Yeni bir flutter örnek projesi açalım. MyApp sınıfı kalsın diğerlerini silelim. Bir tane Giris adında stateless widget oluşturup MyApp sınıfı içerisindeki MaterialApp widgetının home parametresine Giris sınıfını girelim.

```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) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        primarySwatch: Colors.blue,
      ),
      home: Giris(),
    );
  }
}

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.green,
    );
  }
}
```

backgorunColor parametresinde Theme widgetını kullanabiliriz.

ThemeData, programın sitili ile ilgili tüm bilgileri tutar. ThemeData bilgilerine widget ağacının herhangi bir yerinden ulaşmak için Theme sınıfının of metodunu kullanırız.

backgroundColor: Theme.*of*(context). dediğimiz zaman renk paletindeki tüm renk seçenekleri karşımıza gelir.

backgroundColor: Theme.*of*(context).primaryColor, // primaryColor ile primarySwatcha gönderilen ana renge ulaşılır.

primaryColor yerine **primaryColorDark** dersek ana rengin koyu tonunu **primaryColorLight** dersek açık tonunu getirir.

Eğer, MateriaApp widgetının theme: parametresindeki ThemeData widgetının primaryColor parametresini ayrıca belirleyebiliriz.

```dart
theme: ThemeData(
  primarySwatch: Colors.blue,
  primaryColor: Colors.red,
),
```

Şimdi Scaffolttaki parametreyi, **backgroundColor: Theme.*of*(context).primaryColor** yaparsak rengin kırmızıya döndüğünü görürüz.

Bu durumda primaryColor kırmızı renk olsada
backgroundColor: Theme.*of*(context).**primaryColorLight**, yaparsak
primarySwatch: Colors.*blue*, mavi olduğu için ekran açık mavi tonda oluşur.

Program temasında ana renk tonları dışında ikinci bir renkte belirleyebiliriz. İkinci rengimiz accentColor. accentColor artık theme.of ile kullanılmıyor.

Renkleri Colors. şeklinde belirlelemek kolay olsada Theme.of ile ThemeData renklerini kullanmak aynı anda bir çok noktadaki rengin değişimini sağlayacağı için kullanışlı olabilir.

Container() Widget:

İçinde herhangi bir eleman yoksa bulunduğu yerin tümünü kaplar.
Eğer içinde bir eleman var ise o elemanların boyutu kadar yer kaplar.

color: Colors.amber Renklendirir.
child: Text('Mobil') Yazı kadar yer kaplamış olur.
width: 150 Genişlik belirler.
height: 100 Yükseklik belirler.
alignment: Alignment. Dediğimiz anda hizalama seçenekleri çıkar.
center: ortalar vs

constraints: BoxConstraints(), Container kutusu boyutları için kısıtlama getirir.
minWidth: 300 Minimum genişliği belirler
minHeight: 300 Minimum genişliği belirler
maxWidth: 400 Maksimum genişliği belirler
maxHeight: 500 Maksimum genişliği belirler

margin: EdgeInsets.all(15), Containerın dört etrafında boşluk oluşturur.
EdgeInsets.only(left: 15, top: 50), sadece soldan, sağdan vs boşluk için
EdgeInsets.symmetric(vertical: 50, horizontal: 100) dikey ve yatay eksen için boşluk verilebilir.
EdgeInsets.fromLTRB(left, top, right, bottom), saol,sağ,üst, alt değerler direk yazılarak boşluk verilebilir.

padding: EdgeInsets.all(15), Edge seçenekleri burada da geçerlidir. Containerın içindeki nesnelerin container sınırlarına uzaklıklarını belirler.

decoration: BoxDecoration() Contanier çerçevesinin çeşitli özelliklerini belirler
color: Colors.orange, Kutunun rengini belirler.
color NOT: Hem decoration hemde contanier için aynı anda color verilmez hata verir. İki kısımdan birinde color verilmelidir.
shape: BoxShape.circle, Containerın şeklini belirler. Yuvarlak
BoxShape.rectangle, Kare, diktörtgen, yani köşeli.
border: Border.all(), Containerın 4 sınırına bir çizgi ekler.
width: 4 , Çizginin kalınlığı
color: Colors.purple, Çizginin rengi.

Sınır çizgilerini ayrı ayrı da verebiliriz.

Border(top: BorderSide(color: Colors.blue, width: 8)), Üst çizgi

borderRadius: BorderRadius.circular(20) 4 Köşenin şeklini belirler. Radius (yarıçap değeri) olarak 20 verebiliriz.

```
BorderRadius.vertical(top: Radius.circular(20)) Üst/alt çizgi köşeleri
BorderRadius.horizontal(left: Radius.circular(20)) Sol/Sağ çizgi köşeleri
BorderRadius.only(topLeft: Radius.circular(20)) Her köşe ayrı ayrı

boxShadow: []  Gölgelnedirme yapar. List içinde BoxShadowlar bekler.
        BoxShadow(
          color: Colors.green,
          offset: Offset(40, 20),   x ve y ekseninde kaydırır
          blurRadius: 20    flulaştırır.
        ),

        BoxShadow(
           color: Colors.yellow,
           offset: Offset(10, -20),
           blurRadius: 20
        ),
```

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

void main() {
  runApp(const MyApp());
}

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

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(

        primarySwatch: Colors.blue,
      ),
      home: Giris(),
    );
  }
}

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

  @override
  Widget build(BuildContext context) {
    return Theme(data: ThemeData(accentColor: Colors.yellow),
      child: Scaffold(
        appBar: AppBar(),
        //backgroundColor: Theme.of(context).primaryColor,
        body: Container(
          //color: Colors.amber,
          child: Text("MOBİL"),
          width: 150,
          //width: double.infinity,
          height: 100,
          //alignment: Alignment.center,
          constraints: BoxConstraints(
            minWidth: 300,
            minHeight: 300,
            maxWidth: 400,
```

```
            maxHeight: 500,
          ),
          margin: EdgeInsets.all(20),
          //margin: EdgeInsets.only(left: 15, top: 50),
          //margin: EdgeInsets.symmetric(vertical: 50, horizontal: 100),
          //margin: EdgeInsets.fromLTRB(left, top, right, bottom),
          padding: EdgeInsets.all(15),
          decoration: BoxDecoration(
            color: Colors.orange,
            //shape: BoxShape.circle,
            shape: BoxShape.rectangle,
            border: Border.all(
              width: 4,
              color: Colors.purple,
            ),
            //border: Border(top: BorderSide(color: Colors.blue, width:
8)),
            borderRadius: BorderRadius.circular(20),
            //borderRadius: BorderRadius.vertical(top:
Radius.circular(20)),
            //borderRadius: BorderRadius.horizontal(left:
Radius.circular(20)),
            //borderRadius: BorderRadius.only(topLeft:
Radius.circular(20)),
            boxShadow: [
              BoxShadow(
                  color: Colors.green,
                  offset: Offset(40, 20),
                  blurRadius: 20,
                ),
              BoxShadow(
                  color: Colors.yellow,
                  offset: Offset(10, -20),
                  blurRadius: 20
                ),
            ]
          ),
        ),
      ),
    );
  }
}
```

Column() Widget:

mainAxisAlignment: MainAxisAlignment.**start** Ana (Dik) eksenin hizalanması.

start: sütun başına,
center: ortasına,
end: sonuna,
spaceAround: Sütundaki nesneler etrafında rastgele boşluklar bırakarak, en üst ve en altta boşluk olur,
spaceBetween: Sütundaki nesneler arasında olabildiğince boşluklar bırakarak, en üstte ve en altta boşluk olmaz,

spaceEvenly: Sütundaki nesnelerin, sütunun alt üst sınırlarına ve kendi aralarında eşit boşluklar bırakarak,

crossAxisAlignment: CrossAxisAlignment.**start** Diğer (Yatay) eksenin hizalanması.

start: yatayda en sola,
center: yatayda ortaya,
end: yatayda en sağa,
stretch: Sütunu yatay düzlemde uzatır, sütun yatay olarak tüm ekranı kaplar.

mainAxisSize: MainAxisSize.**max** Sütunun dikey uzunluğunu belirler.

max: Sütunu maksimum belirler,
min: Sütunun boyunu içindeki nesneler kadar belirler,

```
Column(
  children: [
    Container(color: Colors.purple, child: Text("MOBİL"),),
    Container(color: Colors.blue, child: Text("MOBİL"),),
    Container(color: Colors.yellow, child: Text("MOBİL"),),
    //Container(width: double.infinity,)
  ],
  //mainAxisAlignment: MainAxisAlignment.start,
  //mainAxisAlignment: MainAxisAlignment.center,
  //mainAxisAlignment: MainAxisAlignment.end,
  //mainAxisAlignment: MainAxisAlignment.spaceAround,
  //mainAxisAlignment: MainAxisAlignment.spaceBetween,
  //mainAxisAlignment: MainAxisAlignment.spaceEvenly,
  //crossAxisAlignment: CrossAxisAlignment.start,
  //crossAxisAlignment: CrossAxisAlignment.center,
  //crossAxisAlignment: CrossAxisAlignment.end,
  //crossAxisAlignment: CrossAxisAlignment.stretch,
  //mainAxisSize: MainAxisSize.min,
  //mainAxisSize: MainAxisSize.max,

),
```

Center Widget(): İçinde bulunduğu nesnenin yeri kadar alan kaplar. İçindeki elemanların ekranın ortasına yerleşmesini sağlar.

heightFactor: 1..5, girilen değer kadar içindeki elemanın yüksekliği kadar yer kaplar. İçindeki nesnede o kaplanan yerin ortasına yerleşir.

widthFactor: 1..5, girilen değer kadar içindeki elemanın genişliği kadar yer kaplar. İçindeki nesnede o kaplanan yerin ortasına yerleşir.

factor NOT: Factorlar kullanılırsa, center widget ekran ortasında değil bulunduğu nesnenin en sol yada en üstten itibaren yerleşir.

```
Center(
  child: Container(
    width: 100,
```

```
    height: 100,
  ),
  //heightFactor: 1,
  //widthFactor: 1,
)
```

BAZI BUTON VE ÖZELLİKLERİ

TextButton() Widget: Text buton ekler.

```
TextButton(onPressed: (){}, child: Text('Outlined')),

TextButton.icon(onPressed: (){}, icon: Icon(Icons.add), label: Text('Text Buton
Icon')),

ElevatedButton (onPressed: (){}, child: Text('Outlined')),

ElevatedButton.icon(onPressed: (){}, icon: Icon(Icons.add), label: Text('Elevated
Icon')),

OutlinedButton(onPressed: (){}, child: Text('Outlined')),

OutlinedButton.icon(onPressed: (){}, icon:Icon(Icons.add), label: Text('Outlined
Icon')),
```

```
        onPressed: (){},   Butona basıldığında çalışacak fonksiyon
        child: Text('TextButon')  Çocuğu için bir Widget

        style: ButtonStyle(
                backgroundColor: Ana renk
               foregroundColor: Yazı rengi
               overlayColor: Tıklandığındaki efekt rengi
```

backgroundColor: MaterialStateProperty.*all*(Colors.*amberAccent*), renk vermek için Material durum özellik fonksiyonunu kullanabiliriz. Her biri için bunu yapabiliriz.

```
        foregroundColor: MaterialStateProperty.all(Colors.red),

        overlayColor: MaterialStateProperty.all(Colors.green),

        overlayColor: MaterialStateProperty.all(Colors.green.withOpacity(0.2)),
```

Butonun durumlarına göre (basılı, üzerine gelinmiş vs) ayrı ayrı durumlar için özellik verebiliriz.

```
        backgroundColor: MaterialStateProperty.resolveWith((states) {
                  if (states.contains(MaterialState.pressed)) {
                    return Colors.purple;
                  }
                  return null;
                }),
```

Buton sitillerini değiştirmek için **ButtonStyle** yerine buton ismiyle yazılan stil fonksiyonunda kullanabiliriz.

```
style: ElevatedButton.styleFrom(
        primary: Colors.amber,    ana renk
        onPrimary: Colors.blue,   yazı ve efekt rengi
        side: BorderSide(color: Colors.purple, width: 2), çerçeve çizgi rengini ve kalınlığını değiştirir.

        shape: StadiumBorder(),   çerçeve şeklini değiştirir.

        VEYA
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(10)
        ),
        Kullanabiliriz.
```

Bu buton stillerini MateriAPP kısmında ThemeData bölümünde yaparak bütün butonların aynı stili kullanmasını sağlayabiliriz.

```
elevatedButtonTheme: ElevatedButtonThemeData(style: ElevatedButton.styleFrom()),

elevatedButtonTheme: ElevatedButtonThemeData(style: ButtonStyle()),
```

FlutterLogo() Widget: Default olarak Flutter logoyu verir.

```
size: 45,   Logonun boyutu
style: FlutterLogoStyle.horizontal , Yazıyı logonun yanına getirir.
       FlutterLogoStyle.stacked,     Yazıyı loğunun altına getirir
textColor: Colors.amber,  Yazının rengini belirler.
```

```
Column(
  children: [
    TextButton(onPressed: (){}, child: Text('Outlined')),

    TextButton.icon(onPressed: (){}, icon: Icon(Icons.add), label: Text('Text Buton Icon')),

    ElevatedButton (
      onPressed: (){},
      child: Text('Elevated'),
      onLongPress: (){print("Uzun tıklandı");},

    /*  style: ButtonStyle(
        //backgroundColor: MaterialStateProperty.all(Colors.green),
        //foregroundColor: MaterialStateProperty.all(Colors.amber),
        //overlayColor: MaterialStateProperty.all(Colors.purple),
        //overlayColor: MaterialStateProperty.all(Colors.purple.withOpacity(0.5)),
        //overlayColor: MaterialStateProperty.all(Colors.purple.shade100),


      /*  backgroundColor: MaterialStateProperty.resolveWith((states)
```

```
        {if (states.contains(MaterialState.pressed)) {return
Colors.yellow;}
        return null;}
      ),*/


     ),*/

    /*  style: ElevatedButton.styleFrom(
        primary: Colors.purple,
        onPrimary: Colors.yellow,
        side: BorderSide(color:Colors.yellow, width: 2),
        shape: StadiumBorder(),
        //shape: RoundedRectangleBorder(borderRadius:
BorderRadius.circular(10))
      ),*/
    ),

    ElevatedButton.icon(onPressed: (){}, icon: Icon(Icons.add), label:
Text('Elevated Icon')),

    OutlinedButton(onPressed: (){}, child: Text('Outlined')),

    OutlinedButton.icon(onPressed: (){}, icon:Icon(Icons.add), label:
Text('Outlined Icon')),

    FlutterLogo(
      size: 150,
      style: FlutterLogoStyle.horizontal,
      //style: FlutterLogoStyle.stacked,
      textColor: Colors.blue,
    ),

  ],
)
```