Flutter para principiantes

Introducción

Flutter es un marco de aplicación de software de código abierto que se utiliza para desarrollar aplicaciones móviles, web y de escritorio. Está creado por Google y se desarrolla con un lenguaje de programación llamado Dart. La principal característica de Flutter es que permite a los desarrolladores crear aplicaciones móviles de aspecto nativo en una fracción de tiempo, lo que permite ahorrar tiempo y esfuerzo a los desarrolladores.

Ventajas de Flutter

Flutter ofrece a los desarrolladores una variedad de ventajas. Primero, es un marco con un código fuente abierto, lo que significa que los desarrolladores pueden acceder al código fuente y modificarlo de acuerdo a sus necesidades. Además, Flutter ofrece un alto rendimiento y una variedad de herramientas y bibliotecas para ayudar a los desarrolladores a crear sus aplicaciones.

Una plataforma de desarrollo de aplicaciones práctica

Flutter ofrece una plataforma de desarrollo de aplicaciones práctica que permite a los desarrolladores crear aplicaciones de forma rápida y eficiente. Esto se debe a que las herramientas proporcionadas por Flutter hacen que el proceso de desarrollo sea mucho más sencillo. Además, Flutter ofrece una variedad de widgets, lo que permite a los desarrolladores crear aplicaciones de aspecto nativo más rápidamente.

La funcionalidad de Flutter

Flutter también ofrece una variedad de funcionalidades. Esto incluye una variedad de plantillas y herramientas para ayudar a los desarrolladores a crear sus aplicaciones. Además, el marco también ofrece una funcionalidad de pruebas y diagnóstico, lo que permite a los desarrolladores mejorar la calidad y rendimiento de sus aplicaciones.

Widgets

En esta guía, vamos a analizar algunos de los widgets básicos y más utilizados de Flutter para diseñar las interfaces de usuario básicas, por lo que no vamos a entrar en todos los widgets, sino solo en algunos de los más utilizados.

Entonces, para comenzar con Flutter, necesitamos conocer los conceptos básicos de Dart.

Es posible que hayas escuchado en alguna parte que todo en Flutter es un widget.

Entonces, ¿qué es un widget en Flutter?

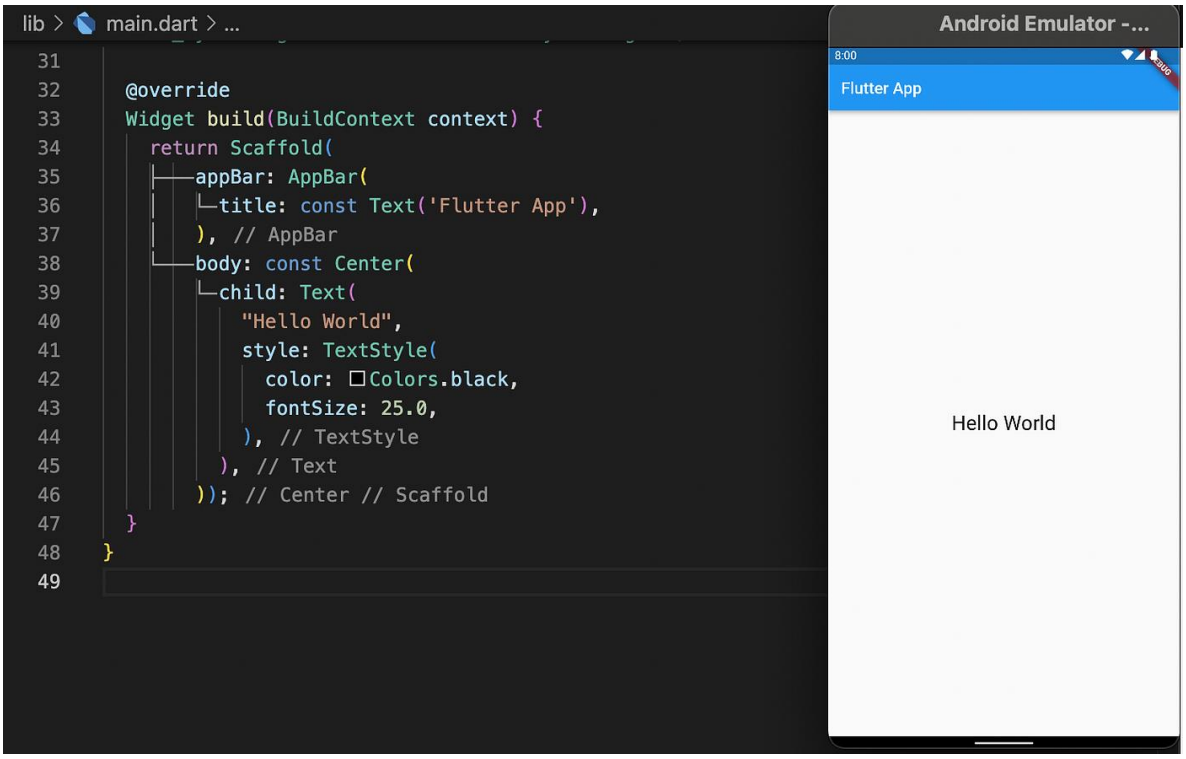
Un widget es básicamente un bloque de construcción de la aplicación Flutter. Es una descripción de una parte de la interfaz de usuario, pero no se limita a ella. Un widget puede mostrar

algo, puede ayudar a definir el diseño, puede ayudar con el diseño, puede manejar la interacción del usuario, etc.

Básicamente, la acumulación de diferentes tipos de widgets en un conjunto fijo de reglas constituye la aplicación Flutter. Por lo tanto, el término, "todo es widget en Flutter".

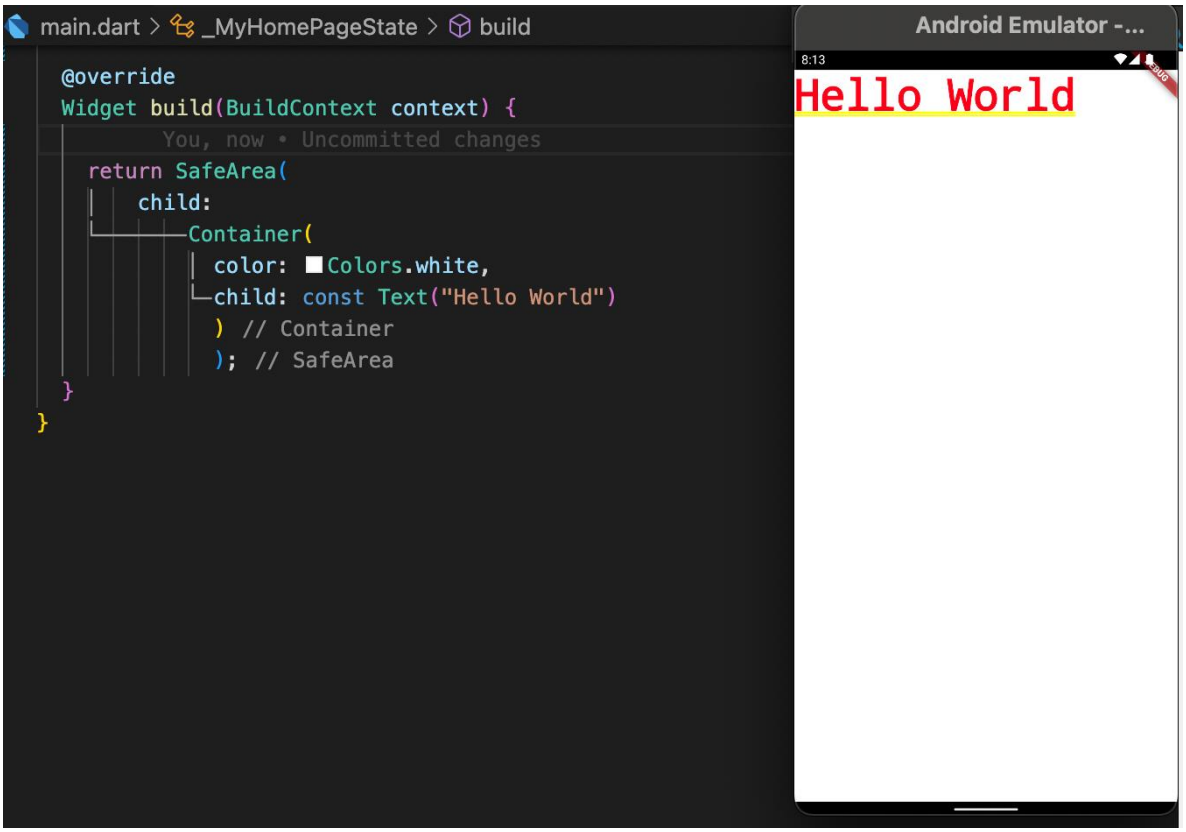
Scaffold

Scaffold es un widget que se utiliza para implementar la estructura básica de diseño de la interfaz de. Este es uno de los widgets más importantes de Flutter, ya que contiene el diseño principal de la aplicación. Este widget puede ocupar toda la pantalla del dispositivo. En otras palabras, podemos decir que es el principal responsable de crear una base para la pantalla de la aplicación en la que se mantienen y representan los widgets secundarios. Flutter proporciona muchos widgets para mostrar interfaz de usuario tales como: Drawer, SnackBar, BottomNavigationBar, AppBar, FloatingActionButton y muchos más. Este widget básicamente ayuda a crear el diseño básico que debe tener una aplicación.



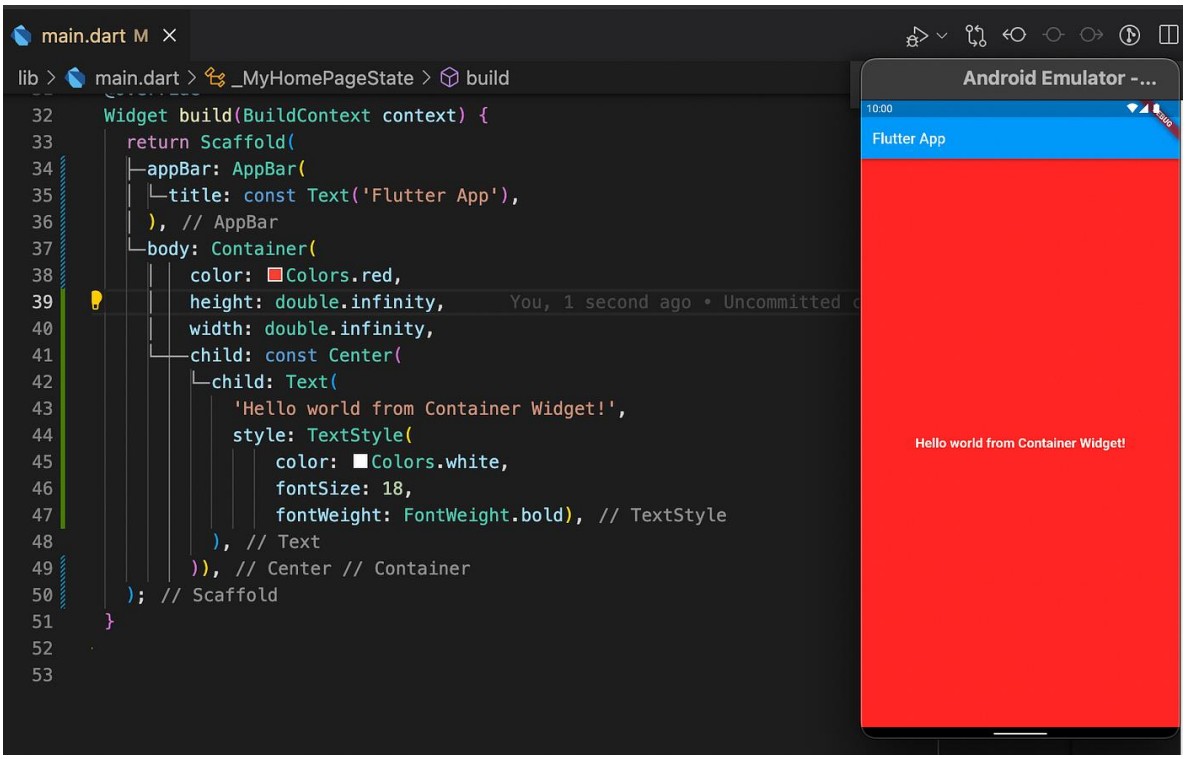
Área segura o SafeArea

El widget de SafeArea en Flutter se usan para no superponerlos con los elementos del móvil, como el estado de la batería, el indicador wifi y otras cosas por el estilo. SafeArea le ayuda a evitar esta superposición de la barra de estado.



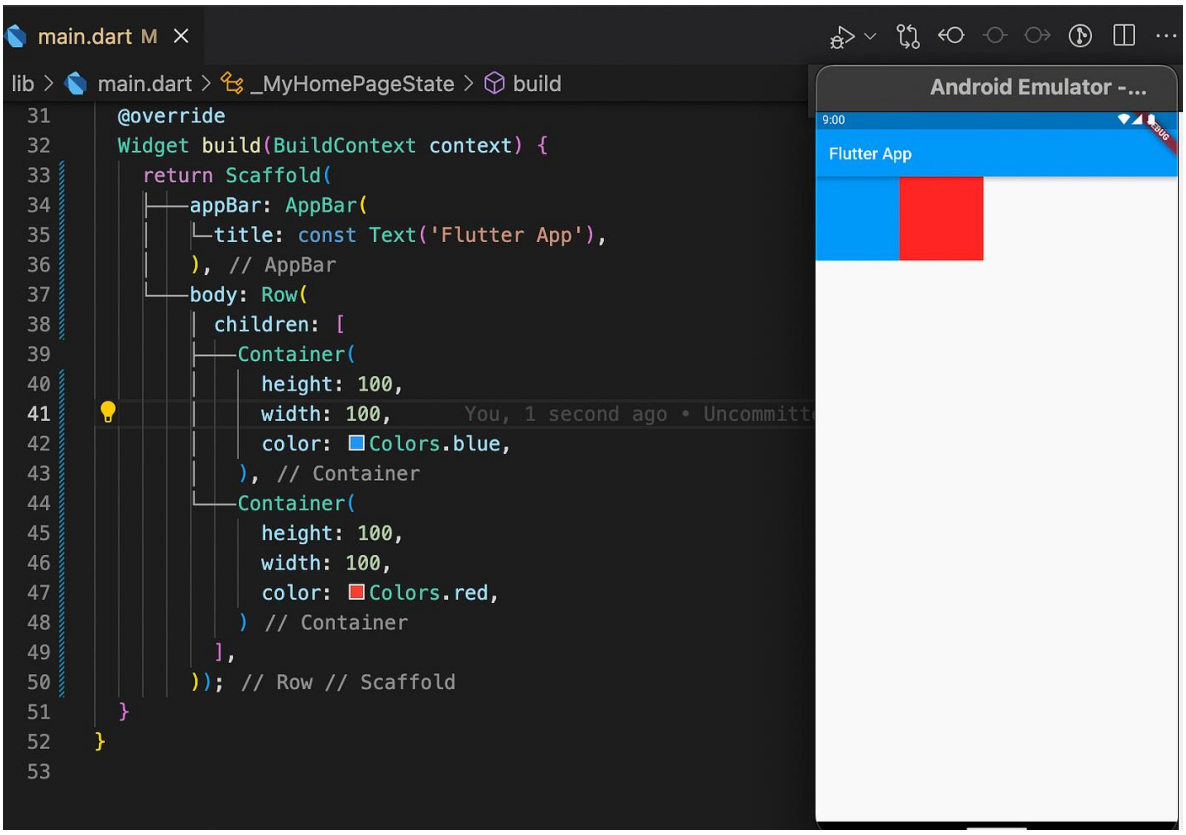
Container

Este es un widget principal que puede contener múltiples widgets secundarios y administrarlos de manera eficiente a través del ancho, la altura, el relleno, el color de fondo, etc. Un widget de contenedor es lo mismo que la etiqueta en html. Combina la pintura, el posicionamiento y el tamaño comunes de los widgets secundarios. También es una clase para almacenar uno o más widgets y posicionarlos en la pantalla según nuestras necesidades. En general, es similar a una caja para almacenar contenido. Permite muchos atributos al usuario para decorar sus widgets secundarios, como el uso de margin, que separa el contenedor con otros contenidos.



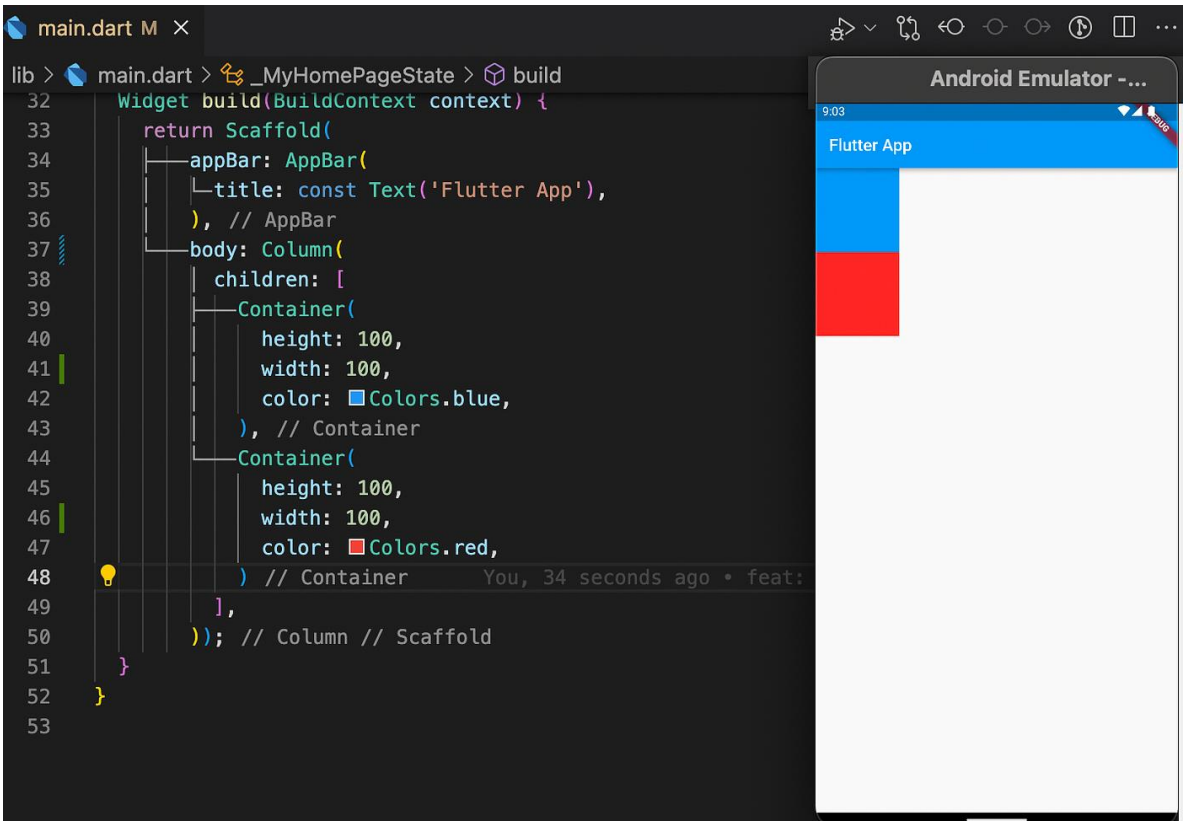
Row

Row (fila) es uno de los widgets más esenciales en Flutter que permite a los desarrolladores alinear “children” (hijos) horizontalmente según nuestras necesidades.



Column

También es uno de los widgets más esenciales en Flutter que permite a los desarrolladores alinear hijos verticalmente según nuestras necesidades.



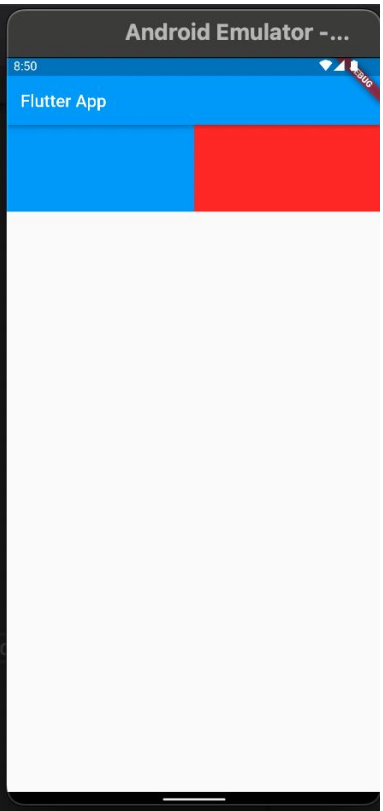
Expanded

Este widget en Flutter ayuda a ocupar todo el espacio restante dentro de los widgets de Row (Fila) o Column (Columna). Con su propiedad flexible, puede controlar cuánto espacio debe ocupar el widget secundario.

main.dart M

lib > main.dart > _MyHomePageState > build

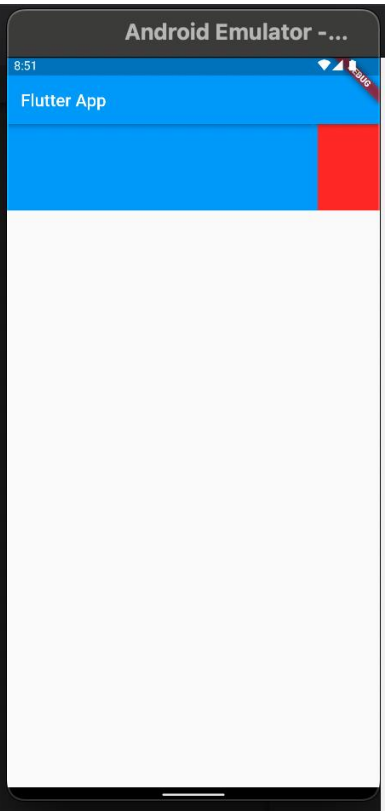
```dart
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Flutter App'),
    ), // AppBar
    body: Row(
      children: [
        Expanded(
          flex: 1,
          child: Container(
            height: 100,
            color: Colors.blue,
          )), // Container // Expanded
        Expanded(
          flex: 1,
          child: Container(
            height: 100,
            color: Colors.red,
          )) // Container // Expanded
      ],
    )); // Row // Scaffold
}
}
```



main.dart M

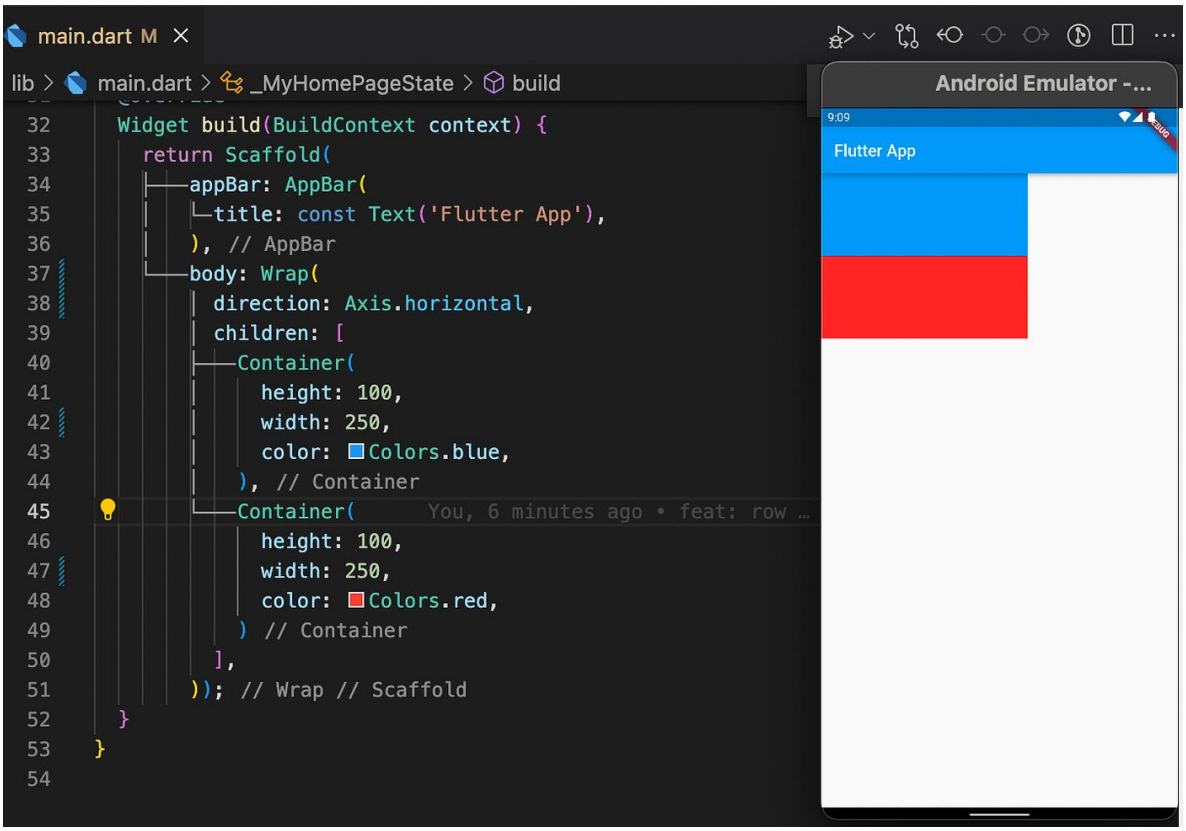
lib > main.dart > ...

```dart
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Flutter App'),
    ), // AppBar
    body: Row(
      children: [
        Expanded(
          flex: 5,
          child: Container(
            height: 100,
            color: Colors.blue,
          )), // Container // Expanded
        Expanded(
          flex: 1,
          child: Container(
            height: 100,
            color: Colors.red,
          )) // Container // Expanded
      ],
    )); // Row // Scaffold
}
}
```



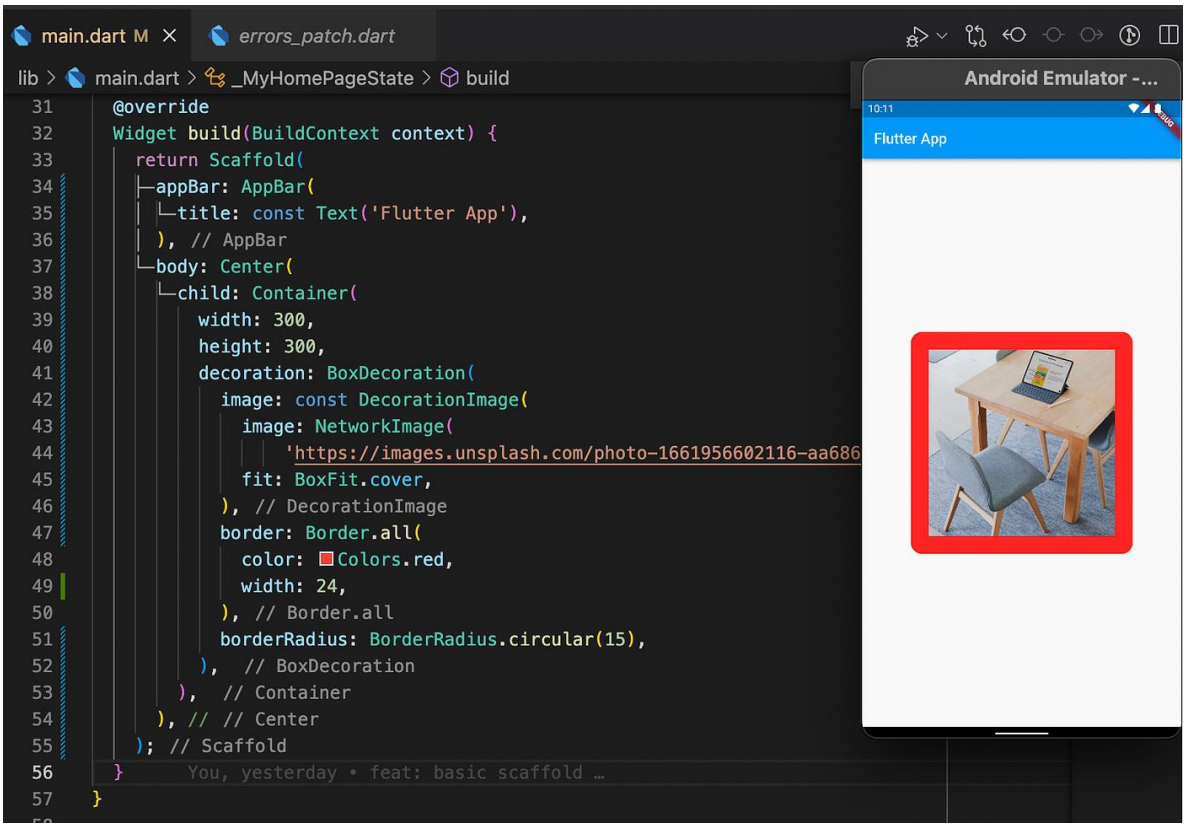
Wrap

A menudo recibimos una advertencia de desbordamiento al alinear los widgets en Flutter. Este error ocurre cuando el contenido dentro de una Columna/Fila excede el tamaño máximo permitido. Wrap le permite colocar widgets en una nueva línea cuando superan el tamaño máximo de la pantalla.



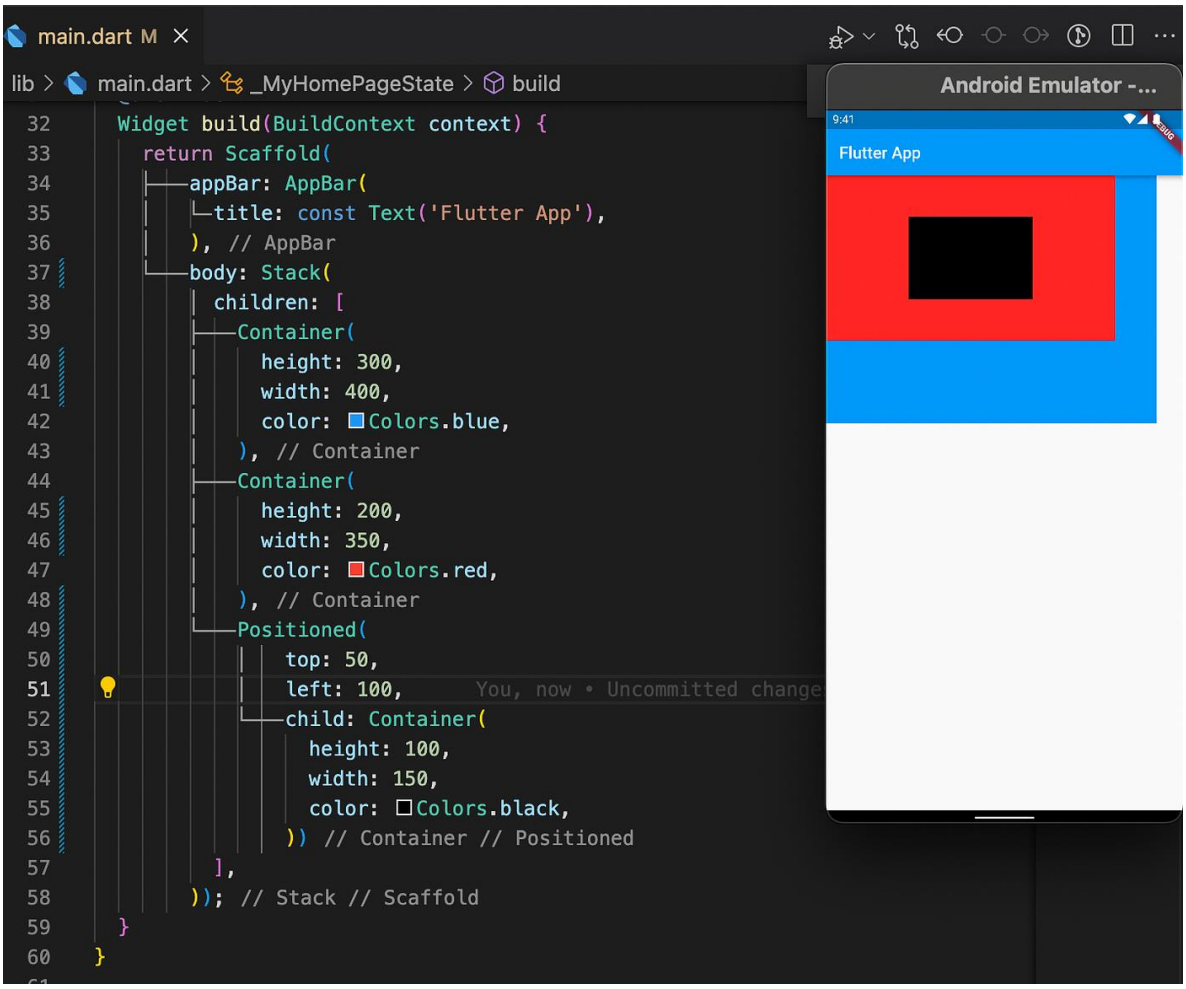
BoxDecoration

Este widget describe básicamente cómo se debe pintar un cuadro en la pantalla. La forma de la caja puede ser rectangular, cuadrada o circular. Viene con un montón de propiedades, podemos agregar una imagen dentro, agregar un radio al borde según el tamaño del radio y proyectar una sombra en el cuadro.



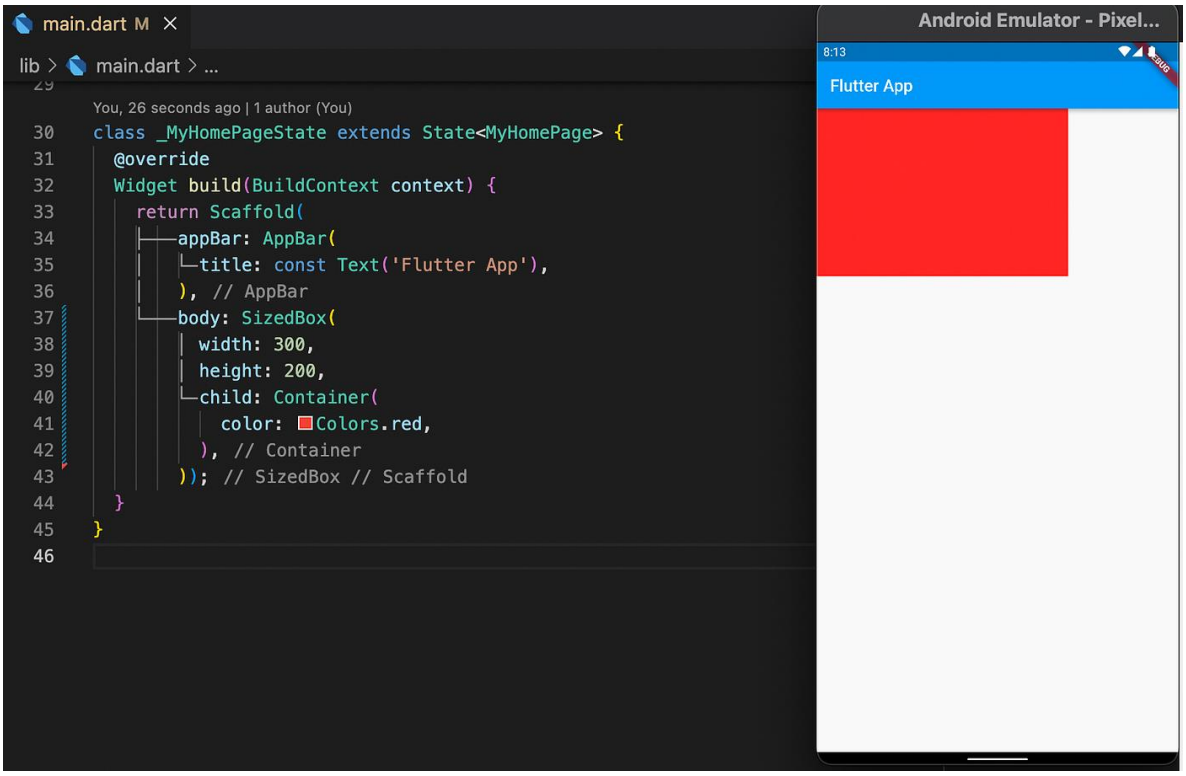
Stack

Este widget en Flutter ayuda a apilar widgets uno encima del otro. Al usar el widget Positioned, podemos posicionar elementos dentro de la pila. Su uso es similar al valor absoluto de la propiedad display css.



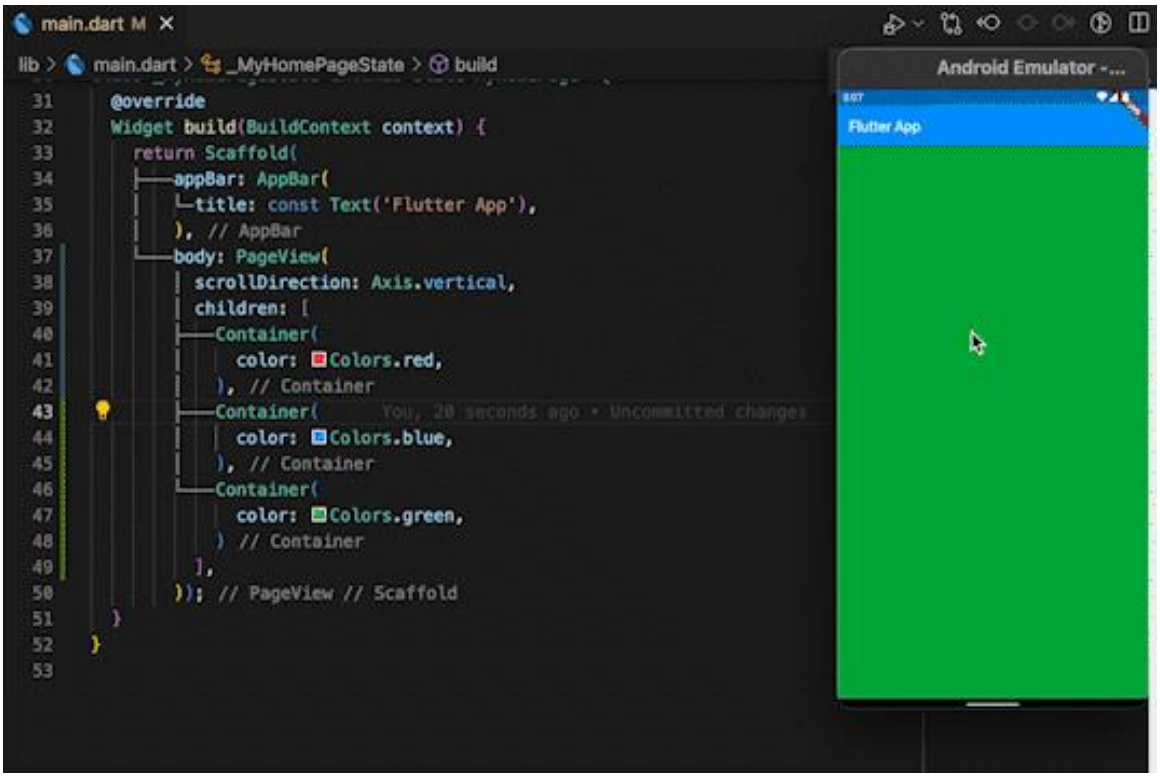
SizedBox

El widget SizedBox se usa cuando necesita crear un cuadro con un tamaño específico (ancho / alto). Este widget permite especificar la dimensión que debe tener el elemento hijo. También puede especificar una dimensión infinita (double.infinite) si desea que ese elemento secundario ocupe todo el tamaño permitido. Otro uso de SizedBox es cuando necesita algo de espacio entre los elementos Columna / Fila.



PageView

Este widget ayuda a crear una lista desplazable de páginas. También puede decidir la orientación de desplazamiento con la propiedad scrollDirection . También tiene muchas otras propiedades útiles como pageSnapping, scrollPhysics, etc. que ayudan a personalizar la transición de las páginas.

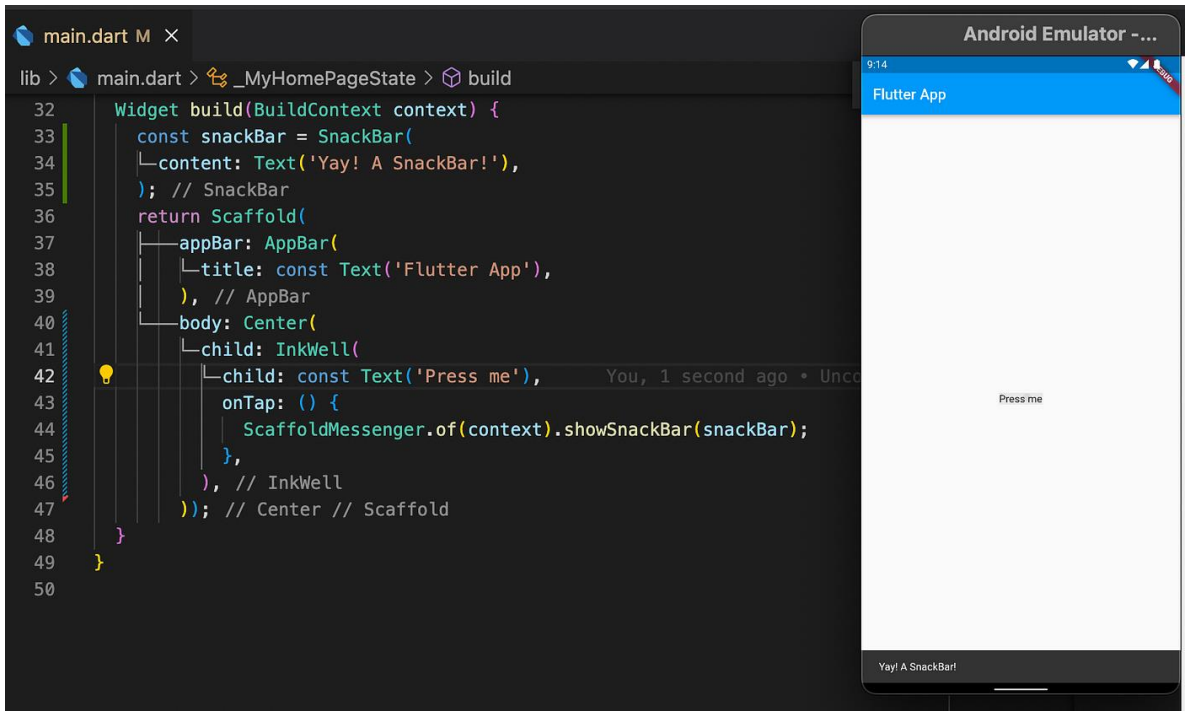


SnackBar

SnackBar es un widget que muestra un mensaje rápido dentro de la aplicación que indica brevemente al usuario cuándo sucedió algo.

InkWell

InkWell responde a la acción táctil realizada por el usuario. InkWell responderá cuando el usuario haga clic en el botón. Hay tantos gestos como tocar dos veces, presionar prolongadamente, tocar hacia abajo, etc. A continuación, se encuentran las tantas propiedades de este widget. Podemos establecer el radio del widget del InkWell usando el radio y también el radio del borde usando borderRadius. Podemos dar el color de salpicadura usando splashColor y podemos hacer muchas cosas.



Como puede ver, Flutter tiene muchos conjuntos diferentes de widgets que nos ayudan a personalizar la aplicación. Aprender cuándo y cómo usarlos es la clave en Flutter.