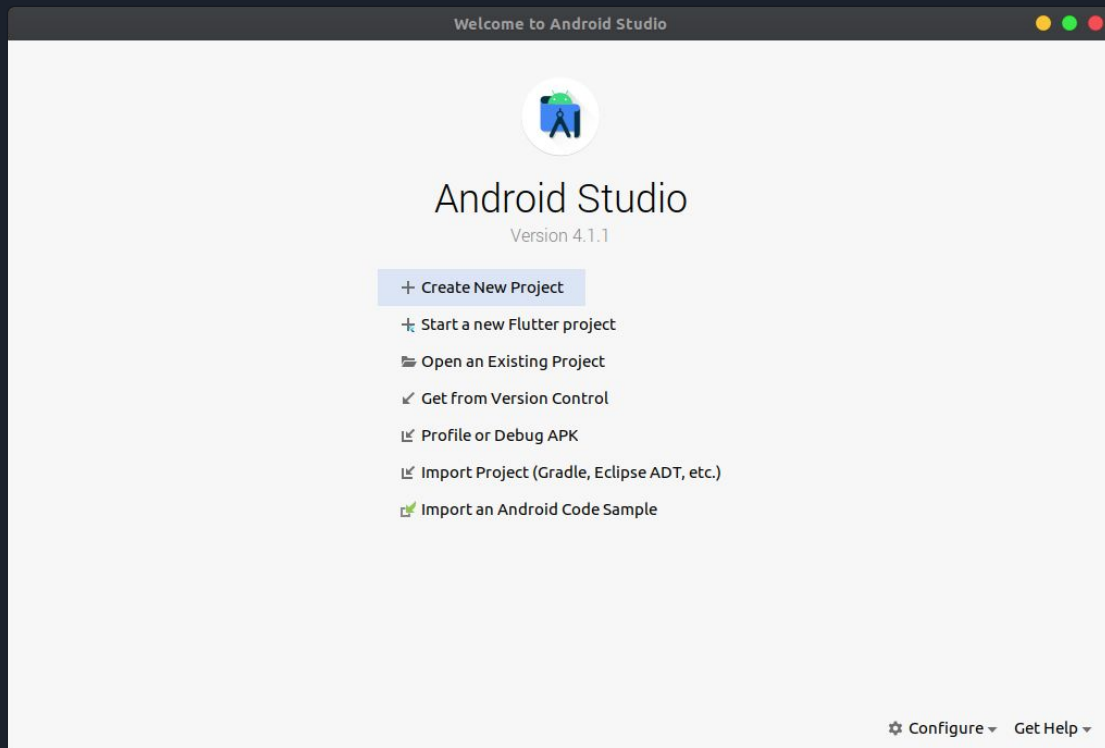




An Introduction to Flutter

IRIS NITK Bootcamp - Session 1

Creating a new Project





New Flutter Project



Flutter Application



Flutter Plugin



Flutter Package



Flutter Module

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 or iOS app.

Previous

Next

Cancel

Finish



New Flutter Application



Configure the new Flutter application

Project name

Flutter SDK path

[⬇ Install SDK...](#)

Project location



Description

☐ Create project offline[Previous](#)[Next](#)[Cancel](#)[Finish](#)



New Flutter Application

**Set the package name**

Applications and plugins need to generate platform-specific code

Package name**AndroidX**

☒ Use androidx.* artifacts

Platform channel language

☒ Include Kotlin support for Android code

☒ Include Swift support for iOS code

[Previous](#)[Next](#)[Cancel](#)[Finish](#)



New Flutter Application

**Set the package name**

Applications and plugins need to generate platform-specific code

Package name**AndroidX**

☒ Use androidx.* artifacts

Platform channel language

☒ Include Kotlin support for Android code

☒ Include Swift support for iOS code

[Previous](#)[Next](#)[Cancel](#)[Finish](#)

File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help

Introduction > lib > main.dart

Redmi 8 (mobile)

main.dart

Loading Devices...

Project

- Introduction ~/Desktop/introduction
 - .dart_tool
 - .idea
 - android [Introduction_android]
 - ios
 - lib
 - main.dart
 - test
 - .gitignore
 - .metadata
 - .packages
 - introduction.iml
 - pubspec.lock
 - pubspec.yaml
 - README.md
- External Libraries
- Scratches and Consoles

```
1 import 'package:flutter/material.dart';
2
3 void main() {
4   runApp(MyApp());
5 }
6
7 class MyApp extends StatelessWidget {
8   // This widget is the root of your application.
9   @override
10  Widget build(BuildContext context) {
11    return MaterialApp(
12      title: 'Flutter Demo',
13      theme: ThemeData(
14        // This is the theme of your application.
15        //
16        // Try running your application with "flutter run". You'll see the
17        // application has a blue toolbar. Then, without quitting the app, try
18        // changing the primarySwatch below to Colors.green and then invoke
```

Terminal: Local x +

harshvardhan@harshvardhan:introduction\$

Build Variants
Structure
FavoritesFlutter Outline
Flutter Inspector
Flutter PerformanceEmulator
Device File Explorer

Logcat Profiler Database Inspector TODO Terminal Dart Analysis

Layout Inspector Event Log



Terminology

<https://flutter.dev/>

Native Android and iOS

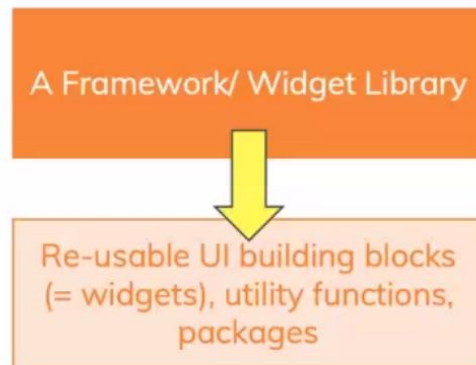
Cross Platform

SDK: This is a complete kit of software development tools for a specific platform. This “kit” can include all sorts of things such as: Libraries, APIs, IDEs, Documentation, etc. For example the Android SDK, which provides everything you may need for Android development.

Framework: A framework is a generic structure that provides a skeleton architecture with which specific software can be implemented.



A “tool” that allows you to build native cross-platform (iOS, Android) apps with one programming language and codebase.





Terminology

UI Toolkit

Seems like a loose term to refer to any collection of “tools” (another loose term) that have a common goal.

Is Flutter only for UI?

Yes, Flutter alone is used for getting the UI or Frontend part done!



Terminology

But wait then how are people creating Flutter apps?

Flutter is powered with **Dart** language. So, everything else apart from the UI is pure dart. Meaning, every function you code for getting API response or getting data from database will be written in Dart.

<https://medium.com/@shashvatshukla/framework-vs-library-vs-platform-vs-api-vs-sdk-vs-toolkits-vs-ide-50a9473999db>



Terminology

- A **plugin** is about making native functionality available to Flutter.
- A **module** is about integrating Flutter with an existing native application.
- A **package** is a namespace that contains a group of similar types of classes, interfaces, and sub-packages. We can think of packages as similar to different folders on our computers where we might keep movies in one folder, images in another folder, software in another folder, etc.



Directory Structure

- <https://stacksecrets.com/flutter/breaking-down-flutter-project-file-folders>



Theory

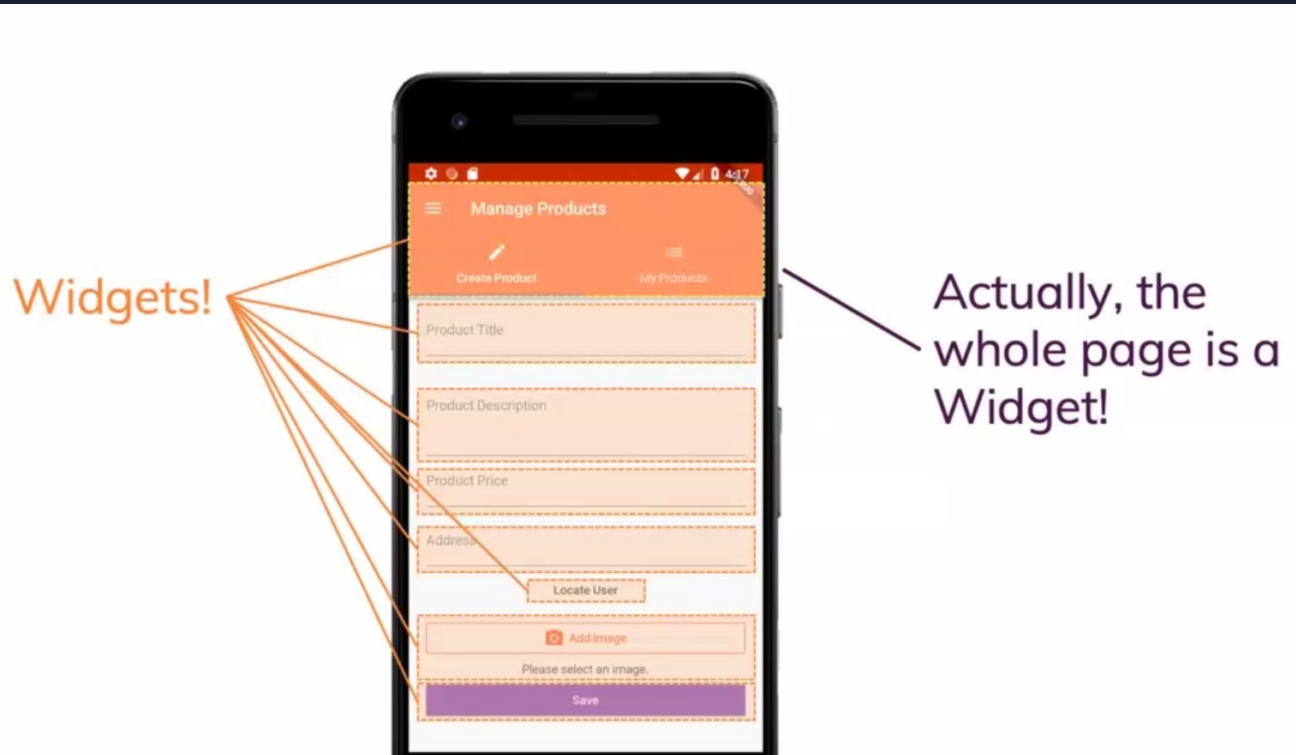
iOS : Swift or Objective C code and using the iOS development environment

Android: Java or Kotlin with Android Framework

Dart: it's an object oriented and strongly typed language and its syntax is a bit like a mixture of Javascript, Java, C#

Strongly typed :not possible for the programmer to work around the restrictions imposed by the type system.

Widget



State

It holds the information about the current behaviour or condition of the widget

Stateful vs. Stateless Widgets

Stateful Widget

When a widget changes
(user interacts with it) it's **Stateful**

CheckBox, RadioButton, Form, TextField

Overrides the **createState()** and returns a **State**

Use when the UI can change dynamically

When the widget's state changes, the state
object calls **setState()**, telling the framework
to redraw the widget.

Stateless Widget

No internal state to manage or no direct user
interaction, it's **Stateless**

Text, RaisedButton, Icon, IconButton

Overrides the **build()** and returns a **Widget**

Use when the UI depends on the information
within object itself



My Checkbox




User click on it and
its state changes



My Checkbox





Using Stateful Widgets

Create a class that extends a "StatefulWidget", that returns a State in "createState()"

Create a "State" class, with properties that may change

Within "State" class, implement the "build()" method

Call the setState() to make the changes. Calling setState() tells framework to redraw widget

