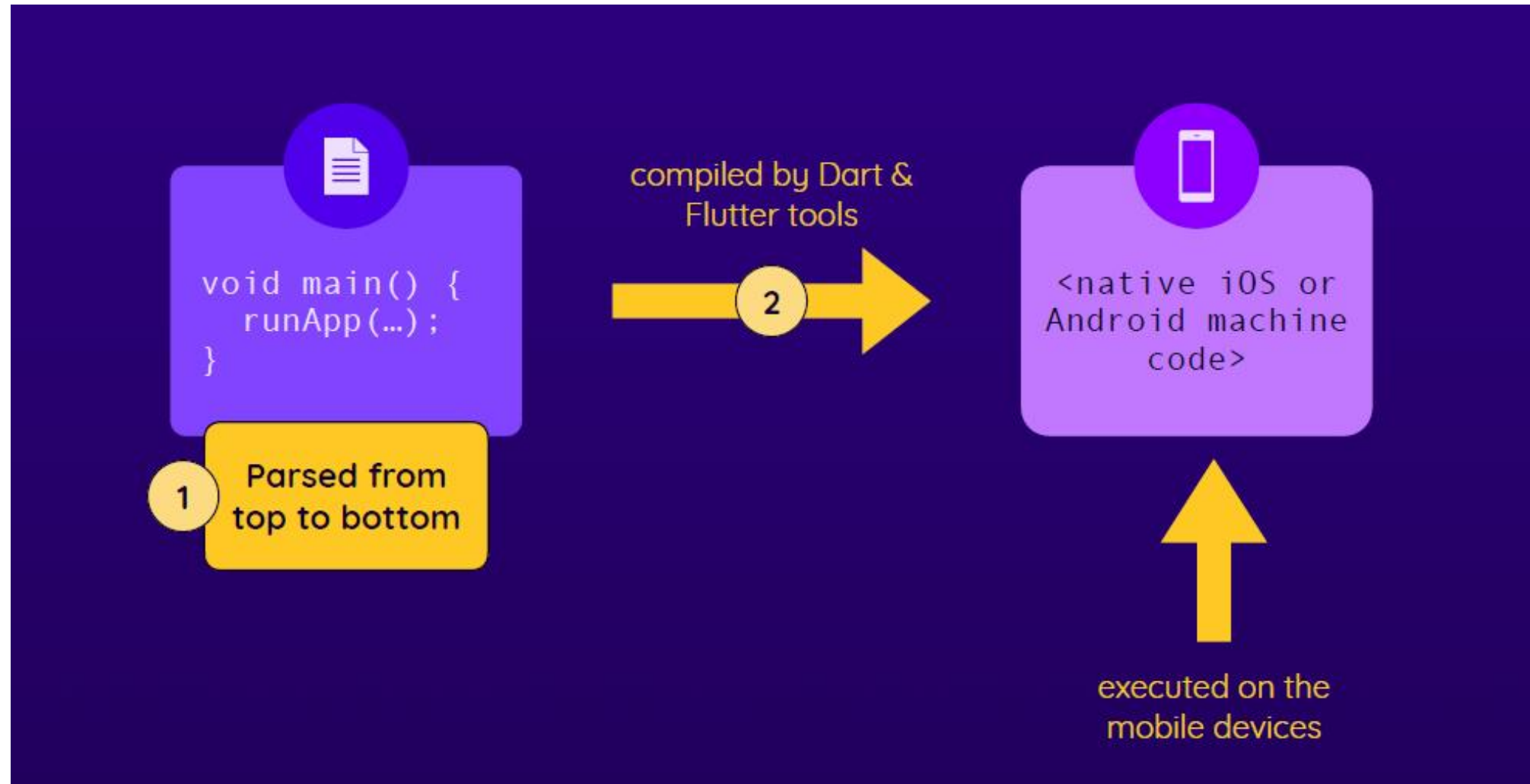


# Mobil Programlama ve Tasarım

Hafta 2

# Dart & Flutter Kodu Derlenir



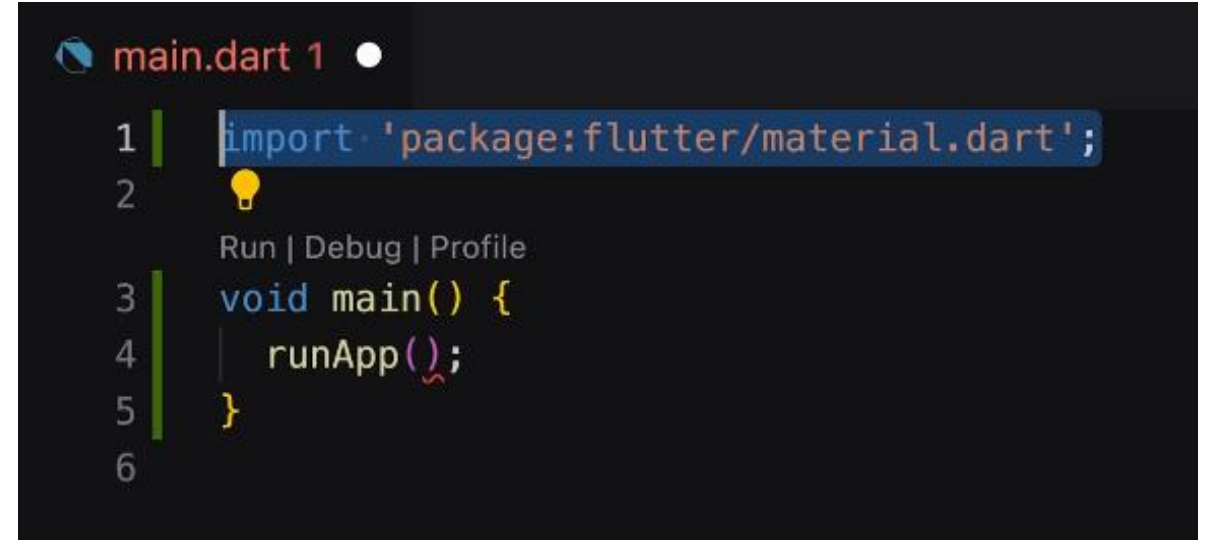
# Başlangıç



A screenshot of an IDE window titled 'main.dart 1'. The code is as follows:

```
1 |  
2 | Run | Debug | Profile  
3 | void main() {  
4 |   runApp();  
5 | }  
6 |
```

The code is in a dark theme. The 'runApp()' method call on line 4 has a red squiggly underline underneath it, indicating a warning or error.



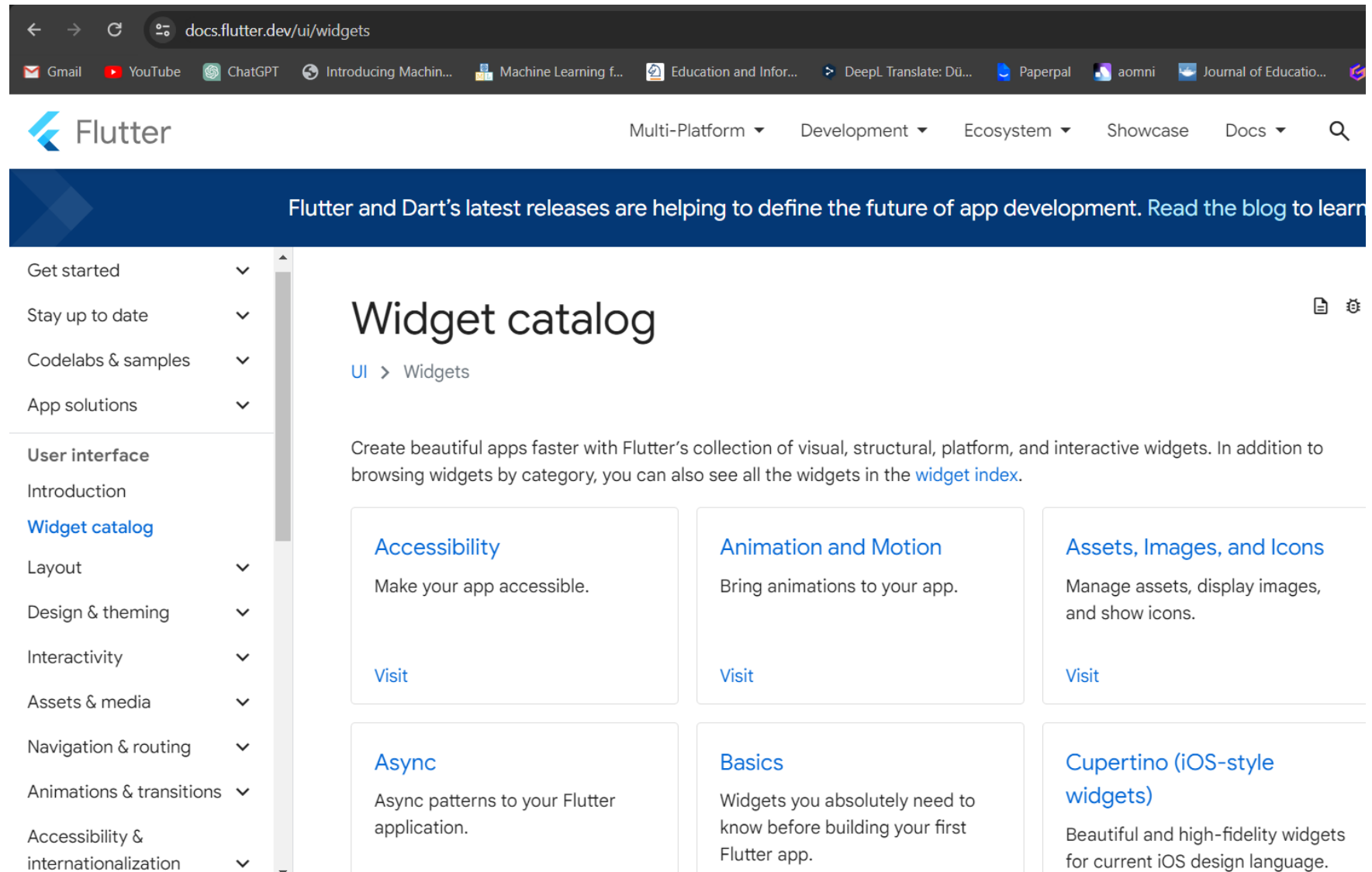
A screenshot of an IDE window titled 'main.dart 1'. The code is as follows:

```
1 | import 'package:flutter/material.dart';  
2 |  
3 | Run | Debug | Profile  
4 | void main() {  
5 |   runApp();  
6 | }
```

The code is in a dark theme. The 'import' statement on line 1 is highlighted with a blue background. A yellow lightbulb icon is visible on line 2, indicating a suggestion or tip.

<https://m3.material.io/>

# Widget catalog



The screenshot shows the Flutter Widget Catalog page in a web browser. The browser's address bar displays 'docs.flutter.dev/ui/widgets'. The page features a dark blue header with the Flutter logo and navigation links for Multi-Platform, Development, Ecosystem, Showcase, and Docs. A blue banner below the header contains the text: 'Flutter and Dart's latest releases are helping to define the future of app development. Read the blog to learn more.' On the left, a sidebar lists various sections, with 'Widget catalog' highlighted in blue. The main content area is titled 'Widget catalog' and includes a breadcrumb 'UI > Widgets'. It provides an introduction to Flutter's widget collection and a grid of category cards. Each card has a title, a brief description, and a 'Visit' link.

docs.flutter.dev/ui/widgets

Flutter

Multi-Platform Development Ecosystem Showcase Docs

Flutter and Dart's latest releases are helping to define the future of app development. Read the blog to learn more.

Get started  
Stay up to date  
Codelabs & samples  
App solutions

User interface  
Introduction  
**Widget catalog**  
Layout  
Design & theming  
Interactivity  
Assets & media  
Navigation & routing  
Animations & transitions  
Accessibility & internationalization

## Widget catalog

UI > Widgets

Create beautiful apps faster with Flutter's collection of visual, structural, platform, and interactive widgets. In addition to browsing widgets by category, you can also see all the widgets in the [widget index](#).

### Accessibility

Make your app accessible.

[Visit](#)

### Animation and Motion

Bring animations to your app.

[Visit](#)

### Assets, Images, and Icons

Manage assets, display images, and show icons.

[Visit](#)

### Async

Async patterns to your Flutter application.

### Basics

Widgets you absolutely need to know before building your first Flutter app.

### Cupertino (iOS-style widgets)

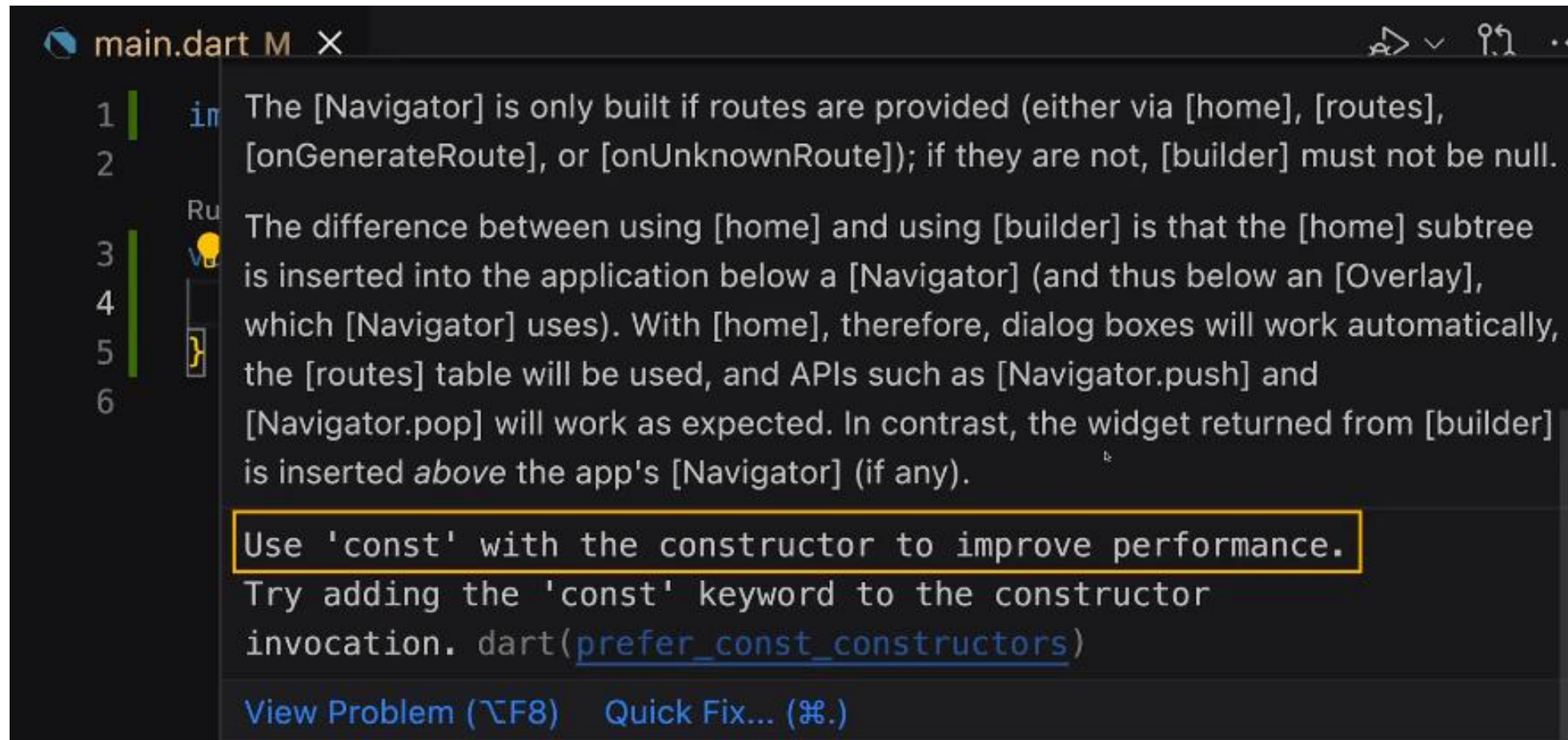
Beautiful and high-fidelity widgets for current iOS design language.

# Uygulama

main.dart M X

```
1  import 'package:flutter/material.dart';
2
   Run | Debug | Profile
3  void main() {
4      runApp(MaterialApp(home: Text('Hello World!')));
5  }
6  |
```

# İyileştirmeler / Mavi Dalgalı Çizgilerle Bildirimler



```
main.dart M X
```

```
1 in The [Navigator] is only built if routes are provided (either via [home], [routes],
2 [onGenerateRoute], or [onUnknownRoute]); if they are not, [builder] must not be null.
3
4 The difference between using [home] and using [builder] is that the [home] subtree
5 is inserted into the application below a [Navigator] (and thus below an [Overlay],
6 which [Navigator] uses). With [home], therefore, dialog boxes will work automatically,
the [routes] table will be used, and APIs such as [Navigator.push] and
[Navigator.pop] will work as expected. In contrast, the widget returned from [builder]
is inserted above the app's [Navigator] (if any).
```

Use 'const' with the constructor to improve performance.  
Try adding the 'const' keyword to the constructor invocation. dart([prefer\\_const\\_constructors](#))

View Problem (⌘F8) Quick Fix... (⌘.)

# Uygulama

main.dart M ●



```
1 | import 'package:flutter/material.dart';
```

```
2
```

Run | Debug | Profile

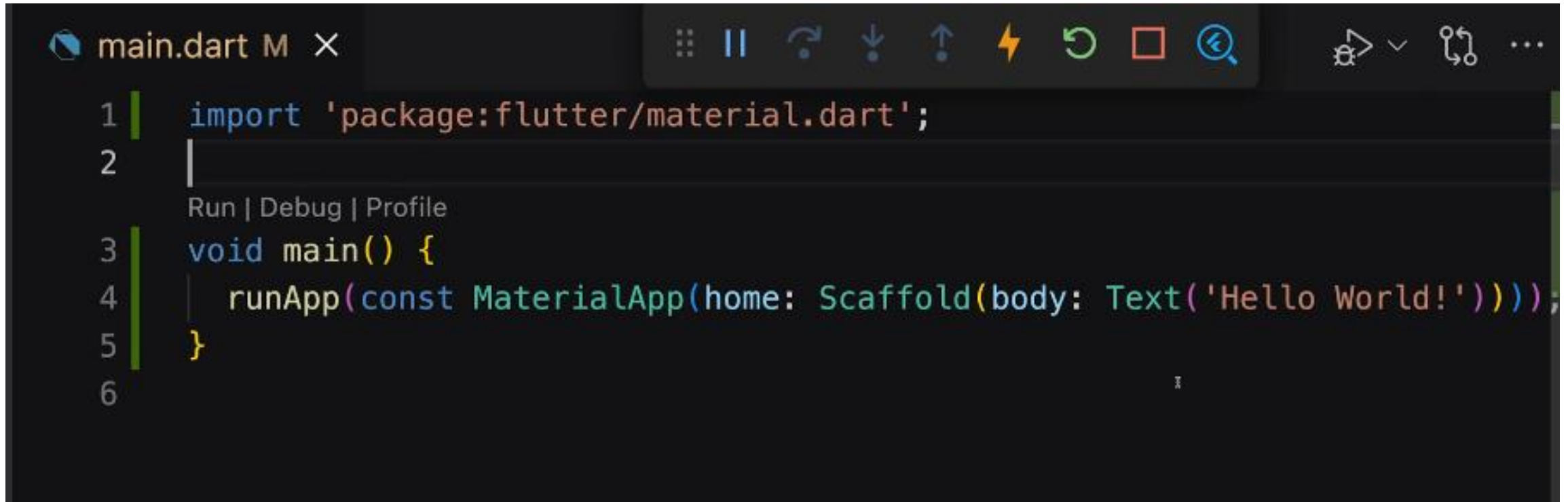
```
3 | void main() {
```

```
4 |   runApp(const MaterialApp(home: Text('Hello World!')));
```

```
5 | }
```

```
6 |
```

# Uygulama



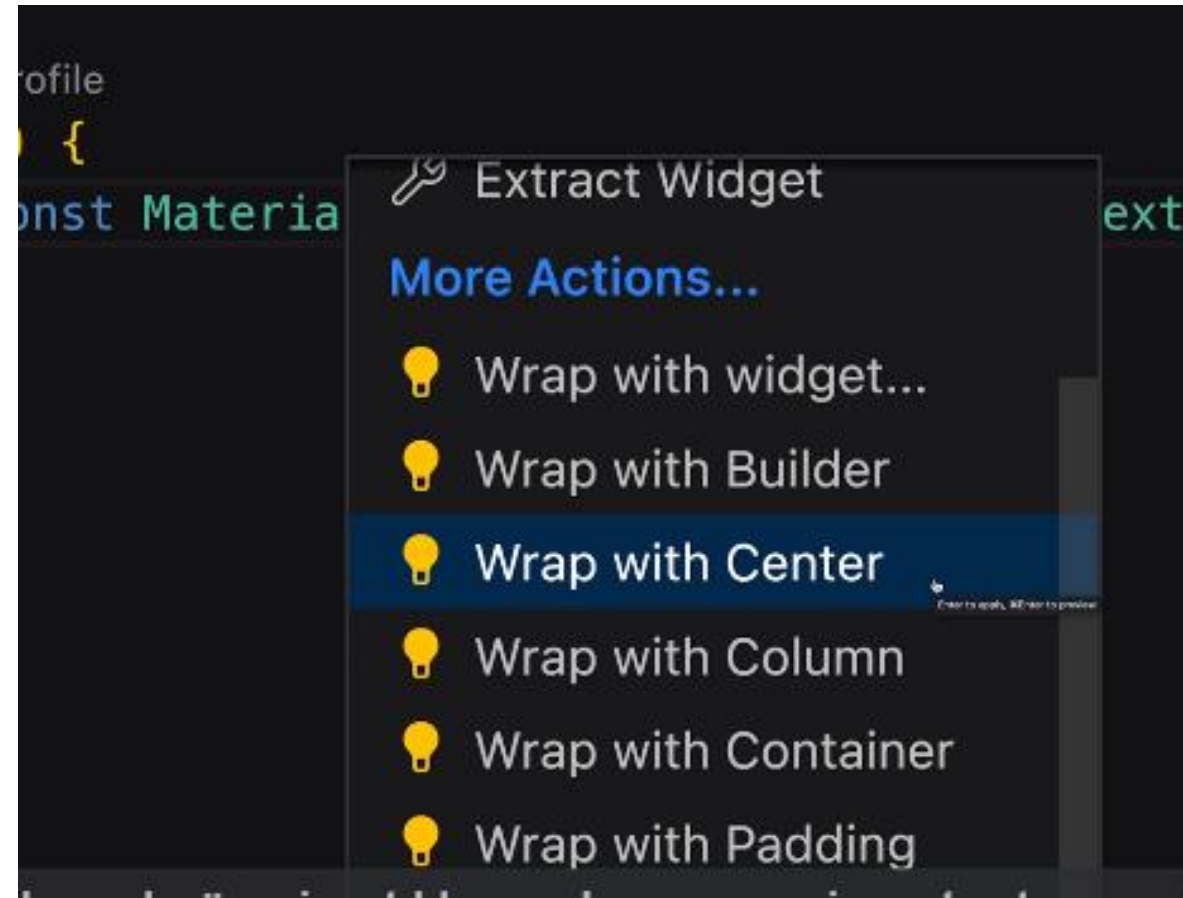
The image shows a screenshot of an IDE window titled "main.dart M". The code is written in Dart and defines a simple Flutter application. The code is as follows:

```
1 import 'package:flutter/material.dart';
2
3 void main() {
4   runApp(const MaterialApp(home: Scaffold(body: Text('Hello World!'))));
5 }
6
```

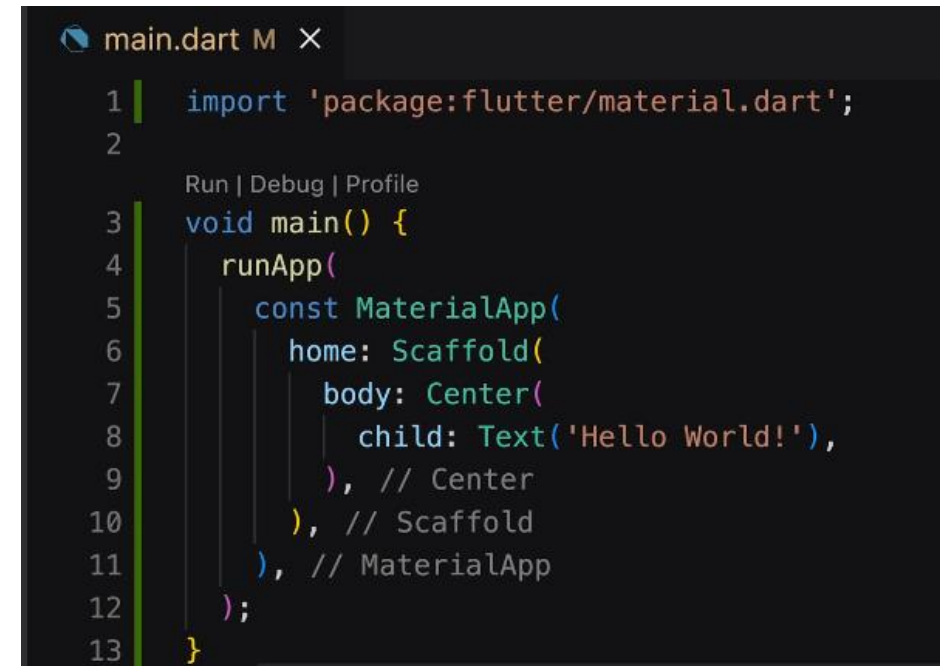
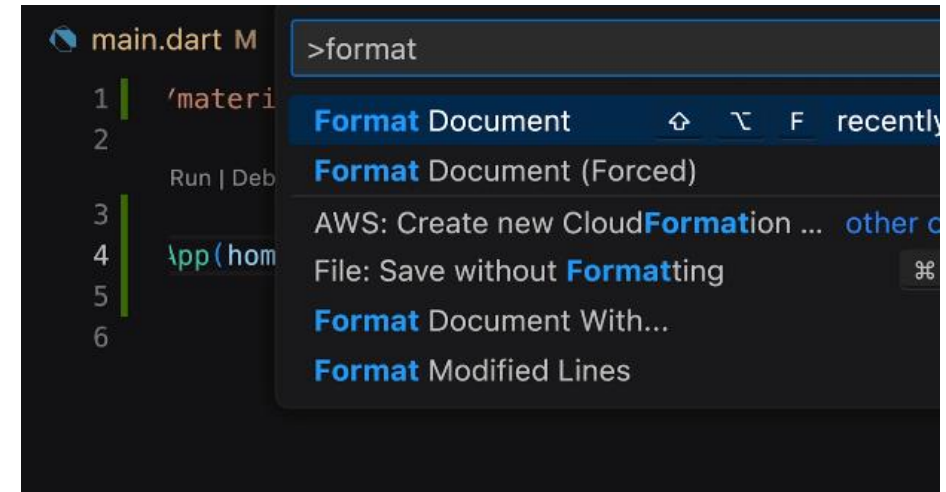
Line numbers 1 through 6 are visible on the left side of the code editor. The code uses standard Dart syntax for imports, function definitions, and widget instantiation. The IDE interface includes a toolbar with various icons for running, debugging, and profiling the application.



# Uygulama



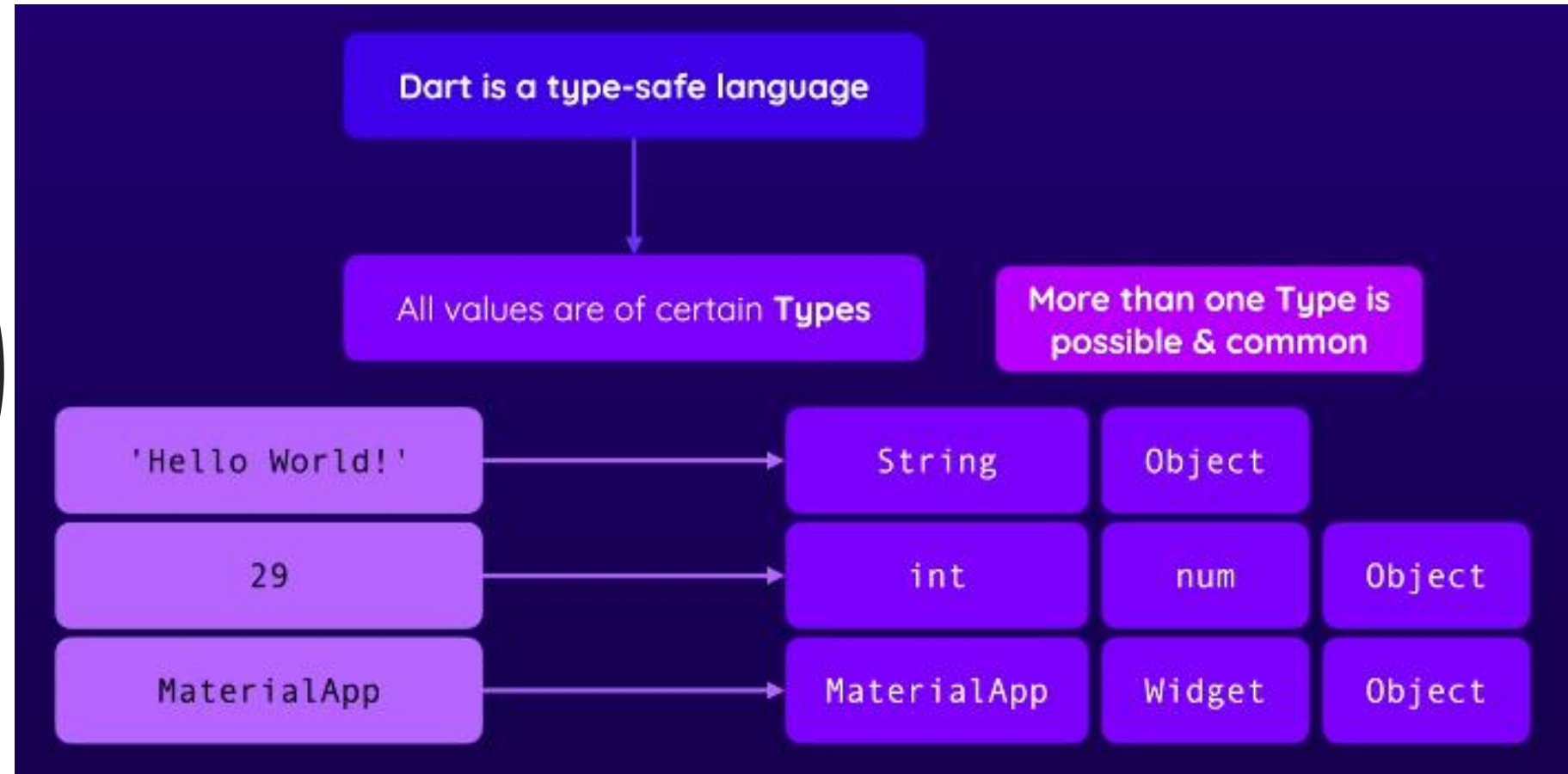
Parantezlerden  
sonra virgül  
koyarak kodları  
okunaklı hale  
getirme  
view->  
command  
palette->  
format document



Uygulamanın  
Buraya  
Kadarki  
Görüntüsü



# Dart Programlama Dilinde Türleri Anlamak



# Dartta Kullanılan Bazı Türler

int	Integer numbers	Numbers <b>without</b> decimal places	29, -15
double	Fractional numbers	Numbers <b>with</b> decimal places	3.91, -12.81
num	Integer or fractional numbers	Numbers <b>with or without</b> decimal places	15, 15.01, -2.91
String	Text	Text, wrapped with single or double quotes	'Hello World'
bool	Boolean values	true or false	true, false
Object	Any kind of object	The base type of all values	'Hi', 29, false

Background  
color

```
main.dart •
1  import 'package:flutter/material.dart';
2
   Run | Debug | Profile
3  void main() {
4      runApp(
5          const MaterialApp(
6              home: Scaffold(
7                  backgroundColor: Colors.deepPurple,
8                  body: Center(
9                      child: Text('Hello World!'),
10                 ), // Center
11             ), // Scaffold
12         ), // MaterialApp
13     );
14 }
```

```
home: Scaffold(
  backgroundColor:  Color.fromARGB(255, 63, 5, 120),
  body: Center(
```

Geçişli Renkte Bir  
Arka Plan  
Olmasını  
İstiyorsak:  
MaterialApp  
Widget'ındaki  
const silinmeli ve  
centera alınmalı

```
main.dart M X
1  import 'package:flutter/material.dart';
2
   Run | Debug | Profile
3  void main() {
4      runApp(
5          MaterialApp(
6              home: Scaffold(
7                  body: Container(
8                      child: const Center(
9                          child: Text('Hello World!'),
10                     ), // Center
11                 ), // Container
12             ), // Scaffold
13         ), // MaterialApp
14     );
```



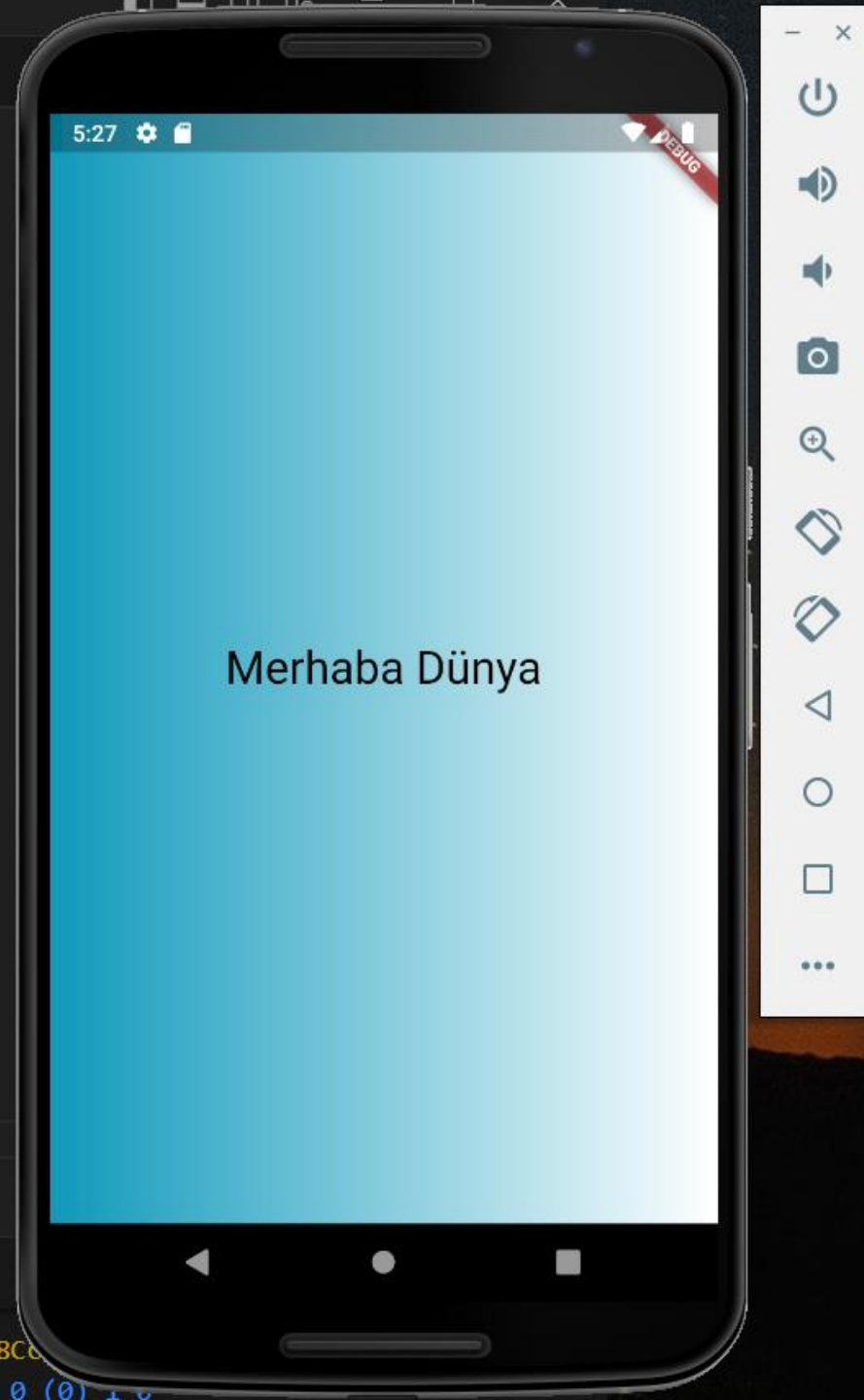
```
def idApp()
  home: Scaffold(
    body: Container(
      decoration: const BoxDecoration(
        gradient: LinearGradient(
          colors: [
            Color.fromARGB(255, 26, 2, 80),
            Color.fromARGB(255, 45, 7, 98),
          ],
        ), // LinearGradient
      ), // BoxDecoration
    child: const Center(
```



# Metne Özellikler Vermek

```
), // BoxDecoration  
child: const Center(  
  child: Text(  
    'Hello World!',  
    style: TextStyle(  
      color: Colors.white,  
      fontSize: 28,  
    ), // TextStyle  
  ), // Text  
), // Center
```

Uygulamanın  
Son Hali



# Özel Widget Oluşturma - Class

```
31 }  
32  
33 class GradientContainer {  
34  
35 }
```

```
32  
33 class GradientContainer extends StatelessWidget {  
34   @override  
35   Widget build(context) {  
36     return ;  
37   }  
38 }
```

```
33 class GradientContainer extends StatelessWidget {  
34  
35 }
```

```
33 class GradientContainer extends StatelessWidget {  
34   build()  
35 }
```

```
class GradientContainer extends StatelessWidget {  
  @override  
  Widget build() {}  
}
```

Ve return yanına container widget  
Ve içeriğini ekliyor, body widget'ında  
Bu class'ı çağırıyoruz.

## İyileştirmeler

```
class GradientContainer extends StatelessWidget {  
  const GradientContainer({super.key});  
  
  @override  
  Widget build(context) {
```

```
4 runApp(  
5   ⚡ const MaterialApp(  
6     home: Scaffold(  
7       body: GradientContainer(),  
8     ), // Scaffold  
9   ), // MaterialApp  
10  );  
11 }  
  
12  
13 class GradientContainer extends StatelessWidget {  
14   const GradientContainer({super.key});  
15 }
```

Kodları Dosyalara Ayıralım

# Uygulama

- Örnekte kullandığımız Text için başka bir Dart sayfasında yeni bir sınıf oluşturun ve ilgili sayfa ile bağlantısını yapın.