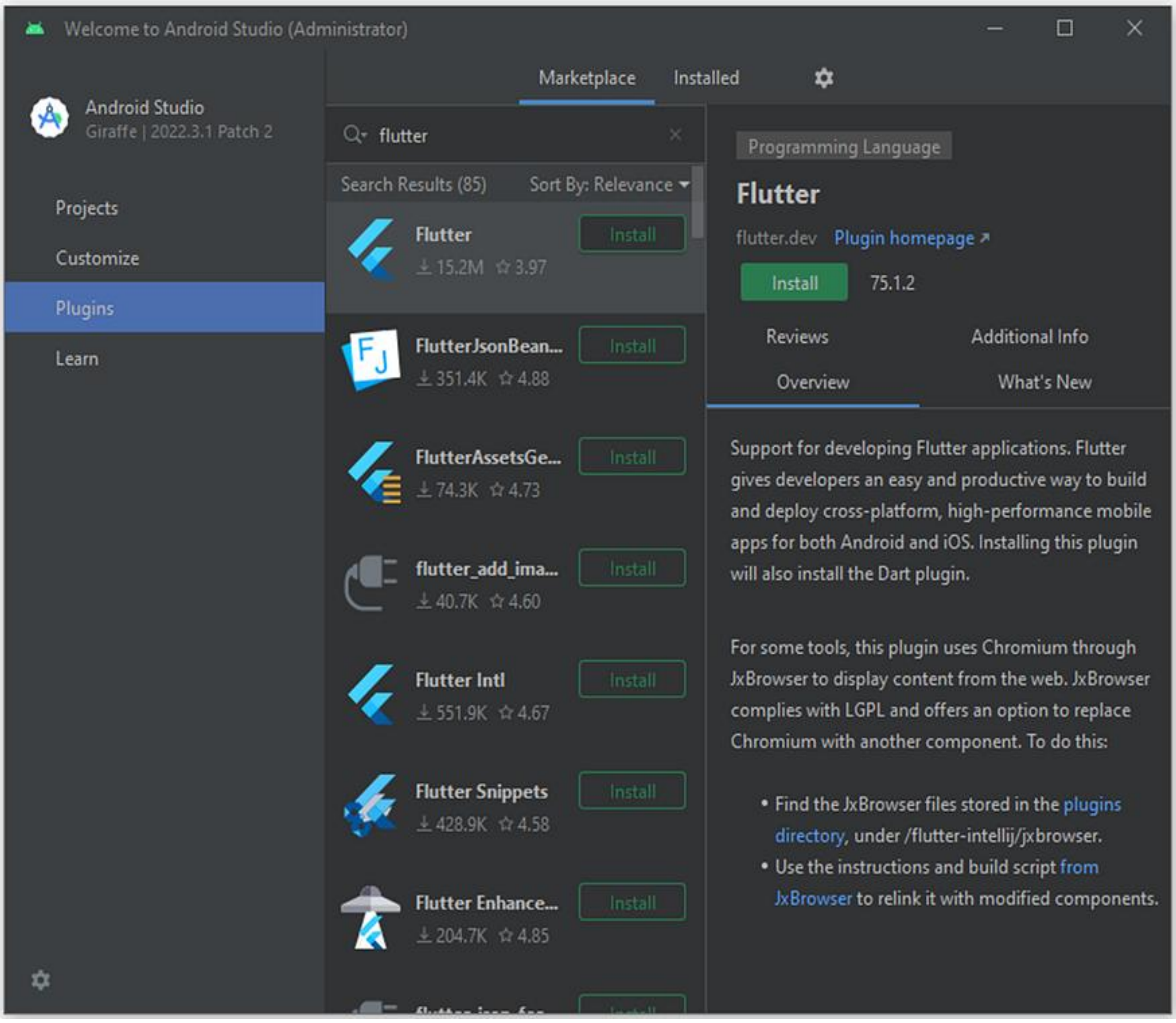
1.4 "Google" means Google Inc., a Delaware corporation with principal place of business at

○ Decline ● Accept

The download of the components will start and Android Studio install. Once completed, click Finish.

After the installation, start Android Studio. On the left side, click Plugins. Search for Flutter and click Install to install the Flutter plugin.

Press enter or click to view image in full size



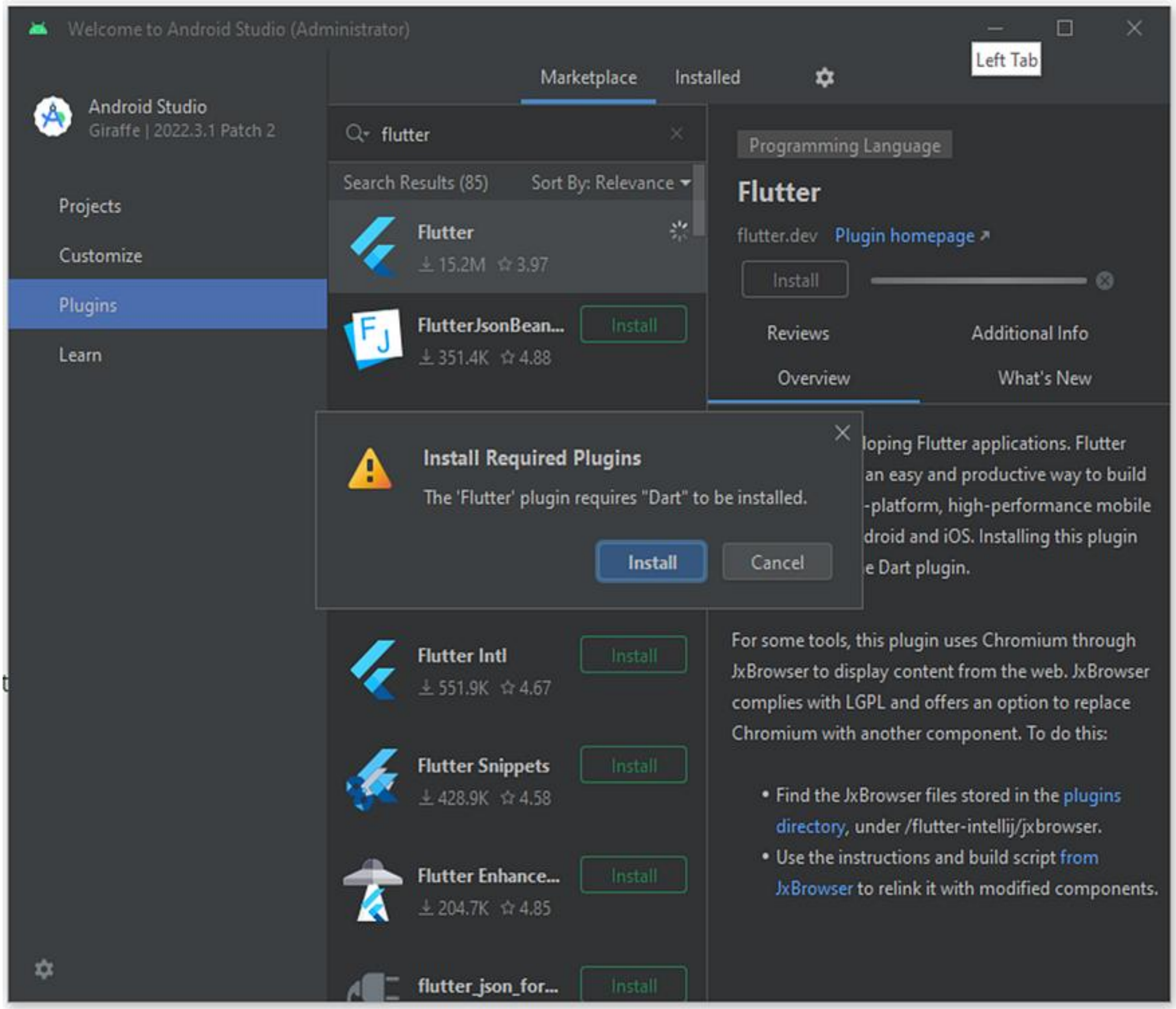
Support for developing Flutter applications. Flutter gives developers an easy and productive way to build and deploy cross-platform, high-performance mobile apps for both Android and iOS. Installing this plugin will also install the Dart plugin.

For some tools, this plugin uses Chromium through JxBrowser to display content from the web. JxBrowser complies with LGPL and offers an option to replace Chromium with another component. To do this:

- Find the JxBrowser files stored in the plugins directory, under /flutter-intellij/jxbrowser.
- Use the instructions and build script from JxBrowser to relink it with modified components.

It will also prompt you to install Dart, a programming language used to create Flutter apps. Click Install at the prompt.

Press enter or click to view image in full size



Install Required Plugins

The 'Flutter' plugin requires "Dart" to be installed.

Install | Cancel

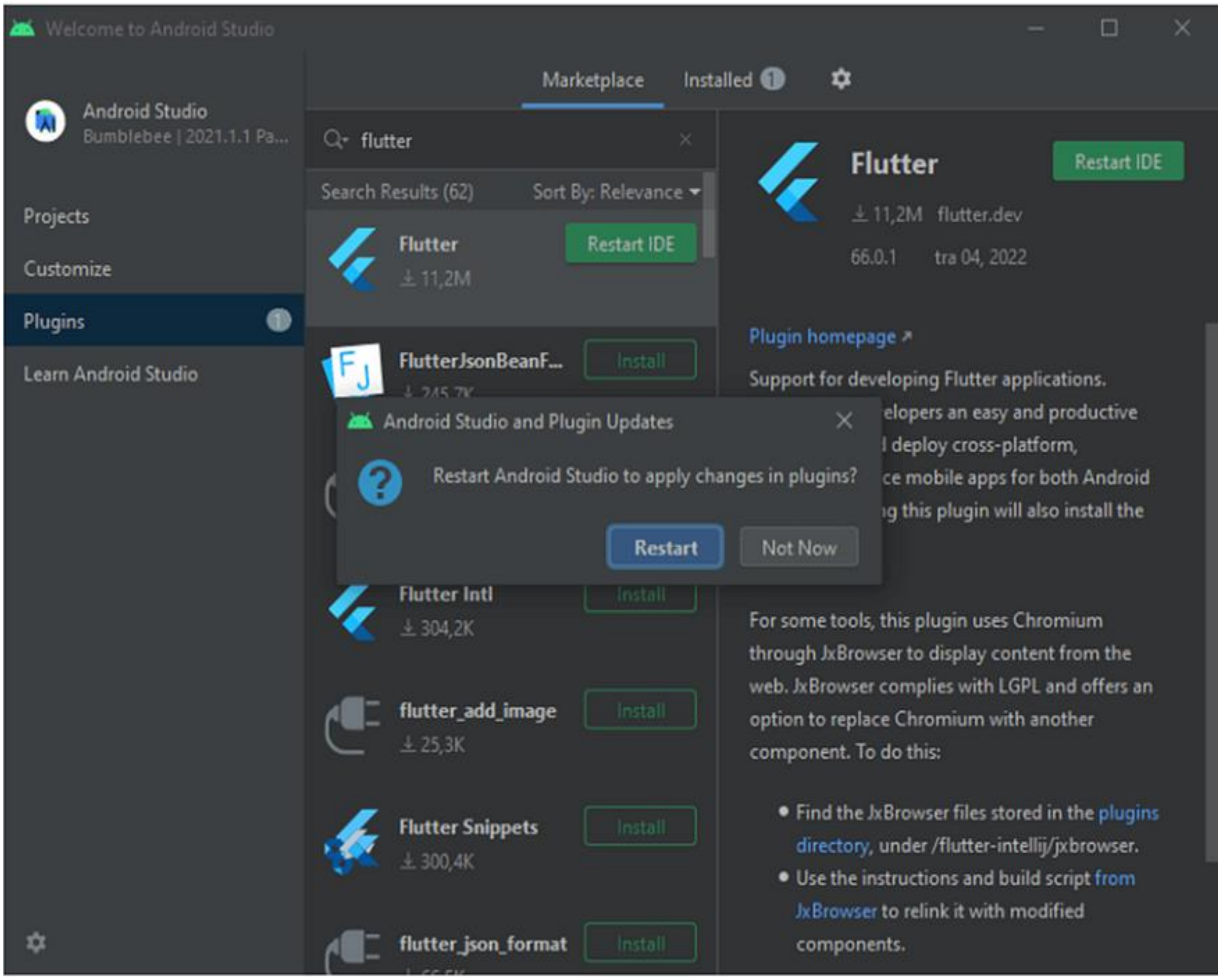
...loping Flutter applications. Flutter ...an easy and productive way to build ...-platform, high-performance mobile ...droid and iOS. Installing this plugin ...e Dart plugin.

For some tools, this plugin uses Chromium through JxBrowser to display content from the web. JxBrowser complies with LGPL and offers an option to replace Chromium with another component. To do this:

- Find the JxBrowser files stored in the plugins directory, under /flutter-intellij/jxbrowser.
- Use the instructions and build script from JxBrowser to relink it with modified components.

Finally, click Restart IDE so that the plugin changes are applied. Click Restart at the prompt to confirm this action.

Press enter or click to view image in full size



Restart Android Studio to apply changes in plugins?

Support for developing Flutter applications. ...elopers an easy and productive ...l deploy cross-platform, ...ce mobile apps for both Android ...g this plugin will also install the

For some tools, this plugin uses Chromium through JxBrowser to display content from the web. JxBrowser complies with LGPL and offers an option to replace Chromium with another component. To do this:

- Find the JxBrowser files stored in the plugins directory, under /flutter-intellij/jxbrowser.
- Use the instructions and build script from JxBrowser to relink it with modified components.

Set up the Android emulator

To configure your Flutter app to run in an Android emulator, follow these steps to create and select an emulator.

1. Enable **VM acceleration** on your development computer.

2. Start **Android Studio**.

3. Go to the **Settings** dialog to view the **SDK Manager**.

   - If you have a project open, go to **Tools > Device Manager.**
   - If the Welcome to Android Studio dialog displays, click the **More Options** icon that follows the **Open** button and click **Device Manager** from the dropdown menu.

4. Click **Virtual**.

5. Click **Create Device**.

   The **Virtual Device Configuration** dialog displays.

6. Select either **Phone** or **Tablet** under **Category**.

7. Select a device definition. You can browse or search for the device.

8. Click **Next**.

9. Click **x86 Images**.

10. Click one system image for the Android version you want to emulate.

    - If the desired image has a Download icon to the right of the Release Name, click it. The SDK Quickfix Installation dialog displays with a completion meter.

    - When the download completes, click Finish.

11. Click **Next**.

The **Virtual Device Configuration** displays its **Verify Configuration** step.

12. To rename the Android Virtual Device (AVD), change the value in the **AVD Name** box.

13. Click **Show Advanced Settings** and scroll to **Emulated Performance**.

14. From the **Graphics** dropdown menu, select **Hardware - GLES 2.0**.

This enables hardware acceleration and improves rendering performance.

15. Verify your AVD configuration. If it is correct, click Finish.
16. To learn more about AVDs, check out Managing AVDs.

17. In the **Device Manager** dialog, click the **Run** icon to the right of your desired AVD. The emulator starts up and displays the default canvas for your selected Android OS version and device.

2.2.3. Agree to Android licenses

Before you can use Flutter and after you install all prerequisites, agree to the licenses of the Android SDK platform.

1. Open an elevated console window.
2. Run the following command to enable signing licenses.

```
C:> flutter doctor --android-licenses
```

If you accepted the Android Studio licenses at another time, this command returns:

```
[=======================================] 100% Computing updates...
All SDK package licenses accepted.
```

You can skip the next step.

3. Before agreeing to the terms of each license, read each with care.

2.2.4. Run Flutter doctor

The **flutter doctor** command validates that all components of a complete Flutter development environment for Windows.

1. Open PowerShell.
2. To verify your installation of all the components, run the following command.

```
PS C:> flutter doctor
```

As you chose to develop for Android, you do not need *all* components. If you followed this guide, the result of your command should resemble:

```
Running flutter doctor...
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.27.0, on Microsoft Windows 11 [Version
10.0.22621.3155], locale en)
[√] Windows version (Installed version of Windows is version 10 or higher)
[√] Android toolchain - develop for Android devices (Android SDK version
35.0.1)
[!] Chrome - develop for the web
[!] Visual Studio - develop Windows apps
[√] Android Studio (version 2024.2)
[√] VS Code (version 1.95)
```

```
[√] Connected device (1 available)
[√] Network resources


! Doctor found issues in 2 categories.
```

2.3.5. Troubleshoot Flutter doctor issues

When the **flutter doctor** command returns an error, it could be for Flutter, VS Code, Android Studio, the connected device, or network resources.

If the **flutter doctor** command returns an error for any of these components, run it again with the verbose flag.

```
PS C:> flutter doctor -v
```

Check the output for other software you might need to install or further tasks to perform.

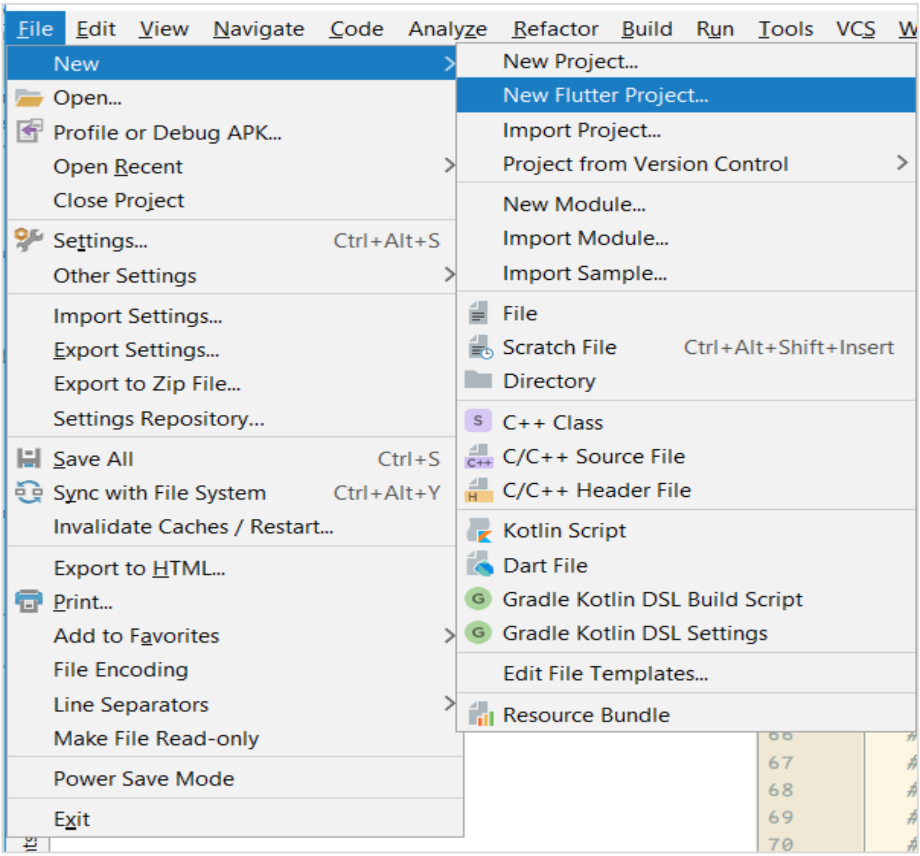
If you change the configuration of your Flutter SDK or its related components, run flutter doctor *again* to verify the installation.

3.1. Getting Started with Flutter: Building Your First App

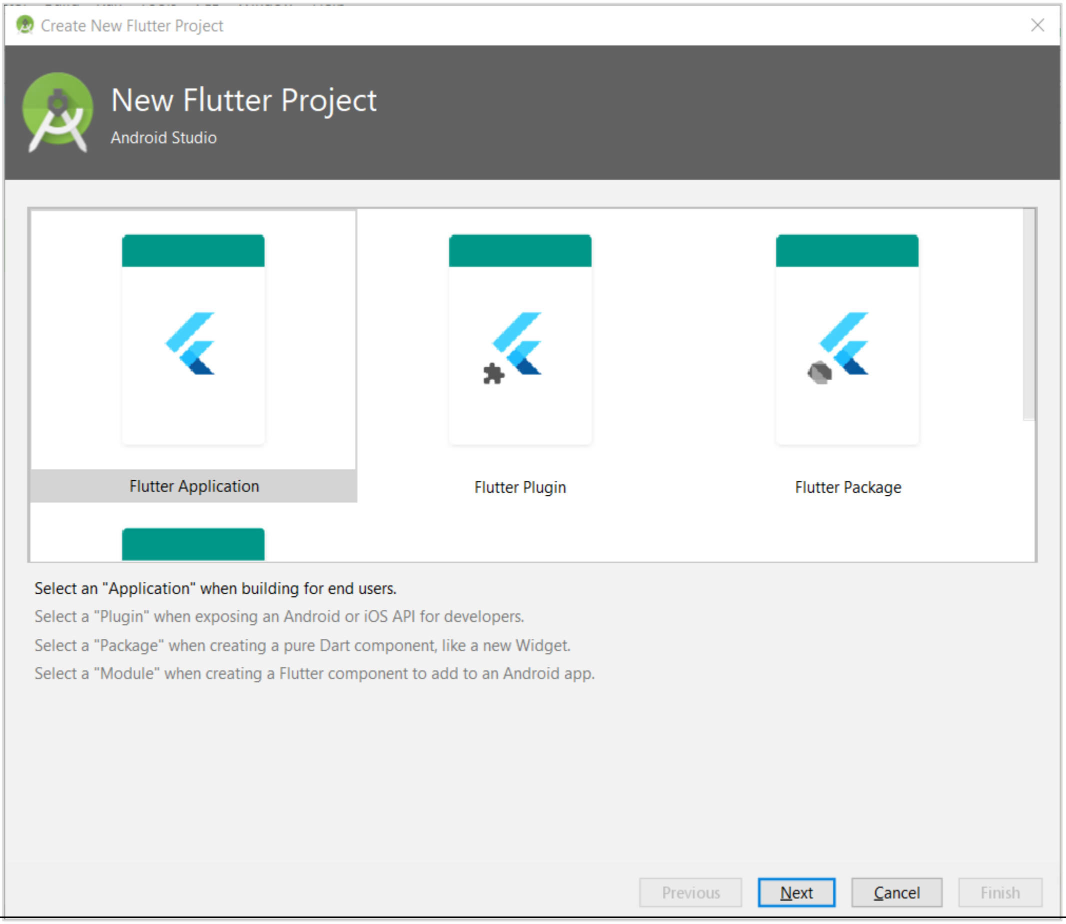
In this chapter, let us create a simple Flutter application to understand the basics of creating a flutter application in the Android Studio.

Step 1: Open **Android Studio or VS code**

Step 2: Create Flutter Project. For this, click **File -> New -> New Flutter Project**



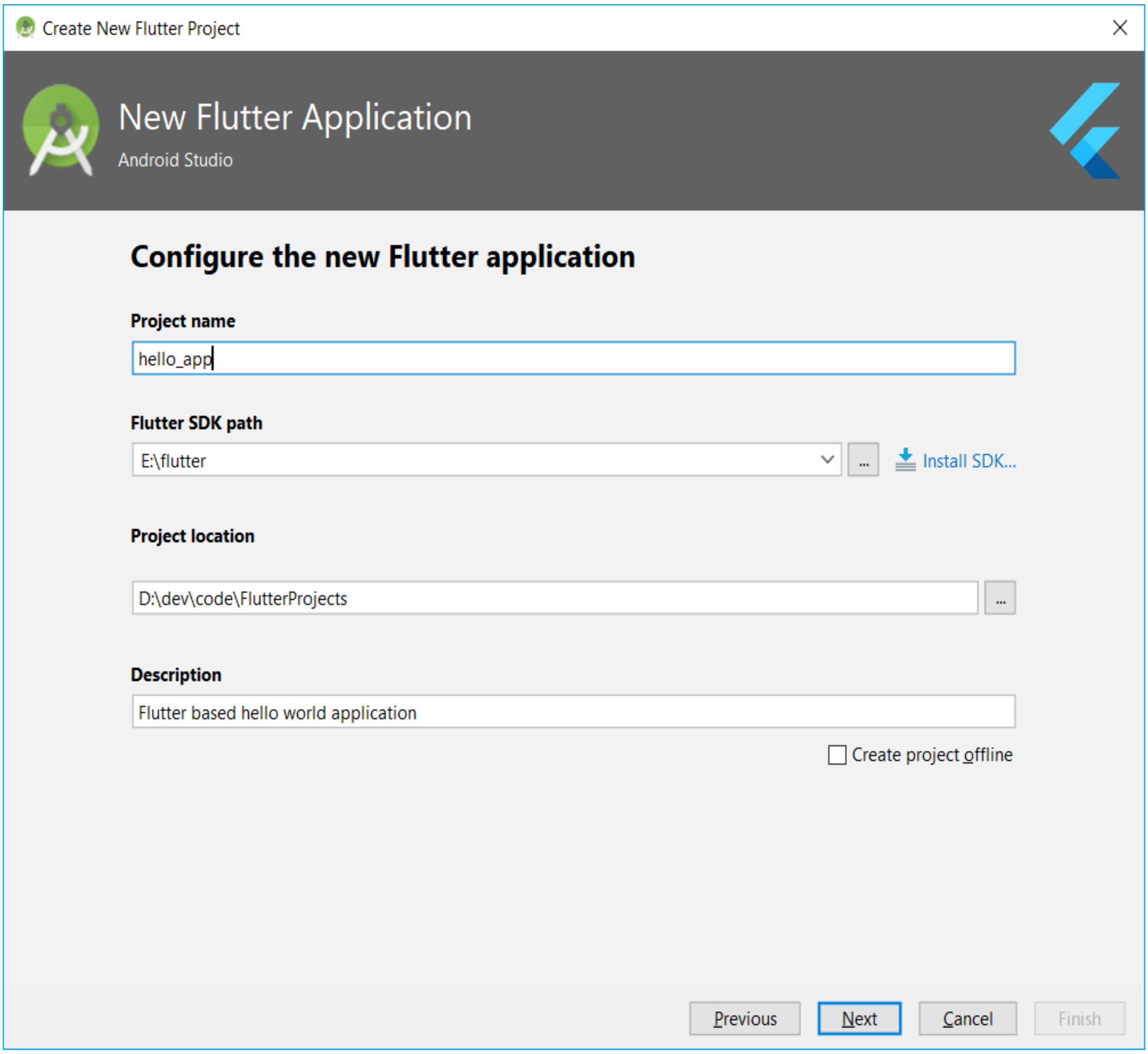
Step 3: Select Flutter Application. For this, select **Flutter Application** and click **Next**.



Select an "Application" when building for end users.
Select a "Plugin" when exposing an Android or iOS API for developers.
Select a "Package" when creating a pure Dart component, like a new Widget.
Select a "Module" when creating a Flutter component to add to an Android app.

Step 4: Configure the application as below and click **Next**.

- Project name: **hello_app**
- Flutter SDK Path: **<path_to_flutter_sdk>**
- Project Location**: <path_to_project_folder>**
- Description: Flutter based hello world application



Configure the new Flutter application

Project name: hello_app

Flutter SDK path: E:\flutter

Project location: D:\dev\code\FlutterProjects

Description: Flutter based hello world application

☐ Create project offline

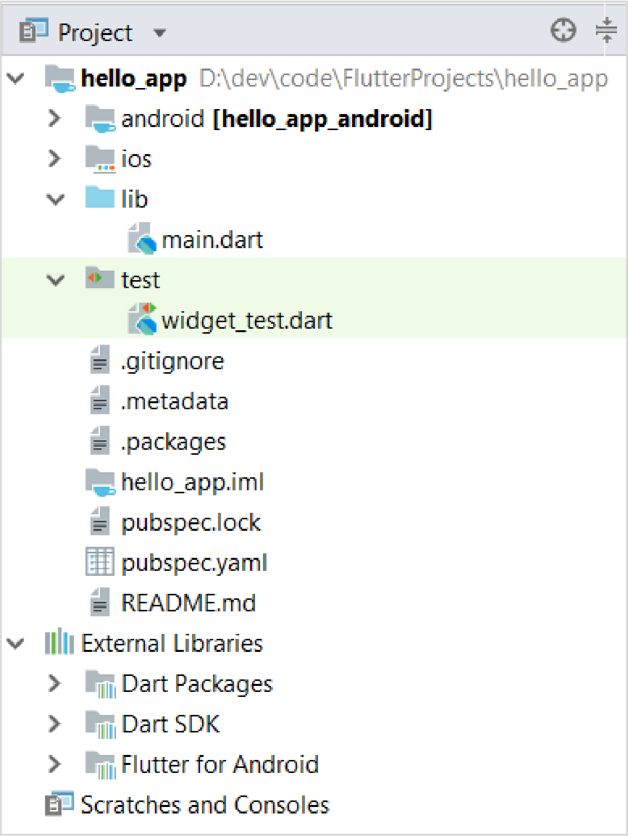
Step 5: Configure Project.

Set the company domain as **flutterapp.tutorialspoint.com** and click **Finish**

Step 6: Enter Company domain.

Android Studio creates a fully working flutter application with minimal functionality. Let us check the structure of the application and then, change the code to do our task.

The structure of the application and its purpose is as follows:



```
Project ▾
hello_app D:\dev\code\FlutterProjects\hello_app
  android [hello_app_android]
  ios
  lib
    main.dart
  test
    widget_test.dart
  .gitignore
  .metadata
  .packages
  hello_app.iml
  pubspec.lock
  pubspec.yaml
  README.md
External Libraries
  Dart Packages
  Dart SDK
  Flutter for Android
Scratches and Consoles
```

Various components of the structure of the application are explained here:

- **android** - Auto generated source code to create android application
- **ios** - Auto generated source code to create ios application
- **lib** - Main folder containing Dart code written using flutter framework
- **lib/main.dart** - Entry point of the Flutter application
- **test** - Folder containing Dart code to test the flutter application
- **test/widget_test.dart** - Sample code
- **.gitignore** - Git version control file
- **.metadata** - auto generated by the flutter tools
- **.packages** - auto generated to track the flutter packages
- **.iml** - project file used by Android studio
- **pubspec.yaml** - Used by Pub, Flutter package manager
- **pubspec.lock** - Auto generated by the Flutter package manager, Pub
- **README.md** - Project description file written in Markdown format

Step 7: Replace the dart code in the lib/main.dart file with the below code:

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

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Hello World Demo Application',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: MyHomePage(title: 'Home page'),
    );
  }
}

class MyHomePage extends StatelessWidget {
  MyHomePage({Key key, this.title}) : super(key: key);

  final String title;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(this.title),
      ),
      body: Center(
        child:
            Text(
              'Hello World',
            )
      ),
    );
  }
}
```

Let us understand the dart code line by line.

- Line 1: imports the flutter package, material. The material is a flutter package to create user interface according to the Material design guidelines specified by Android.

- Line 3: This is the entry point of the Flutter application. Calls runApp function and pass it an object of MyApp class. The purpose of the runApp function is to attach the given widget to the screen.

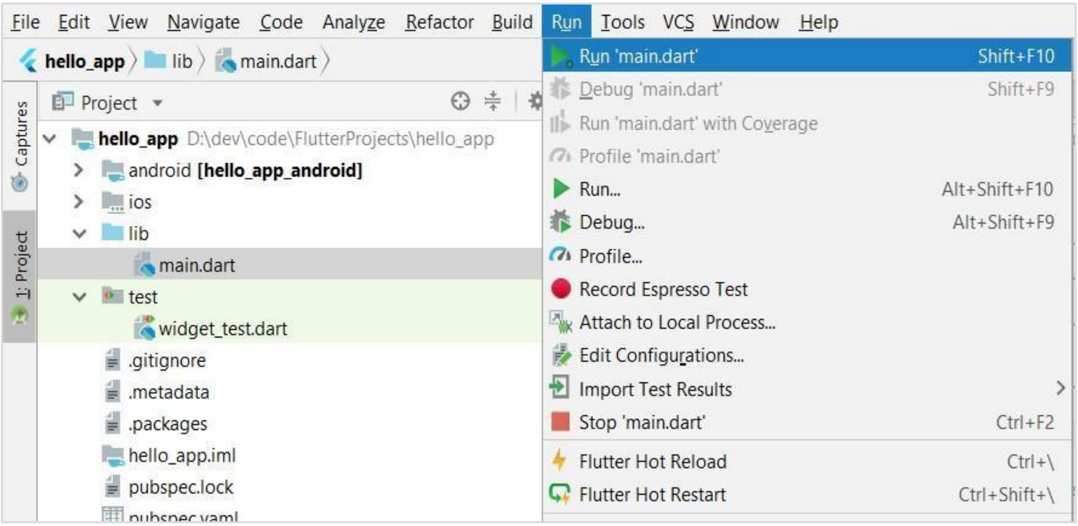
- Line 5 - 17: Widget is used to create UI in flutter framework. StatelessWidget is a widget, which does not maintain any state of the widget. MyApp extends

StatelessWidget and overrides its build method. The purpose of the build method is to create a part of the UI of the application. Here, build method uses MaterialApp, a widget to create the root level UI of the application. It has three properties - title, theme and home.

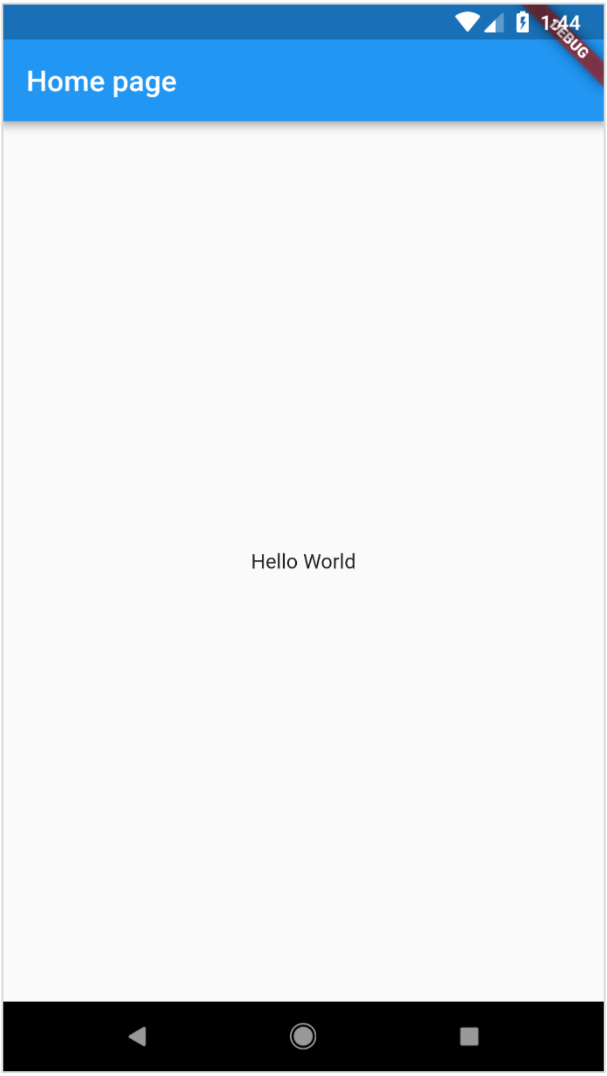
- title is the title of the application.
- theme is the theme of the widget. Here, we set blue as the overall color of the application using ThemeData class and its property, primarySwatch.
- home is the inner UI of the application, which we set another widget, MyHomePage

- **Line 19 - 38:** MyHomePage is same as MyApp except it returns Scaffold Widget. Scaffold is a top level widget next to MaterialApp widget used to create UI conforming material design. It has two important properties, appBar to show the header of the application and body to show the actual content of the application. AppBar is another widget to render the header of the application and we have used it in appBar property. In body property, we have used Center widget, which centers it child widget. Text is the final and inner most widget to show the text and it is displayed in the center of the screen.

Step 8: Now, run the application using, Run -> Run main.dart


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
hello_app > lib > main.dart
Project
hello_app D:\dev\code\FlutterProjects\hello_app
android [hello_app_android]
ios
lib
main.dart
test
widget_test.dart
.gitignore
.metadata
.packages
hello_app.iml
pubspec.lock
pubspec.yaml
Captures
1: Project
Run 'main.dart' Shift+F10
Debug 'main.dart' Shift+F9
Run 'main.dart' with Coverage
Profile 'main.dart'
Run... Alt+Shift+F10
Debug... Alt+Shift+F9
Profile...
Record Espresso Test
Attach to Local Process...
Edit Configurations...
Import Test Results
Stop 'main.dart' Ctrl+F2
Flutter Hot Reload Ctrl+\
Flutter Hot Restart Ctrl+Shift+\

Step 9: Finally, the output of the application is as follows:



3.1.1. Creating a New Flutter Project

Open your terminal and use the following command to create a new Flutter project:

```
flutter create my_first_app
```

This command generates a new Flutter project named my_first_app.

3.1.2. Understanding the Project Structure

Navigate to the lib folder within your project. This is where you'll find the main code for your Flutter app. Open the main.dart file. Here's a breakdown of the contents:

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'My First App',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Flutter Basics'),
        ),
        body: Center(
          child: Text('Hello, Flutter!'),
        ),
      ),
    );
  }
}
```

3.1.3. Breaking Down the Code

Let’s dive into the code and understand each part:

- We import the material.dart library, which contains the widgets for Material Design components.
- In the main function, we use runApp to start our Flutter app. We pass in an instance of MyApp.
- The MyApp class is a StatelessWidget, which means its content won't change once it's built. It returns a MaterialApp widget, the root of our app's widget tree.

- Inside MaterialApp, we specify the app's title and the home property, which defines the app's main screen.

- The Scaffold widget provides a basic app structure, including an app bar and a body.

- The AppBar widget creates a top app bar with a title.

- Inside the Scaffold's body, we use the Center widget to center its child, which is a simple Text widget displaying "Hello, Flutter!".

3.1.4. Running the App

Open your terminal again and navigate to your project's root directory. Run the following command to launch your app on an emulator or connected device:

```
flutter run
```

Voila! You've just built and run your first Flutter app. You should see a screen with an app bar displaying "Flutter Basics" and a centered text saying "Hello, Flutter!".

4.1 Lab Task

Perform the following task

Task 1:

Include screenshots showing the installation process of the required development tools flutter & android studio.

Task 2:

Run the default Flutter application code and include screenshots of its execution.