

# Manual de Compilación y Ejecución

## Proyecto: Flutter Snake Game

### 1. Introducción

Este documento describe cómo compilar, ejecutar y comprender la estructura del proyecto **Flutter Snake Game**.

El objetivo es que cualquier usuario o desarrollador pueda ejecutar el proyecto en su equipo y entender cómo está organizado, sin necesidad de conocer el funcionamiento interno del código.

### 2. Requisitos del sistema

Para poder ejecutar el proyecto es necesario disponer de:

- Sistema operativo Windows, macOS o Linux
- Git
- Flutter SDK (versión 3.x o superior)
- Dart SDK (incluido en Flutter)
- Entorno de desarrollo recomendado:
  - Android Studio o
  - Visual Studio Code
- Dispositivo de prueba:
  - Emulador Android
  - Navegador web (Chrome)
  - Dispositivo físico

Para verificar la instalación de Flutter: flutter doctor

```
C:\Users\Usuario>flutter doctor
```

```
A new version of Flutter is available!
```

```
To update to the latest version, run "flutter upgrade".
```

```
Doctor summary (to see all details, run flutter doctor -v):
```

```
[✓] Flutter (Channel stable, 3.38.8, on Microsoft Windows [Versión 10.0.26200.7623], locale es-ES)
```

```
[✓] Windows Version (11 Home 64-bit, 25H2, 2009)
```

```
[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/to/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
```

```
[✓] Visual Studio - develop Windows apps (Visual Studio Community 2019 16.11.1)
```

```
[✓] Connected device (3 available)
```

```
[✓] Network resources
```

```
! Doctor found issues in 1 category.
```

### 3. Descarga del proyecto

El proyecto se descarga desde un repositorio Git.

Pasos:

1. Abrir una terminal
2. Ejecutar: `git clone https://github.com/Onyx2006/flutter-snake.git`  
`cd flutter_snake`

```
C:\Users\Usuario\Downloads>git clone https://github.com/Onyx2006/flutter-snake.git
Cloning into 'flutter-snake'...
remote: Enumerating objects: 393, done.
remote: Counting objects: 100% (393/393), done.
remote: Compressing objects: 100% (285/285), done.
remote: Total 393 (delta 121), reused 328 (delta 70), pack-reused 0 (from 0)
Receiving objects: 100% (393/393), 3.62 MiB | 1.35 MiB/s, done.
Resolving deltas: 100% (121/121), done.
```

```
C:\Users\Usuario\Downloads>cd flutter-snake
C:\Users\Usuario\Downloads\flutter-snake>
```

### 4. Instalación de dependencias

Una vez dentro del proyecto, se deben instalar las dependencias necesarias: `flutter pub get`

Este comando descarga automáticamente todas las librerías necesarias.

```
C:\Users\Usuario\Downloads\flutter-snake>flutter pub get
Resolving dependencies... (1.3s)
Downloading packages...
characters 1.4.0 (1.4.1 available)
go_router 17.0.1 (17.1.0 available)
hooks 1.0.0 (1.0.1 available)
lints 6.0.0 (6.1.0 available)
matcher 0.12.17 (0.12.18 available)
material_color_utilities 0.11.1 (0.13.0 available)
meta 1.17.0 (1.18.1 available)
test_api 0.7.7 (0.7.9 available)
Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
Building with plugins requires symlink support.

Please enable Developer Mode in your system settings. Run
start ms-settings:developers
to open settings.
```

Sale un error y es totalmente normal en sistemas operativos Windows. Esto se debe a que Flutter (y muchos de sus plugins) usan **enlaces simbólicos (symlinks)** para funcionar correctamente. En **Windows**, por seguridad, **NO están permitidos por defecto**.

Para solucionarlo pulsamos **Win + I** para entrar en la configuración, entramos en el apartado de sistemas, buscamos el apartado que dice Opciones avanzadas. Entramos y activamos el Modo para desarrolladores.



Cerramos terminal y volvemos a abrirla para que se apliquen los cambios y ponemos **flutter pub get** para instalar las dependencias necesarias para arrancar el proyecto.

```
C:\Users\Usuario\Downloads\flutter-snake>flutter pub get
Resolving dependencies... (1.1s)
Downloading packages...
  characters 1.4.0 (1.4.1 available)
  go_router 17.0.1 (17.1.0 available)
  hooks 1.0.0 (1.0.1 available)
  lints 6.0.0 (6.1.0 available)
  matcher 0.12.17 (0.12.18 available)
  material_color_utilities 0.11.1 (0.13.0 available)
  meta 1.17.0 (1.18.1 available)
  test_api 0.7.7 (0.7.9 available)
Got dependencies!
8 packages have newer versions incompatible with dependency constraints.
Try 'flutter pub outdated' for more information.
```

## 5. Estructura del proyecto

El proyecto está organizado siguiendo una **arquitectura clara y modular**, separando la lógica del juego, la interfaz y la navegación.

### 5.1 Carpeta lib/models/

Contiene la **lógica del juego** y las entidades principales.

Aquí se definen:

- La lógica de la serpiente (movimiento, crecimiento y colisiones)
- Direcciones de movimiento
- Generación y control de la comida

Esta carpeta **no contiene interfaz gráfica**, solo reglas del juego.

### 5.2 Carpeta lib/widgets/

Contiene **componentes visuales reutilizables**.

Ejemplos:

- Segmentos de la serpiente
- Tablero del juego
- Botones personalizados
- Barra superior (Bar)

Estos widgets no representan pantallas completas, sino piezas que se reutilizan en distintas partes de la aplicación.

### 5.3 Carpeta lib/screens/

Contiene las **pantallas completas** de la aplicación.

Cada archivo representa una vista principal:

- Pantalla principal / menú
- Pantalla de juego
- Pantalla de fin de partida

Las pantallas combinan widgets y lógica para construir la experiencia del usuario.

### 5.4 Carpeta lib/providers/

Contiene los **gestores de estado** de la aplicación.

Se utiliza para:

- Controlar el tema (claro / oscuro)
- Compartir información global entre pantallas

### 5.5 Carpeta lib/routing/

Gestiona la **navegación entre pantallas**.

Define:

- Rutas de la aplicación
- Qué pantalla se muestra en cada URL
- Flujo de navegación entre juego, menú y pantalla final

### 5.6 Carpeta /lib/themes/

Contiene la configuración visual global:

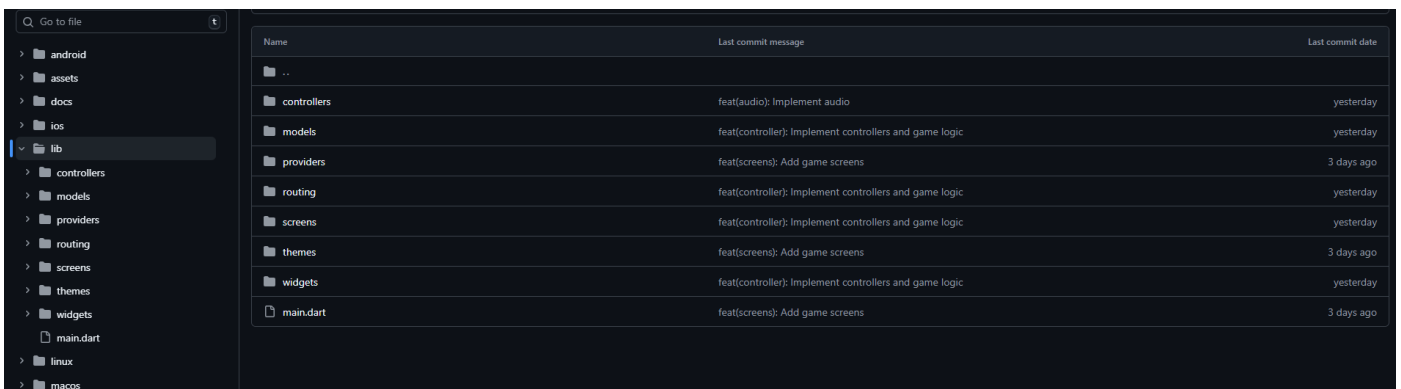
- Colores
- Temas claros y oscuros
- Estilos reutilizables

### 5.7 Carpeta assets/

Incluye los recursos multimedia del juego:

- assets/images/: imágenes del tablero, serpiente y comida
- assets/sounds/: sonidos del juego

Estos recursos están declarados en el archivo pubspec.yaml y son necesarios para la ejecución correcta.



Name	Last commit message	Last commit date
..		
controllers	feat(audio): Implement audio	yesterday
models	feat(controller): Implement controllers and game logic	yesterday
providers	feat(screens): Add game screens	3 days ago
routing	feat(controller): Implement controllers and game logic	yesterday
screens	feat(controller): Implement controllers and game logic	yesterday
themes	feat(screens): Add game screens	3 days ago
widgets	feat(controller): Implement controllers and game logic	yesterday
main.dart	feat(screens): Add game screens	3 days ago

## 5.8 Carpeta /lib/controller/

Es el cerebro del juego teniendo las siguiente responsabilidades:

- Mantener el estado global del juego
- Controlar el bucle (Timer)
- Detectar colisiones
- Actualizar puntuación
- Gestionar Game Over
- Reproducir sonidos
- Notificar cambios a la UI (ChangeNotifier)

Es el **único punto que conecta lógica y UI**.

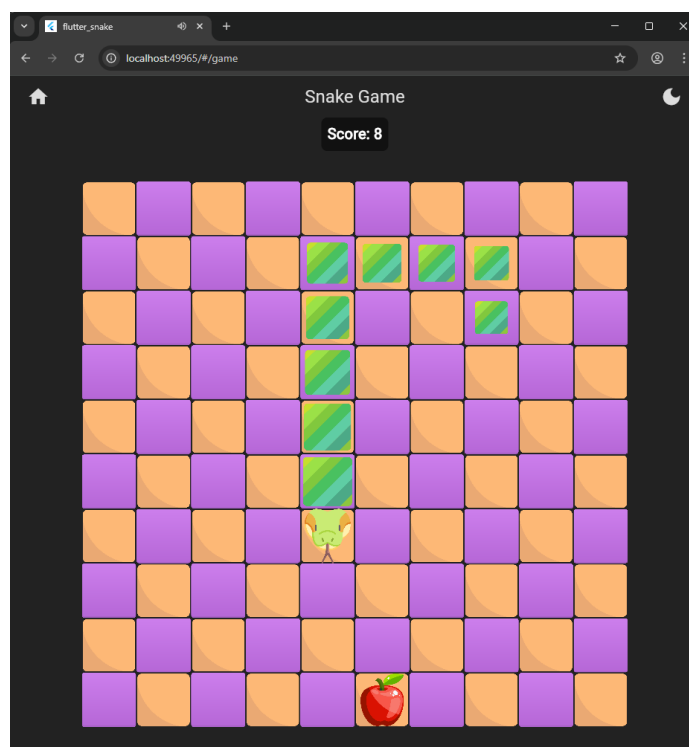
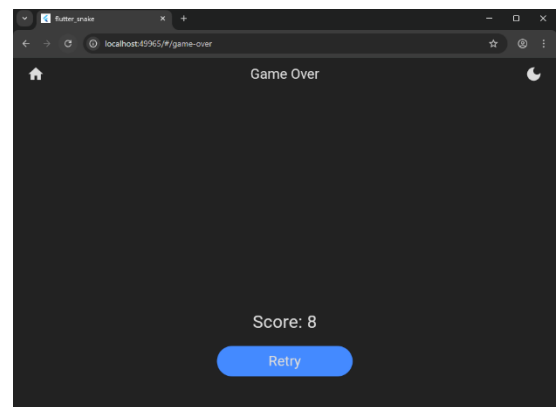
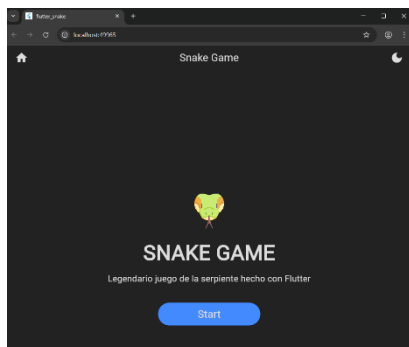
## 6. Ejecución del proyecto

Desde la carpeta raíz del proyecto: flutter run

Flutter detectará automáticamente los dispositivos disponibles y ejecutará la aplicación.

Opciones adicionales:

- Ejecutar en navegador: flutter run -d chrome



## 7. Funcionamiento general del juego

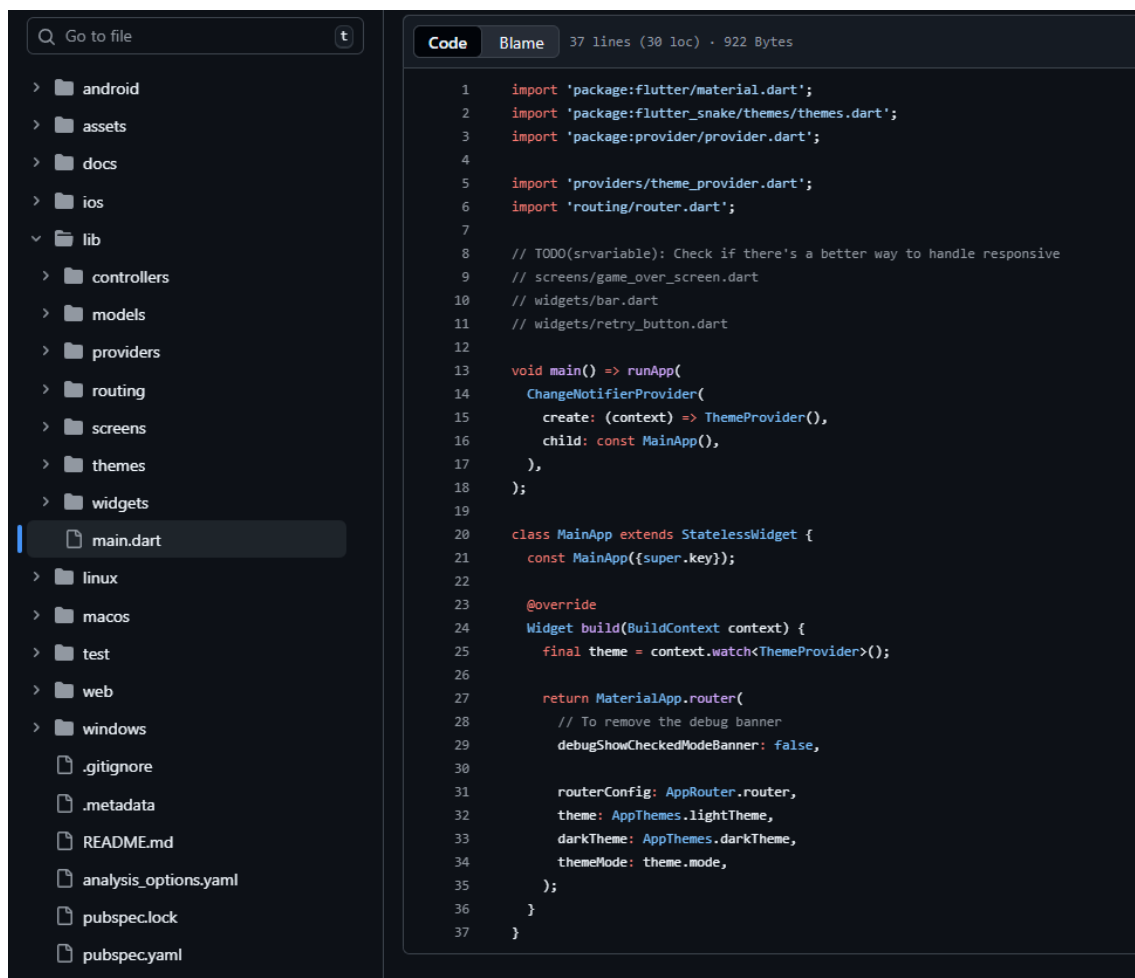
- La lógica del juego se ejecuta desde los modelos
- El tablero y los elementos visuales se renderizan mediante widgets
- Las pantallas controlan el flujo del juego
- Los sonidos y eventos se gestionan desde los controladores

Esta separación permite mantener el proyecto ordenado, escalable y fácil de mantener.

El punto de entrada de la aplicación es el fichero **main.dart**, ubicado en la carpeta lib/.

Este fichero se encarga de inicializar Flutter, configurar los proveedores globales de la aplicación (como el gestor de temas) y arrancar el widget raíz.

La navegación entre pantallas y la carga de la pantalla principal se delegan al sistema de rutas definido en **router.dart**, manteniendo una separación clara entre configuración global, navegación y lógica de negocio.



```
Code Blame 37 lines (30 loc) · 922 Bytes

1  import 'package:flutter/material.dart';
2  import 'package:flutter_snake/themes/themes.dart';
3  import 'package:provider/provider.dart';
4
5  import 'providers/theme_provider.dart';
6  import 'routing/router.dart';
7
8  // TODO(srvariable): Check if there's a better way to handle responsive
9  // screens/game_over_screen.dart
10 // widgets/bar.dart
11 // widgets/retry_button.dart
12
13 void main() => runApp(
14   ChangeNotifierProvider(
15     create: (context) => ThemeProvider(),
16     child: const MainApp(),
17   ),
18 );
19
20 class MainApp extends StatelessWidget {
21   const MainApp({super.key});
22
23   @override
24   Widget build(BuildContext context) {
25     final theme = context.watch<ThemeProvider>();
26
27     return MaterialApp.router(
28       // To remove the debug banner
29       debugShowCheckedModeBanner: false,
30
31       routerConfig: AppRouter.router,
32       theme: AppThemes.lightTheme,
33       darkTheme: AppThemes.darkTheme,
34       themeMode: theme.mode,
35     );
36   }
37 }
```

El controlador del juego (game\_controller), se inyecta a nivel de pantalla, evitando que la lógica específica del juego dependa del punto de entrada global de la aplicación.

## 8. Problemas comunes

### El proyecto no compila

- Ejecutar flutter doctor
- Comprobar que Flutter está correctamente instalado

### No se muestran imágenes o sonidos

- Verificar la carpeta assets
- Ejecutar de nuevo flutter pub get

### Error de dependencias

- Eliminar la carpeta .dart\_tool
- Ejecutar nuevamente flutter pub get

**No tener git instalado para copiar el framework Flutter desde**  
<https://github.com/flutter/flutter>