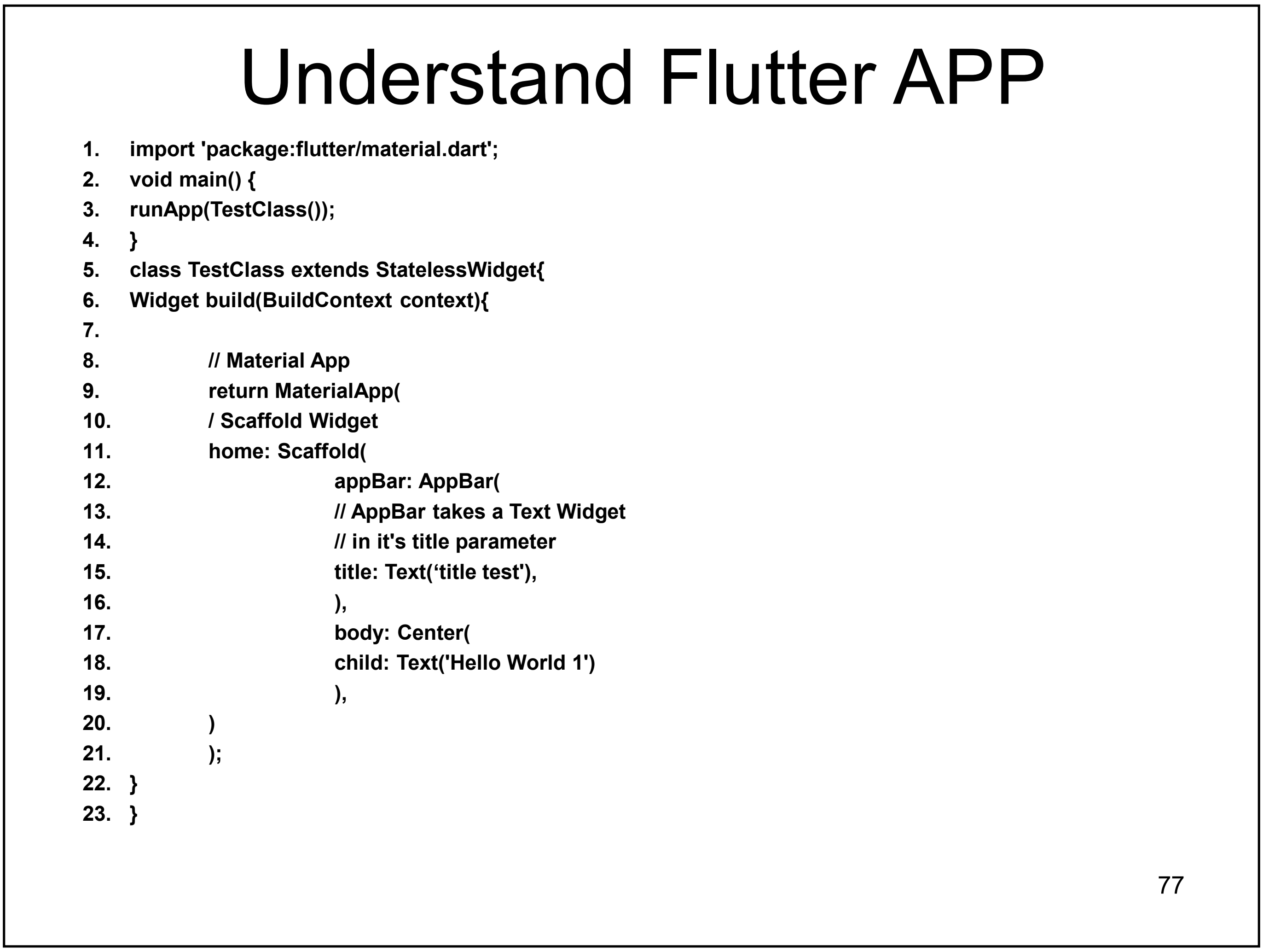
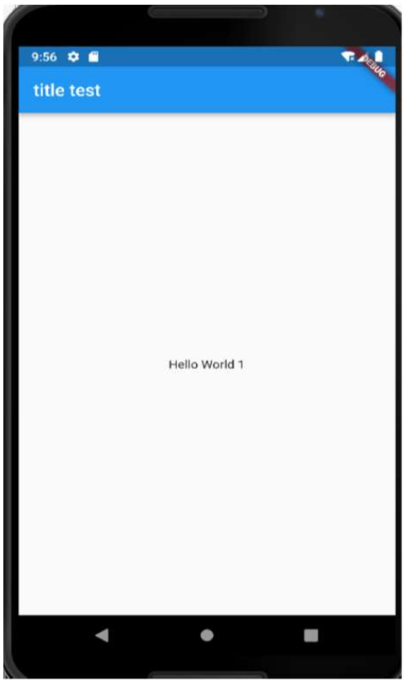


Understand Flutter APP

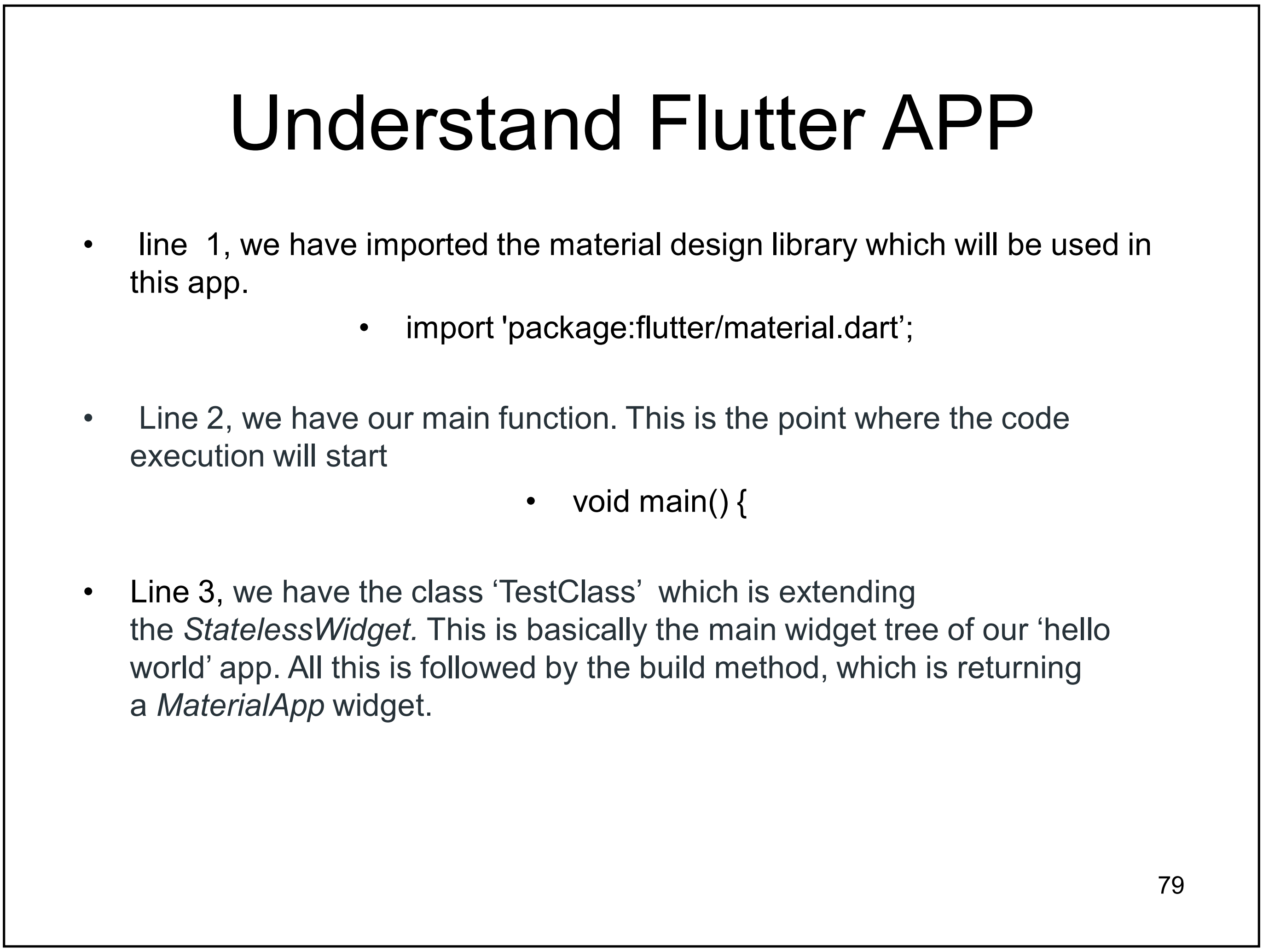
```
import 'package:flutter/material.dart';
void main() {
runApp(TestClass());
}
class TestClass extends StatelessWidget{
Widget build(BuildContext context){

        // Material App
        return MaterialApp(
        / Scaffold Widget
        home: Scaffold(
                    appBar: AppBar(
                    // AppBar takes a Text Widget
                    // in it's title parameter
                    title: Text(‘title test'),
                    ),
                    body: Center(
                    child: Text('Hello World 1')
                    ),
        )
        );
}
}
```



title test

Hello World 1



Understand Flutter APP

- line 1, we have imported the material design library which will be used in this app.
    - import 'package:flutter/material.dart’;

- Line 2, we have our main function. This is the point where the code execution will start
    - void main() {

- Line 3, we have the class ‘TestClass’ which is extending the *StatelessWidget.* This is basically the main widget tree of our ‘hello world’ app. All this is followed by the build method, which is returning a *MaterialApp* widget.

Understand Flutter APP

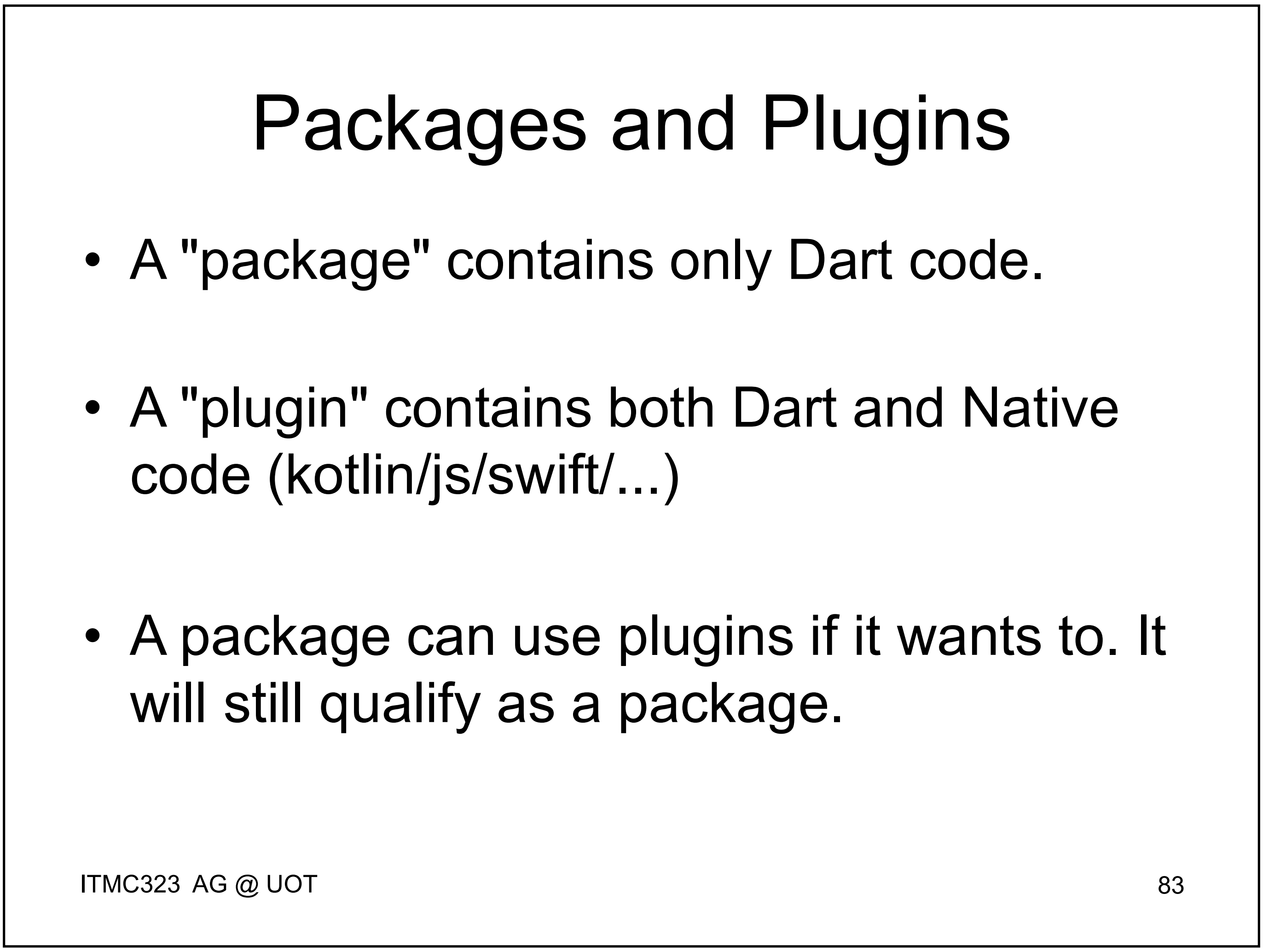
- Then we have employed home property of the *MaterialApp*, which in turn is holding the *Scaffold* widget.
- The Scaffold widget is containing the whole screen of the app. We have used the *appBar* property which is taking the *AppBar* widget as the object. And in turn the *AppBar* widget is holding ‘title test’ as the title. Then we have the *body,* which is again the property of the *MaterialApp*. *Center* is the object of the *body* and it’s child is *Text* widget which reads ‘Hello World 1’.

Packages

Like many languages, Dart allows developers to share modular code, which can be used by other developers to easily get commonly needed functionality that would otherwise require a complicated solution.

Packages and Plugins

- **Packages**
  - At a minimum, a Dart package is a directory containing a pubspec file. Additionally, a package can contain dependencies (listed in the pubspec), Dart libraries, apps, resources, tests, images, and examples. The pub.dev site lists many packages—developed by Google engineers and generous members of the Flutter and Dart community— that you can use in your app.
- **Plugins**
  - A plugin package is a special kind of package that makes platform functionality available to the app. Plugin packages can be written for Android (using Kotlin or Java), iOS (using Swift or Objective-C), web, macOS, Windows, Linux, or any combination thereof. For example, a plugin might provide Flutter apps with the ability to use a device's camera.



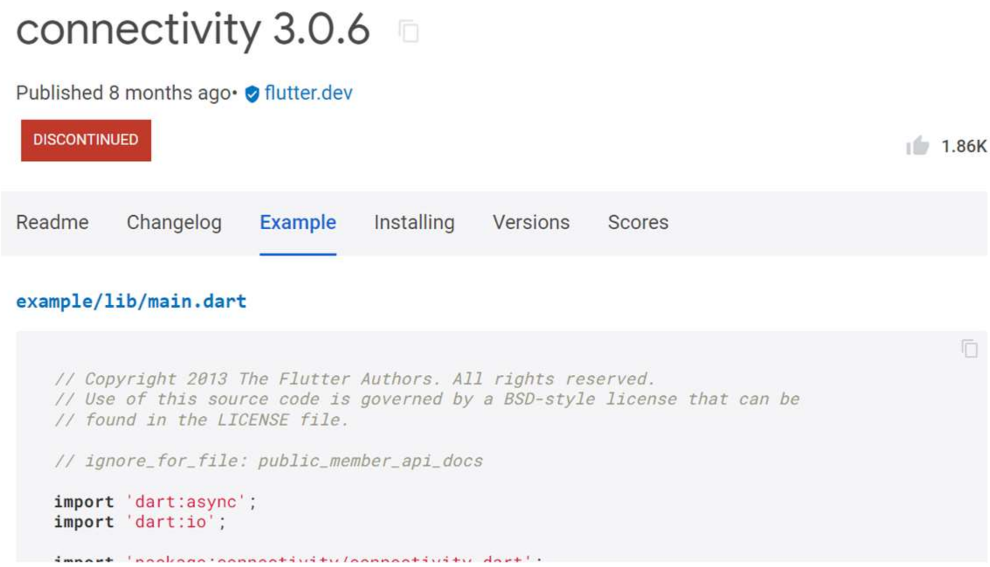
Packages and Plugins

- A "package" contains only Dart code.
- A "plugin" contains both Dart and Native code (kotlin/js/swift/...)
- A package can use plugins if it wants to. It will still qualify as a package.

Finding Packages

https://pub.dev/

Use Packages



connectivity 3.0.6

Published 8 months ago• flutter.dev

DISCONTINUED

example/lib/main.dart

```
// Copyright 2013 The Flutter Authors. All rights reserved.
// Use of this source code is governed by a BSD-style license that can be
// found in the LICENSE file.

// ignore_for_file: public_member_api_docs

import 'dart:async';
import 'dart:io';

import 'package:connectivity/connectivity.dart';
```