YouTube BM DERSLERİ

www.youtube.com/BMdersleri

MAKÜ
BURDUR MEHMET AKİF ERSOY ÜNİVERSİTESİ

Flutter ile Mobil Programlamaya Giriş

Flutter

10.HAFTA
VARLIK VE RESİM EKLEME
BUTONLAR, TEXTFIELD

Hazırlayan : **Zeynep İrem KESLER 1911404048**

Tarih : 17/05/2022

Sürüm : v1

Ders Yürütücüsü : Doç. Dr. İsmail KIRBAŞ

İÇİNDEKİLER

Varlık ve Resim Ekleme

- Aşağıdaki resim biçimleri desteklenir: JPEG, PNG, GIF, Hareketli GIF, WebP, Hareketli WebP, BMP ve WBMP. Ek biçimler, temel alınan platform tarafından desteklenebilir. Flutter, tanınmayan biçimlerin kodunu çözmek için platform API'sini çağırmaya çalışacak ve platform API'si görüntünün kodunun çözülmesini destekliyorsa, Flutter onu oluşturabilecektir.
- Piksel yoğunluğuna duyarlı varlık çözünürlüğünü otomatik olarak gerçekleştirmek için, bir AssetImage kullanarak görüntüyü belirtin ve pencere öğesi ağacında Görüntü pencere öğesinin üzerinde bir MaterialApp , WidgetsApp veya MediaQuery pencere bileşeninin bulunduğundan emin olun .
- Resim , bu sınıftaki çeşitli alanların anlamlarını daha ayrıntılı olarak açıklayan paintImage kullanılarak boyanmıştır .

Varlık ve Resim Ekleme

Varsayılan kurucu , internetten bir resim görüntülemek için NetworkImage gibi herhangi bir ImageProvider ile kullanılabilir .

```
const Image(
  image: NetworkImage('https://flutter.github.io/assets-for-api-docs/assets/widgets/owl.jpg'),
)
```

Varlık ve Resim Ekleme

Görüntü Widget'ı ayrıca kolaylık sağlamak için farklı görüntü türlerini görüntülemek için çeşitli oluşturucular sağlar. Bu örnekte, internetten bir görüntü görüntülemek için Image.network yapıcısını kullanın.

```
Image.network('https://flutter.github.io/assets-for-api-docs/assets/widgets/owl-2.jpg')
```

Varlık ve Resim Ekleme

Görüntü İşlem Widget'ları:

`Image.asset`

Varlık paketindeki resmi görüntülemek için kullanılır.

Implementation

```
Image.asset(
  String name, {
  Key? key,
  AssetBundle? bundle,
  this.frameBuilder,
  this.errorBuilder,
  this.semanticLabel,
  this.excludeFromSemantics = false,
  double? scale,
  this.width,
  this.height,
  this.color,
  this.opacity,
  this.colorBlendMode,
  this.fit,
  this.alignment = Alignment.center,
  this.repeat = ImageRepeat.noRepeat,
  this.centerSlice,
  this.matchTextDirection = false,
  this.gaplessPlayback = false,
  this.isAntiAlias = false,
  String? package,
  this.filterQuality = FilterQuality.low,
  int? cacheWidth,
  int? cacheHeight,
}) : image = ResizeImage.resizeIfNeeded(
       cacheWidth,
       cacheHeight,
       scale != null
         ? ExactAssetImage(name, bundle: bundle, scale: scale, package: package)
         : AssetImage(name, bundle: bundle, package: package),
     ),
     loadingBuilder = null,
     assert(alignment != null),
     assert(repeat != null),
     assert(matchTextDirection != null),
     assert(cacheWidth == null || cacheWidth > 0),
     assert(cacheHeight == null || cacheHeight > 0),
```

Varlık ve Resim Ekleme

`Image.file`

Bir dosyadaki görüntüyü görüntülemek için kullanılır.

Implementation

```
Image.file(
  File file, {
  Key? key,
  double scale = 1.0,
  this.frameBuilder,
  this.errorBuilder,
  this.semanticLabel,
  this.excludeFromSemantics = false,
  this.width,
  this.height,
  this.color,
  this.opacity,
  this.colorBlendMode,
  this.fit,
  this.alignment = Alignment.center,
  this.repeat = ImageRepeat.noRepeat,
  this.centerSlice,
  this.matchTextDirection = false,
  this.gaplessPlayback = false,
  this.isAntiAlias = false,
  this.filterQuality = FilterQuality.low,
  int? cacheWidth,
  int? cacheHeight,
}) :
     // FileImage is not supported on Flutter Web therefore neither this method.
     assert(
       !kIsWeb,
       'Image.file is not supported on Flutter Web. '
       'Consider using either Image.asset or Image.network instead.',
      ),
```

Varlık ve Resim Ekleme

`Image.memory`

Uint8List'ten resmi görüntülemek için kullanılır.

Implementation

```
Image.memory(
  Uint8List bytes, {
  Key? key,
  double scale = 1.0,
  this.frameBuilder,
  this.errorBuilder,
  this.semanticLabel,
  this.excludeFromSemantics = false,
  this.width,
  this.height,
  this.color,
  this.opacity,
  this.colorBlendMode,
  this.fit,
  this.alignment = Alignment.center,
  this.repeat = ImageRepeat.noRepeat,
  this.centerSlice,
  this.matchTextDirection = false,
  this.gaplessPlayback = false,
  this.isAntiAlias = false,
  this.filterQuality = FilterQuality.low,
  int? cacheWidth,
  int? cacheHeight,
}) : image = ResizeImage.resizeIfNeeded(cacheWidth, cacheHeight, MemoryImage(bytes, scale: scale)),
     loadingBuilder = null,
     assert(alignment != null),
     assert(repeat != null),
     assert(matchTextDirection != null),
     assert(cacheWidth == null || cacheWidth > 0),
     assert(cacheHeight == null || cacheHeight > 0),
     assert(isAntiAlias != null),
     super(key: key);
```

Varlık ve Resim Ekleme

`Image.network`

Bir URL'den resim görüntülemek için kullanılır.

Implementation

```
//
// TODO(garyq): We should eventually support custom decoding of network images
// on Web as well, see https://github.com/flutter/flutter/issues/42789.
Image.network(
  String src, {
  Key? key,
  double scale = 1.0,
  this.frameBuilder,
  this.loadingBuilder,
  this.errorBuilder,
  this.semanticLabel,
  this.excludeFromSemantics = false,
  this.width,
  this.height,
  this.color,
  this.opacity,
  this.colorBlendMode,
  this.fit,
  this.alignment = Alignment.center,
  this.repeat = ImageRepeat.noRepeat,
  this.centerSlice,
  this.matchTextDirection = false,
  this.gaplessPlayback = false,
  this.filterQuality = FilterQuality.low,
  this.isAntiAlias = false,
  Map<String, String>? headers,
  int? cacheWidth,
  int? cacheHeight,
}) : image = ResizeImage.resizeIfNeeded(cacheWidth, cacheHeight, NetworkImage(src, scale: scale, headers: headers)),
     assert(alignment != null),
     assert(repeat != null),
     assert(matchTextDirection != null),
     assert(cacheWidth == null || cacheWidth > 0),
     assert(cacheHeight == null || cacheHeight > 0),
     assert(isAntiAlias != null),
```

Varlık ve Resim Ekleme

- Flutter uygulamaları hem kodu hem de *varlıkları* (bazen kaynaklar olarak adlandırılır) içerebilir. Varlık, uygulamanızla birlikte paketlenip dağıtılan ve çalışma zamanında erişilebilen bir dosyadır. Yaygın varlık türleri arasında statik veriler (örneğin, JSON dosyaları), yapılandırma dosyaları, simgeler ve resimler (JPEG, WebP, GIF, hareketli WebP/GIF, PNG, BMP ve WBMP) bulunur.

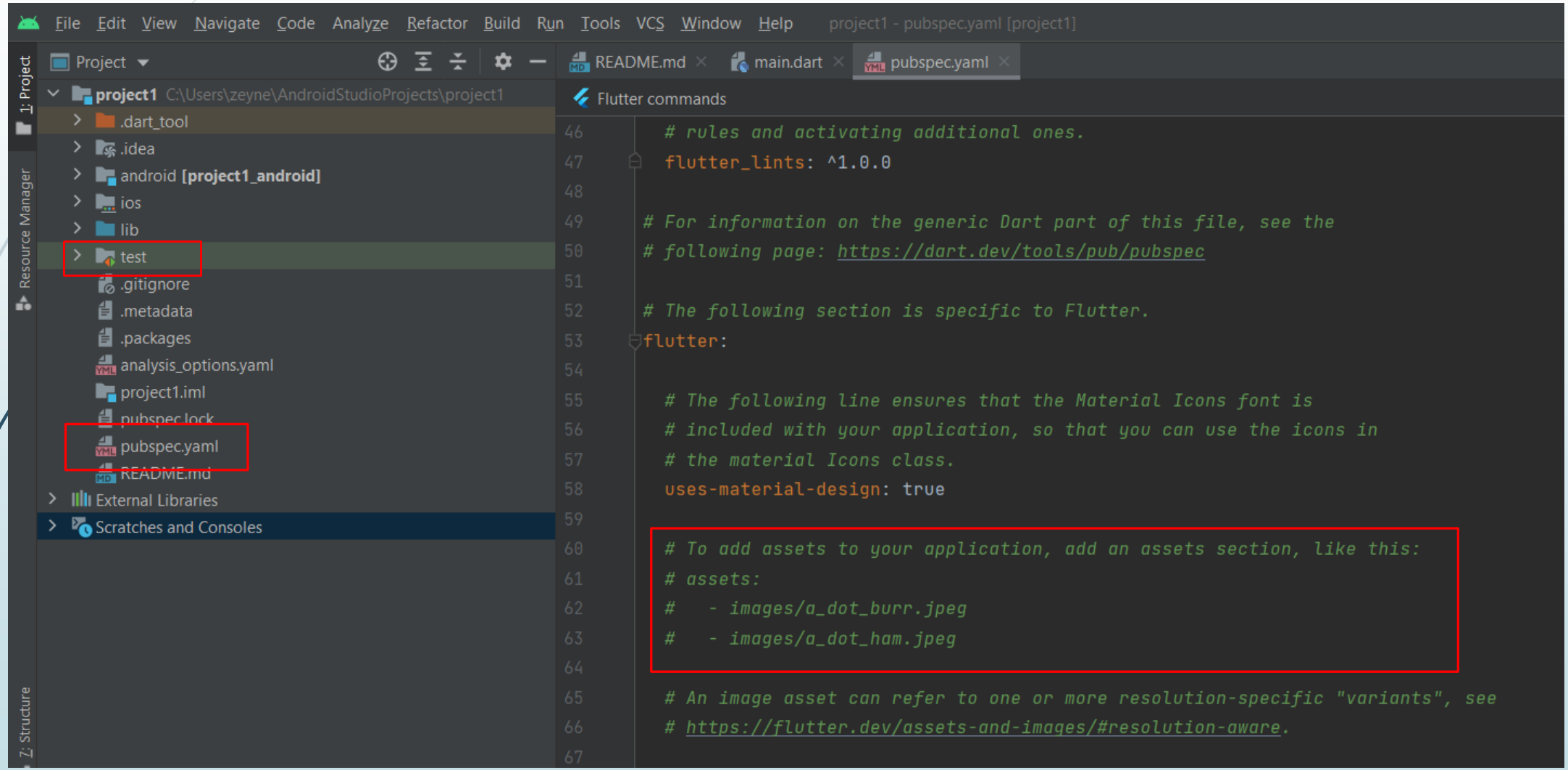
Varlıkları Belirtme:

```
flutter:
assets:
    - assets/my_icon.png
    - assets/background.png
```

NOT: Tüm varlıkları bir dizine dahil etmek için dizin adını '**/**' sonunda karakterle belirtin.

```
flutter:
assets:
    - directory/
    - directory/subdirectory/
```

Varlık ve Resim Ekleme



Varlık ve Resim Ekleme

Varlık Paketleme:

Flutter bölümünün varlıklar alt bölümü, uygulamaya dahil edilmesi gereken dosyaları belirtir. Her varlık, varlık dosyasının bulunduğu açık bir yolla (dosyaya göre) tanımlanır . Varlıkların beyan edildiği sıra önemli değildir.

NOT: Bir derleme sırasında Flutter, varlıkları , uygulamaların çalışma zamanında okuduğu varlık paketi adı verilen özel bir arşive yerleştirir.

Varlık Çeşitleri:

Oluşturma süreci, varlık değişkenleri kavramını destekler. Bir varlığın farklı bağlamlarda görüntülenebilecek farklı sürümleri vardır. **pubspec.yaml**'in varlıklar bölümünde bir varlığın yolu belirtildiğinde, oluşturma işlemi bitişik alt dizinlerde aynı ada sahip dosyaları arar. Bu tür dosyalar daha sonra belirtilen varlıkla birlikte varlık paketine dahil edilir.

Varlık ve Resim Ekleme

Varlık Çeşitleri:

Örneğin, uygulama dizininizde aşağıdaki dosyalar varsa:

```
.../pubspec.yaml
.../graphics/my_icon.png
.../graphics/background.png
.../graphics/dark/background.png
...etc.
```

Ve **pubspec.yaml** dosyanız şunları içerir:

```
flutter:
  assets:
    - graphics/background.png
```

Ardından her ikisi de **graphics/background.png** varlık **graphics/dark/background.png** paketinize dahil edilir. Birincisi ana varlık olarak kabul edilirken, ikincisi bir değişken olarak kabul edilir.

Öte yandan, grafik dizini belirtilirse:

```
flutter:
  assets:
    - graphics/
```

Ardından **graphics/my_icon.png**, **graphics/background.png** ve **graphics/dark/background.png** dosyaları da dahildir.

Flutter, çözünürlüğe uygun görüntüleri seçerken varlık değişkenlerini kullanır. Gelecekte, bu mekanizma, farklı yerel ayarlar veya bölgeler için varyantları, okuma talimatlarını vb. içerecek şekilde genişletilebilir.

Varlık ve Resim Ekleme

Varlıklar Yükleniyor:

Uygulamanız varlıklarına bir nesne aracılığıyla erişebilir. (AssetBundle)

Varlık paketindeki iki ana yöntem, mantıksal bir anahtar verildiğinde paketten bir dize/metin varlığı (**loadString()**) veya bir görüntü/ikili varlık ( ) yüklemenize izin verir. Mantıksal anahtar , derleme zamanında dosyada **load()** belirtilen varlığın yolu ile eşlenir.

```
name: my_awesome_application
flutter:
  assets:
   - images/hamilton.jpeg
   - images/lafayette.jpeg
```

Varlık ve Resim Ekleme

Resimler yükleniyor:

Bir görüntüyü yüklemek için, **AssetImage** bir parçacığın **build()** yöntemindeki sınıfı kullanın.

```
return const Image ( image : AssetImage (
'graphics/background.png' ));
```

Varsayılan varlık paketini kullanan her şey, görüntüleri yüklerken çözünürlük farkındalığını devralır. **ImageStream**(veya gibi bazı alt düzey sınıflarla çalışıyorsanız, **ImageCache** ölçekle ilgili parametreleri de fark edeceksiniz.)

Resmi yüklemek için şunu kullanın:

```
return const AssetImage ( 'icons/heart.png' , package: 'my_icons' );
```

Varlık ve Resim Ekleme

Resimler yükleniyor:

`.../lib/backgrounds/background1.png`

`.../lib/backgrounds/background2.png`

`.../lib/backgrounds/background3.png`

Diyelim ki ilk resmi dahil etmek için **pubspec.yaml**, uygulamanın bunu **assets** bölümde belirtmesi gerekir:

```
flutter:
  assets:
    -packages/fancy_backgrounds/backgrounds/background1.png
```

NOT: Bu **lib/** nedenle varlık yoluna dahil edilmemelidir.

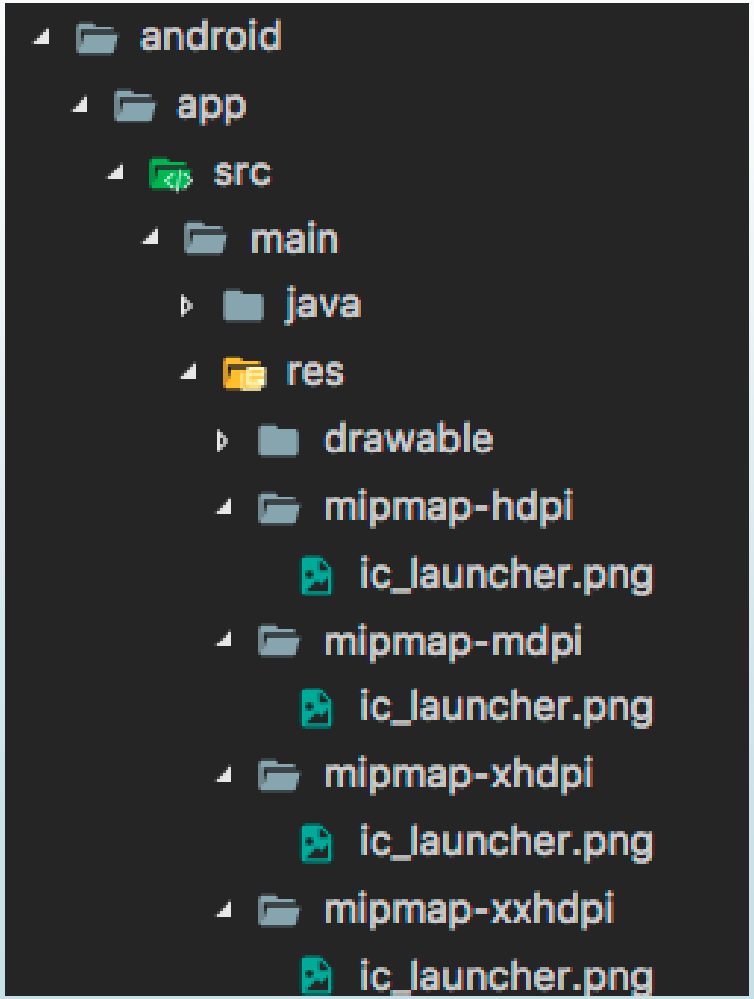
Bir paket geliştiriyorsanız, paket içinde bir varlık yüklemek için paketin **'pubspec.yaml'** dosyasında belirtin:

```
flutter:
  assets:
    - assets/images/
```

Uygulama simgesinin güncellenmesi

Android:

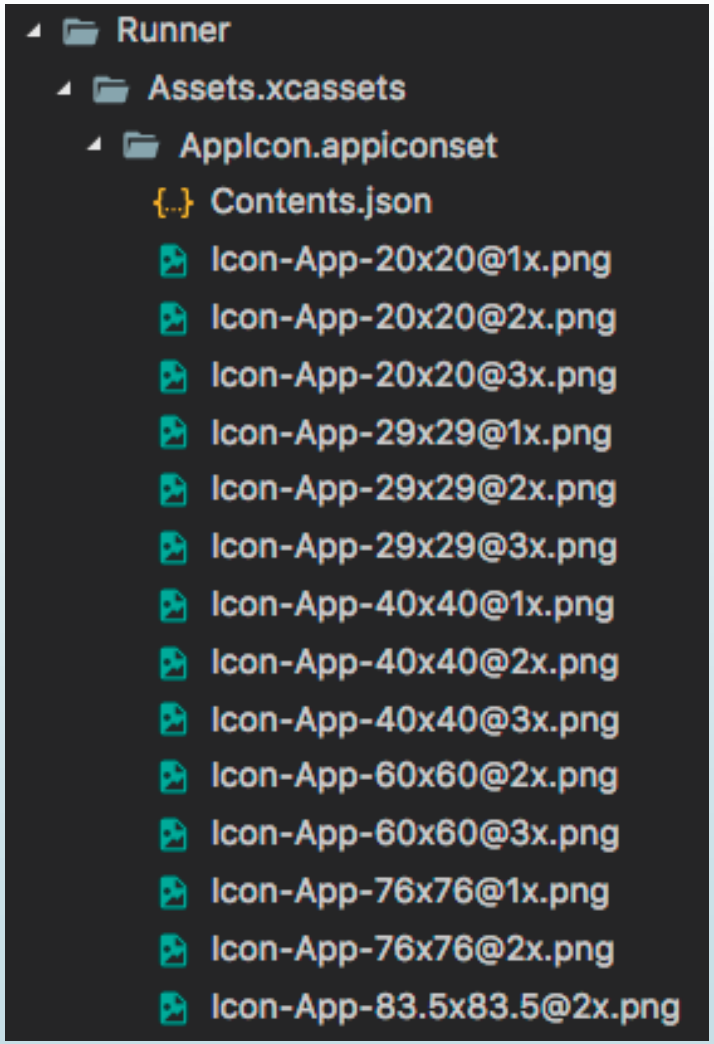
Flutter projenizin kök dizininde **.../android/app/src/main/res.** Gibi çeşitli **bitmap** kaynak klasörleri **mipmap-hdpi**, adlı yer tutucu görüntüleri içerir **ic_launcher.png.** Bunları, Android Geliştirici Kılavuzu tarafından belirtildiği gibi, ekran yoğunluğu başına önerilen simge boyutuna uygun olarak istediğiniz varlıklarla değiştirin .



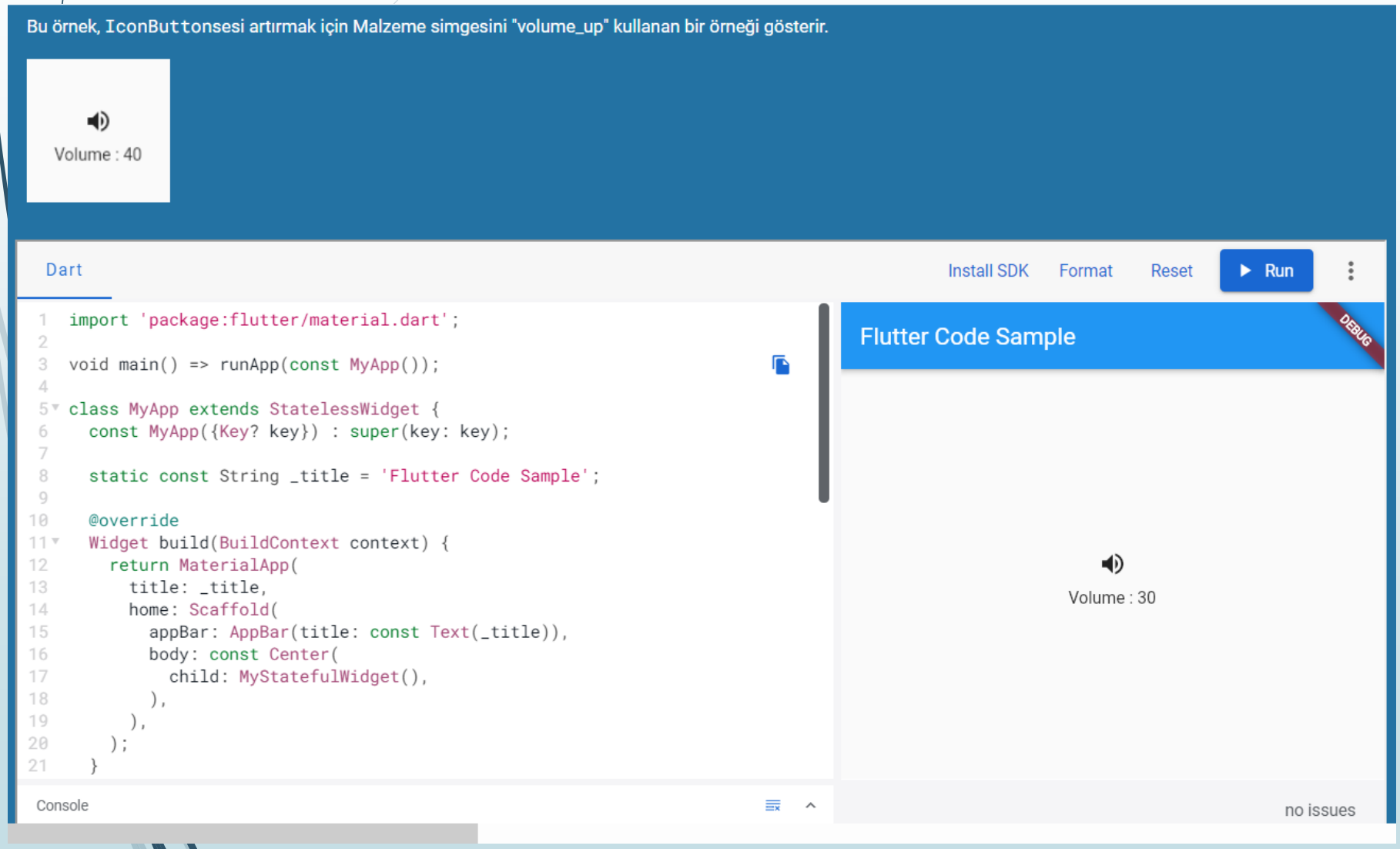
Uygulama simgesinin güncellenmesi

iOS:

Flutter projenizin kök dizininde **.../ios/Runner.Assets.xcassets/AppIcon.appiconset** Dizin zaten yer tutucu resimler içeriyor . Bunları, Apple İnsan Arayüzü Yönergeleri tarafından belirtildiği şekilde dosya adlarında belirtilen uygun boyuttaki görüntülerle değiştirin . Orijinal dosya adlarını saklayın.



Buton Widget'lar



IconButton:

- Bir malzeme tasarımı simgesi butonudur.
- Simge butonu, dokunmalara renkle doldurarak tepki veren bir Malzeme widget'ına yazdırılan bir resimdir.
- Simge butonları genellikle **AppBar.actions** alanında kullanılır, ancak başka birçok yerde de kullanılabilirler.
- **onPressed** geri arama null ise , düğme devre dışı bırakılır ve dokunmaya tepki vermez.
- IconButton

Buton Widget'lar

OutlinedButton:

- Bir Materyal Tasarımı "Özetlenen Buton"; esasen ana hatlarıyla belirlenmiş bir kenarlığa sahip bir TextButton'dur.
- OutlinedButton

İşte temel OutlinedButton örneği .

Dart | Install SDK | Format | Reset | Run

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

void main() => runApp(const MyApp());

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  static const String _title = 'Flutter Code Sample';

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: _title,
      home: Scaffold(
        appBar: AppBar(title: const Text(_title)),
        body: const Center(
          child: MyStatelessWidget(),
        ),
      ),
    );
  }
}
```

Flutter Code Sample

DEBUG

Click Me

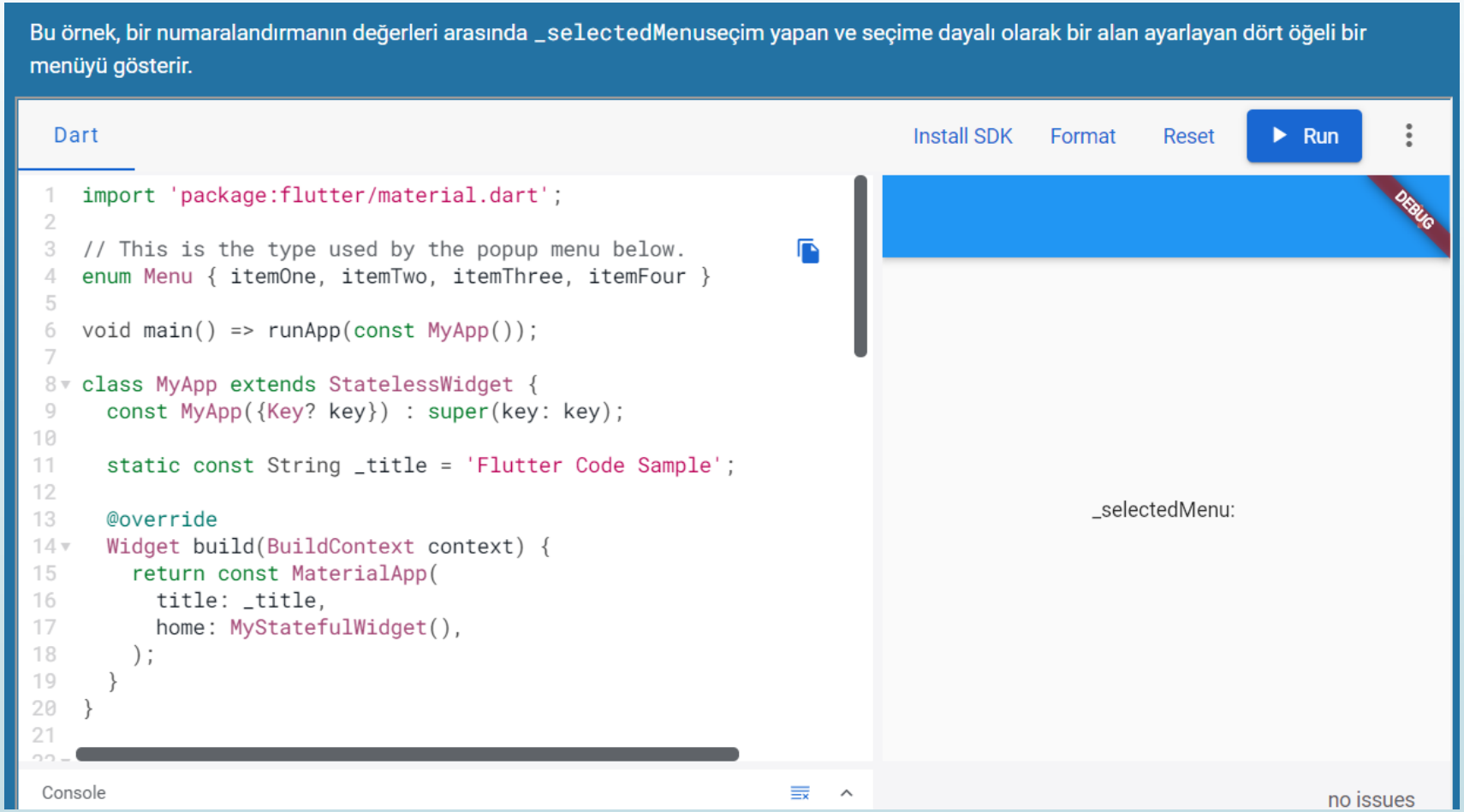
Console

no issues

Buton Widget'lar

PopupMenuButton:

- Basıldığında bir menü görüntüler ve bir öğe seçildiğinden menü kapatıldığında Selected'i çağırır. onSelected'e iletilen değer, seçilen menü öğesinin değeridir.
- PopupMenuButton



Buton Widget'lar

TextButton:

- Bir Materyal Tasarımı "Metin Butonu".
- TextButton

Bu örnek, devre dışı bırakılmış bir TextButton, etkinleştirilmiş bir TextButton ve son olarak degrade arka plana sahip bir TextButton'un nasıl oluşturulacağını gösterir.

Dart | Install SDK | Format | Reset | Run

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

void main() => runApp(const MyApp());

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  static const String _title = 'Flutter Code Sample';

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: _title,
      home: Scaffold(
        appBar: AppBar(title: const Text(_title)),
        body: const MyStatelessWidget(),
      ),
    );
  }
}
```

Flutter Code Sample

DEBUG

Disabled

Enabled

Gradient

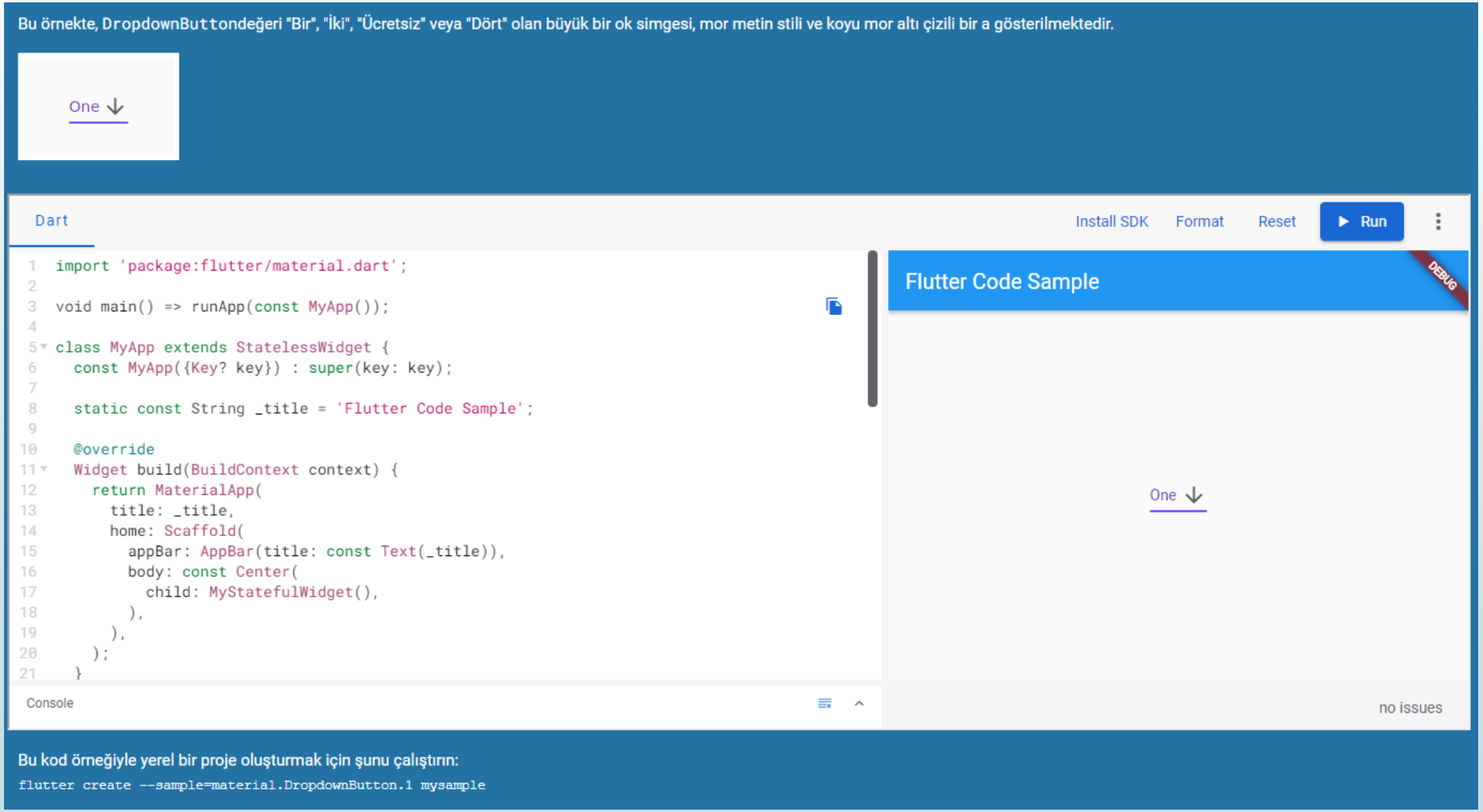
Console

no issues

Buton Widget'lar

DropdownButton:

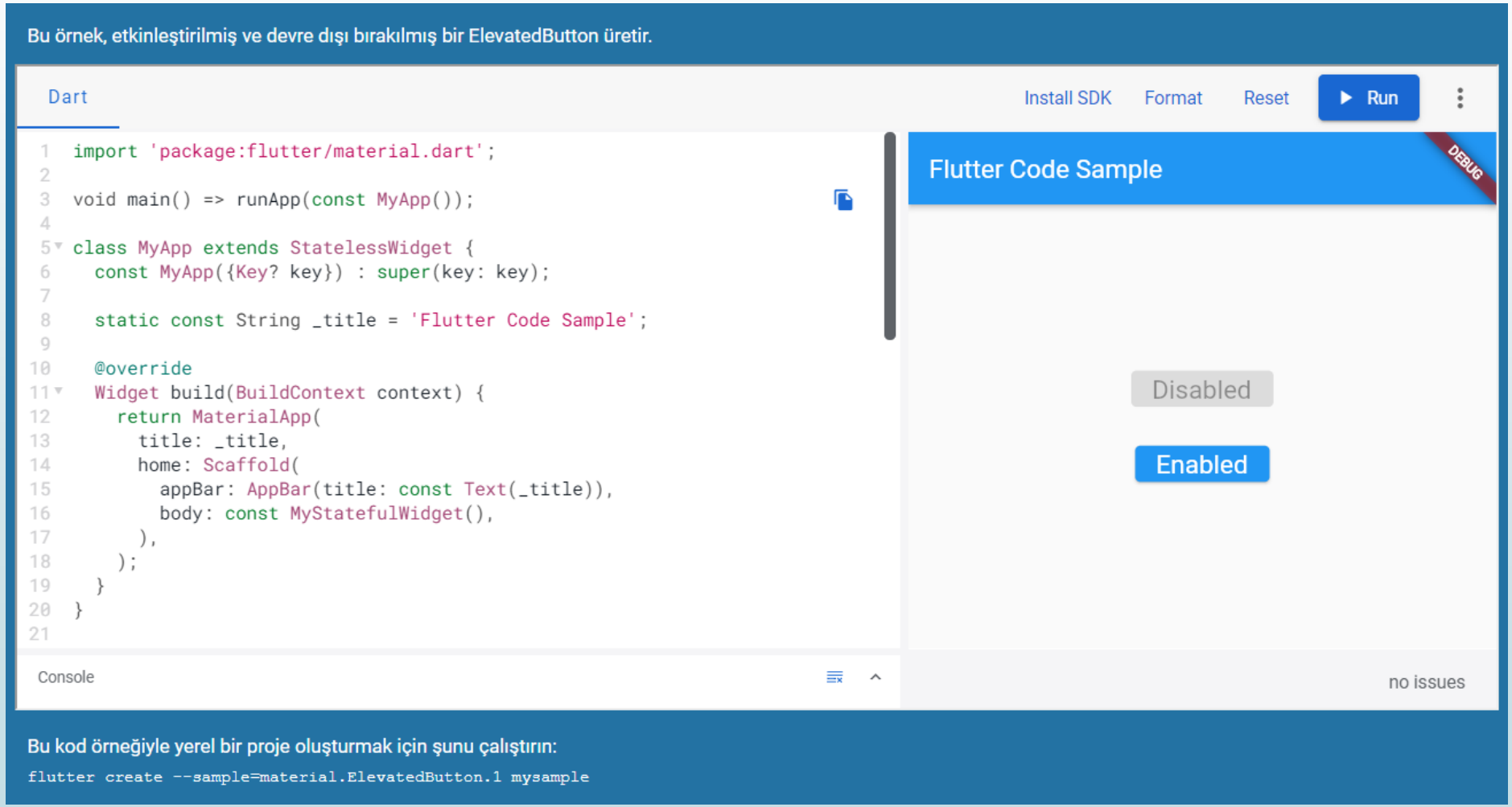
- Öğe listesinden seçim yapmak için bir malzeme tasarımı butonu.
- DropdownButton



Buton Widget'lar

ElevatedButton:

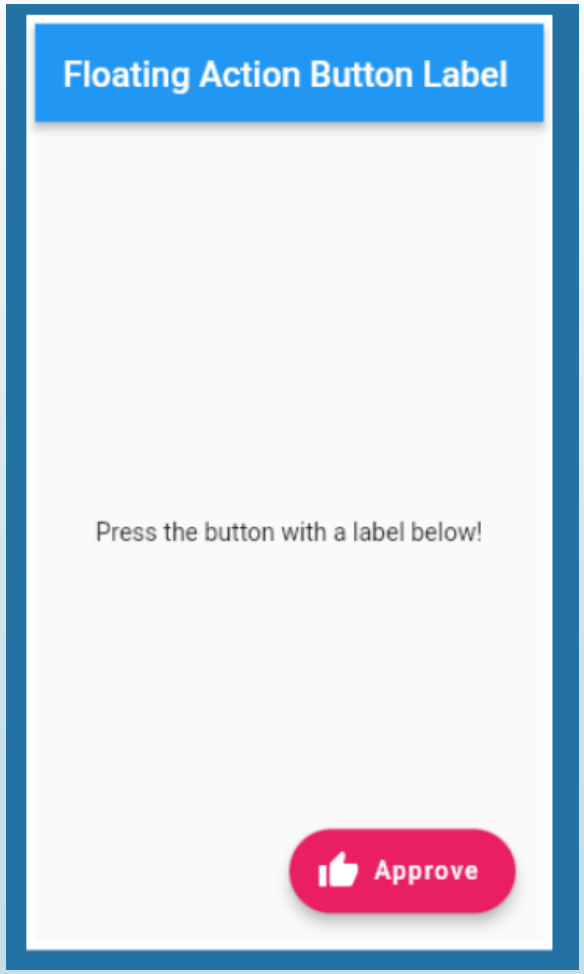
- Bir Materyal Tasarımı "yükseltilmiş buton".
- ElevatedButton

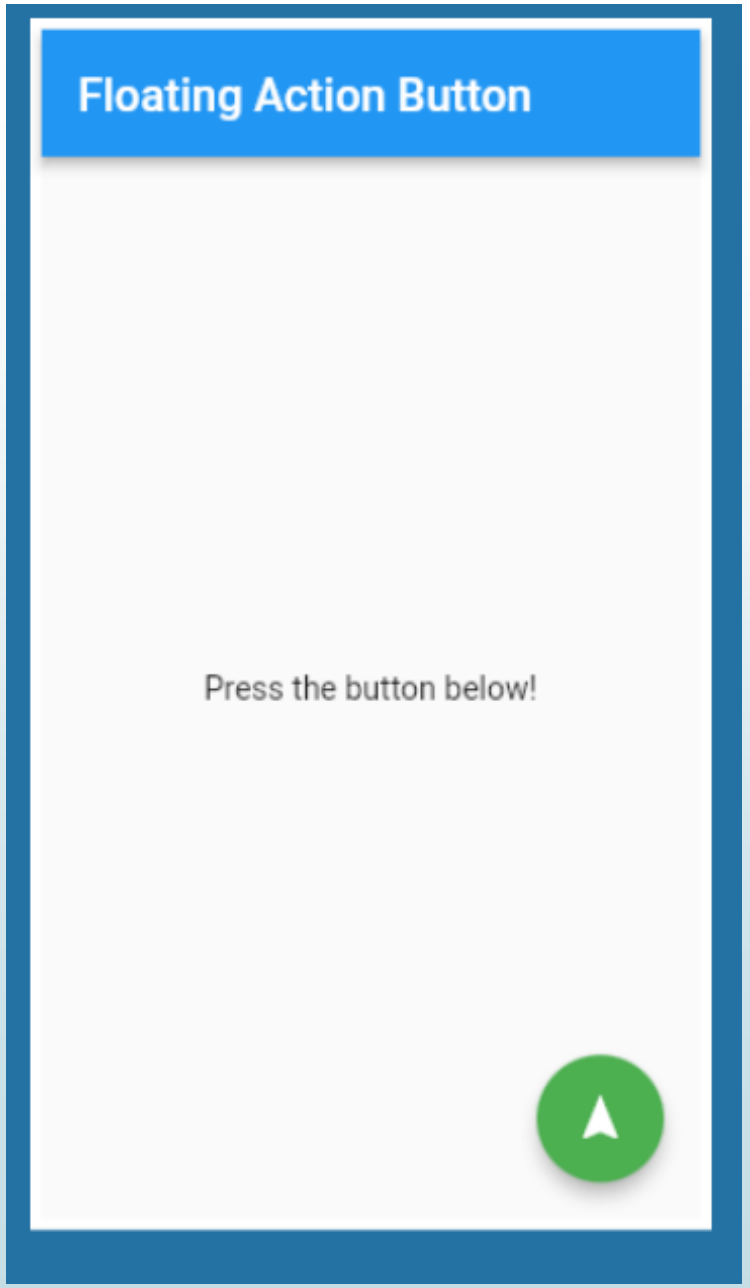


Buton Widget'lar

FloatingActionButton:

- Bir malzeme tasarımı kayan eylem butonu.
- FloatingActionButton





TextField

En sık kullanılan metin girişi widget'ıdır. Varsayılan olarak, TextField bir alt çizgi ile dekore edilmiştir. Özelliği **InputDecoration** olarak sağlayarak bir etiket, simge, satır içi ipucu metni ve hata metni ekleyebilirsiniz .

```
TextField(
  decoration: InputDecoration(
    border: OutlineInputBorder(),
    hintText: 'Enter a search term',
  ),
),
```

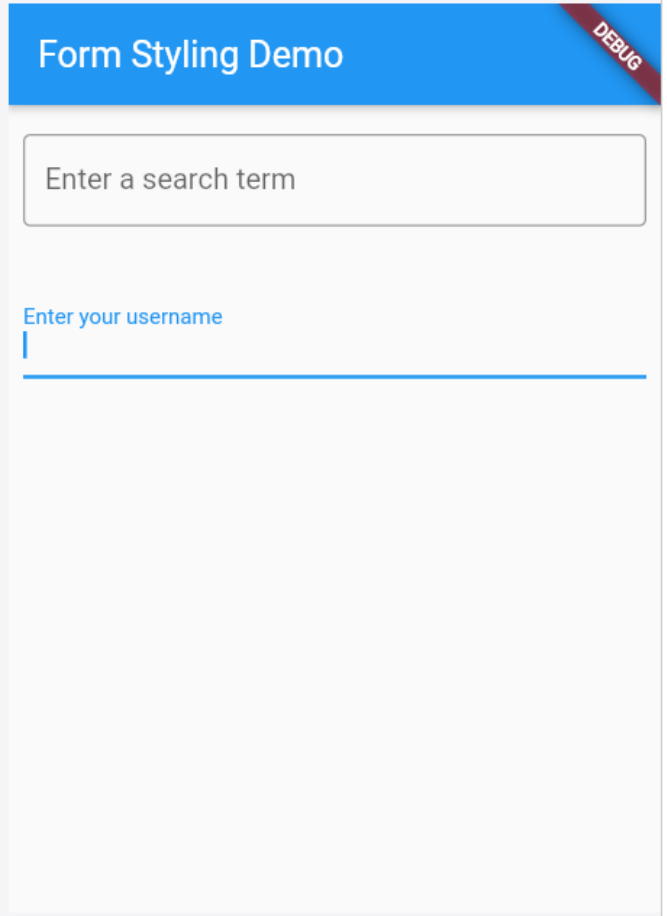
TextField

Örnek

Dart

Install SDK Format Reset Run

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

void main() => runApp(const MyApp());

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    const appTitle = 'Form Styling Demo';
    return MaterialApp(
      title: appTitle,
      home: Scaffold(
        appBar: AppBar(
          title: const Text(appTitle),
        ),
        body: const MyCustomForm(),
      ),
    );
  }
}

class MyCustomForm extends StatelessWidget {
  const MyCustomForm({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
```



Console

no issues

Yardımcı Kaynaklar

- Adım Adım Flutter İle Mobil Uygulamalar (Rakıcı Oğuz , 2021)
- https://flutter.dev/

İlginiz için teşekkürler…

Hazırlayan : **Zeynep İrem KESLER 1911404048**
E-posta : zeynepiremkesler@gmail.com
Tarih : 17/05/2022
Ders Yürütücüsü : Doç. Dr. İsmail KIRBAŞ
E-posta : ismkir@gmail.com