Flutter
程式介紹

Michael Lin

Learn Flutter

pubspec.yaml

main.dart

build.gradle

使用 Flutter 開發行動應用程式時，重要文件之一是 pubspec.yaml。

pubspec.yaml 檔案用作 Flutter 專案的清單，您可以在其中定義應用程式配置和運行所需的依賴項、資產、元資料和其他關鍵資訊。

在本文中，我們將探討 pubspec.yaml 檔案的各個方面，並了解它如何在 Flutter 開發中發揮重要作用。什麼是 pubspec.yaml？pubspec.yaml 檔案是 Flutter 專案中用於管理依賴項和資源的 YAML（另一種標記語言）檔案。它位於專案目錄的根目錄中，並提供用於聲明專案的依賴項、資產、版本約束等的結構化格式。定義依賴關係 pubspec.yaml 檔案的主要用途之一是管理專案的依賴項。依賴項是 Flutter 應用程式賴以提供附加功能的外部套件或程式庫。透過在 pubspec.yaml 檔案中聲明依賴項，您可以啟用 Flutter 套件管理器 pub 來為您的專案取得和管理這些套件。新增資產除了管理依賴項之外，pubspec.yaml 檔案還允許您包含應用程式所需的圖像、字體和其他檔案等資產。透過聲明資產，您可以確保它們在您的應用程式中捆綁並可存取。元數據和配置pubspec.yaml 檔案還允許您為 Flutter 應用程式提供額外的元資料和配置。

您可以指定應用程式的名稱、版本、作者、描述等。這是一個例子：運行 pub 指令在 pubspec.yaml 檔案中定義依賴項、資產和其他配置後，您可以使用 pub 命令列工具來管理您的專案。常見的 pub 指令包括：get：取得 pubspec.yaml 檔案中指定的所有相依性。升級：在指定的版本限制內將相依性升級到最新版本。建置：建置專案並執行依賴項指定的任何必要的轉換或程式碼產生。若要執行這些命令，請導覽至終端機中專案的根目錄並執行 flutter pub <command>。例如，flutter pub get 取得 pubspec.yaml 檔案中指定的專案相依性。

pubspec.yaml

```
name: foodie
description: "A new Flutter project."

publish_to: 'none' # Remove this
line if you wish to publish to
pub.dev
version: 1.0.0+1

environment:
  sdk: ^3.5.3

dependencies:
  flutter:
    sdk: flutter
  flutter_localizations:
    sdk: flutter
  intl: ^0.19.0
  image_gallery_saver: ^2.0.3
  image_picker: ^1.0.7
  image: ^4.0.17

flutter:
  generate: true
  uses-material-design: true
```

- 名稱和描述

- name: foodie
- 指定專案的名稱，這裡是 “foodie” 。

- description: "A new Flutter project."
- 專案的簡短描述，這裡是 “A new Flutter project” 。

- 發佈
- publish_to: 'none'
- 這行指定不將該專案發佈至pub.dev。若要發佈，刪除此行。

- 版本
- version: 1.0.0+1
- 定義專案的當前版本。格式為 major.minor.patch+build。

- 環境
- sdk: ^3.5.3
- 指定Dart SDK的版本限制。你的專案與3.5.3版本及更高版本相容。

- 依賴項
- flutter: sdk: flutter
- dependencies:
- flutter:
- sdk: flutter
- 將Flutter添加為來自Flutter SDK的依賴項 。

pubspec.yaml

```
name: foodie
description: "A new Flutter project."

publish_to: 'none' # Remove this
line if you wish to publish to
pub.dev
version: 1.0.0+1

environment:
  sdk: ^3.5.3

dependencies:
  flutter:
    sdk: flutter
  flutter_localizations:
    sdk: flutter
  intl: ^0.19.0
  image_gallery_saver: ^2.0.3
  image_picker: ^1.0.7
  image: ^4.0.17

flutter:
  generate: true
  uses-material-design: true
```

- flutter_localizations: sdk: flutter
- flutter_localizations:
- sdk: flutter
- 包含Flutter的本地化支持。

- intl: ^0.19.0
- 添加 intl 套件，用於國際化和本地化。

- image_gallery_saver: ^2.0.3
- 保存圖片至相簿的套件。

- image_picker: ^1.0.7
- 從相簿或相機選取圖片的套件。

- image: ^4.0.17
- 高級圖片處理的套件。

- Flutter 設定
- generate: true
- flutter:
- generate: true
- 在構建時自動生成文件。

- uses-material-design: true
- 指示專案使用 Material Design 庫。

使用 Flutter 開發行動應用程式時，他是重要文件之一是 pubspec.yaml。

pubspec.yaml 檔案用作 Flutter 專案的清單，您可以在其中定義應用程式配置和運行所需的依賴項、資產、元資料和其他關鍵資訊。

在本文中，我們將探討 pubspec.yaml 檔案的各個方面，並了解它如何在 Flutter 開發中發揮重要作用。什麼是 pubspec.yaml？pubspec.yaml 檔案是 Flutter 專案中用於管理依賴項和資源的 YAML（另一種標記語言）檔案。它位於專案目錄的根目錄中，並提供用於聲明專案的依賴項、資產、版本約束等的結構化格式。定義依賴關係 pubspec.yaml 檔案的主要用途之一是管理專案的依賴項。依賴項是 Flutter 應用程式賴以提供附加功能的外部套件或程式庫。透過在 pubspec.yaml 檔案中聲明依賴項，您可以啟用 Flutter 套件管理器 pub 來為您的專案取得和管理這些套件。新增資產除了管理依賴項之外，pubspec.yaml 檔案還允許您包含應用程式所需的圖像、字體和其他檔案等資產。透過聲明資產，您可以確保它們在您的應用程式中捆綁並可存取。元數據和配置pubspec.yaml 檔案還允許您為 Flutter 應用程式提供額外的元資料和配置。

您可以指定應用程式的名稱、版本、作者、描述等。這是一個例子：運行 pub 指令在 pubspec.yaml 檔案中定義依賴項、資產和其他配置後，您可以使用 pub 命令列工具來管理您的專案。常見的 pub 指令包括：get：取得 pubspec.yaml 檔案中指定的所有相依性。升級：在指定的版本限制內將相依性升級到最新版本。建置：建置專案並執行依賴項指定的任何必要的轉換或程式碼產生。若要執行這些命令，請導覽至終端機中專案的根目錄並執行 flutter pub <command>。例如，flutter pub get 取得 pubspec.yaml 檔案中指定的專案相依性。

main.dart

```dart
import 'package:flutter/material.dart';
import 'package:flutter_localizations/flutter_localizations.dart';
import 'package:flutter_gen/gen_l10n/app_localizations.dart';
import 'entrance_page.dart';
import 'photo_taking.dart';
```

- 這些行是導入相關的套件和文件：
- flutter/material.dart：引入Flutter的Material設計庫。
- flutter_localizations/flutter_localizations.dart：引入Flutter的本地化支持。
- flutter_gen/gen_l10n/app_localizations.dart：引入生成的本地化資源文件。
- entrance_page.dart 和 photo_taking.dart：引入專案中的頁面文件。

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

- 這段是主函數，啟動整個應用並運行 MyApp

```dart
class MyApp extends StatefulWidget {
  @override
  _MyAppState createState() => _MyAppState();

  static void setLocale(BuildContext context, Locale newLocale) {
    _MyAppState? state = context.findAncestorStateOfType<_MyAppState>();
    state?.setLocale(newLocale);
  }
}
```

- 這裡定義了一個狀態化的Widget MyApp，並包括一個靜態方法 setLocale 來設定應用的語言環境。

main.dart

```dart
class _MyAppState extends State<MyApp> {
  Locale _locale = Locale('zh', 'TW');

  void setLocale(Locale locale) {
    setState(() {
      _locale = locale;
    });
  }

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Menu Photography App',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      locale: _locale,
      localizationsDelegates: [
        AppLocalizations.delegate,
        GlobalMaterialLocalizations.delegate,
        GlobalWidgetsLocalizations.delegate,
        GlobalCupertinoLocalizations.delegate,
      ],
      supportedLocales: [
        Locale('en', ''),
        Locale('zh', 'TW'),
        Locale('zh', 'CN'),
        Locale('ja', ''),
        Locale('ko', ''),
        Locale('zh', ''),
      ],
      home: EntrancePage(),
      routes: {
        '/camera': (context) => PhotoTakingScreen(),
      },
    );
  }
}
```

- 這部分是 MyApp 的狀態類 _MyAppState：
- _locale：定義初始語言環境為繁體中文（台灣）。
- setLocale：方法用於更新語言環境。
- build：構建應用的UI，包括應用名稱、主題色、語言環境、本地化代理和支援的語言。
- 簡單來說，這段程式碼定義了一個多語言支援的Flutter應用，主要用於拍照和菜單管理。

android\app\build.gradle

def localProperties = new
創建一個新的 Properties 物件，用來儲存屬性資料。

def localPropertiesFile = rootProject.file('local.properties')

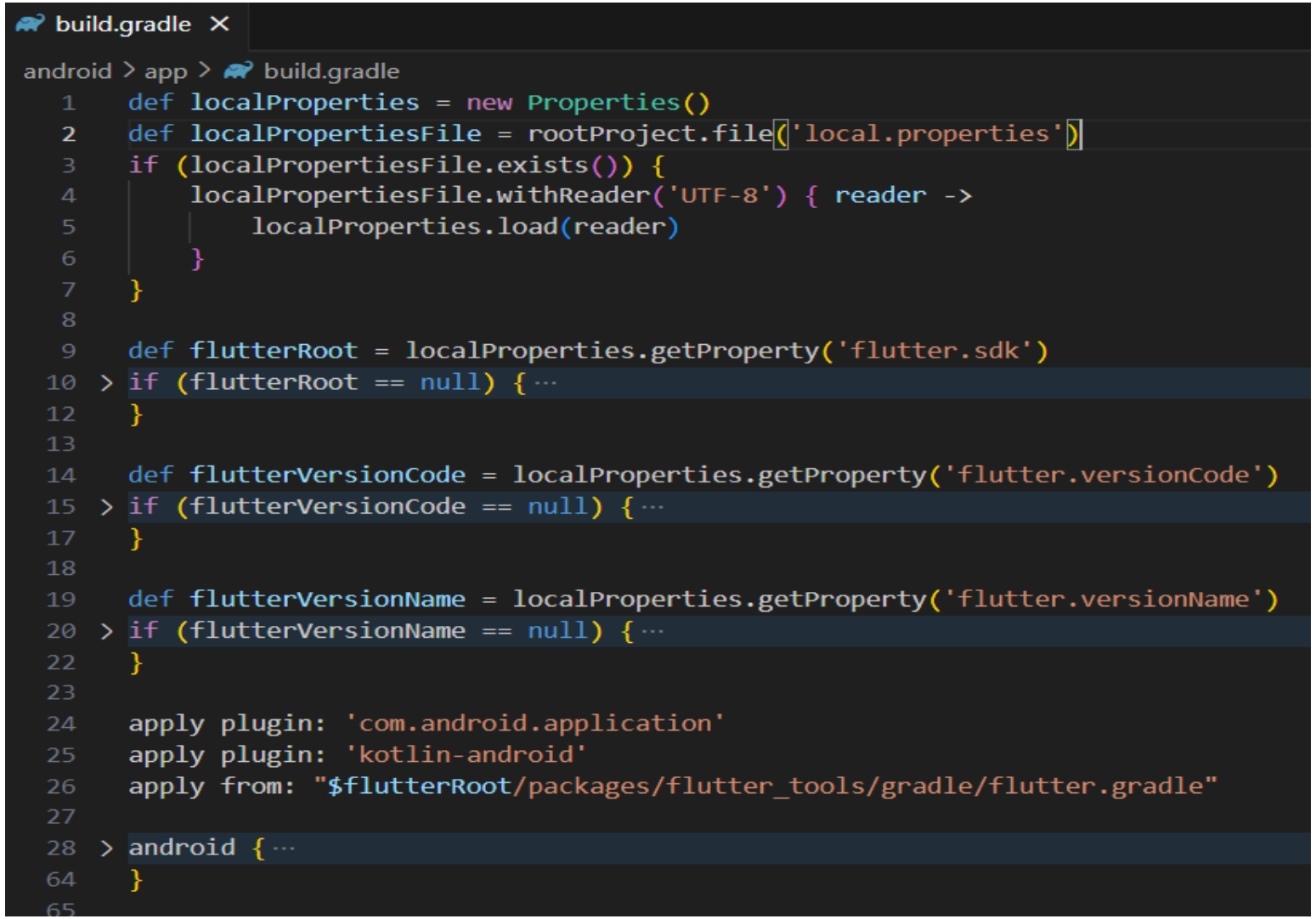
將 local.properties 文件指定為 rootProject 專案目錄下的文件。

if (localPropertiesFile.exists()) {

檢查 local.properties 文件是否存在。

localPropertiesFile.withReader('UTF-8') { reader ->
    localProperties.load(reader)
}

如果文件存在，使用 UTF-8 編碼讀取文件內容，並將這些屬性加載到 localProperties 物件中。



android\app\build.gradle

從 local.properties 文件中獲取 flutter.sdk 屬性，即Flutter SDK的路徑。

如果 flutterRoot 為空，拋出一個異常，提示找不到Flutter SDK，並要求在 local.properties 文件中定義 flutter.sdk 的位置。

從 local.properties 文件中獲取 flutter.versionCode 屬性。

如果這個屬性不存在，則默認設置為 1。

從 local.properties 文件中獲取 flutter.versionName 屬性。

如果這個屬性不存在，則默認設置為 1.0。

```
10  def flutterRoot = localProperties.getProperty('flutter.sdk')
11  if (flutterRoot == null) {
12      throw new GradleException("Flutter SDK not found. Define location with flutt
13  }
14
15  def flutterVersionCode = localProperties.getProperty('flutter.versionCode')
16  if (flutterVersionCode == null) {
17      flutterVersionCode = '1'
18  }
19
20  def flutterVersionName = localProperties.getProperty('flutter.versionName')
21  if (flutterVersionName == null) {
22      flutterVersionName = '1.0'
23  }
```