

Flutter ile Uygulama Geliştirme Kursu | Android & IOS

Material Design

Kasım ADALAN
Elektronik ve Haberleşme Mühendisi
Android - IOS Developer and Trainer

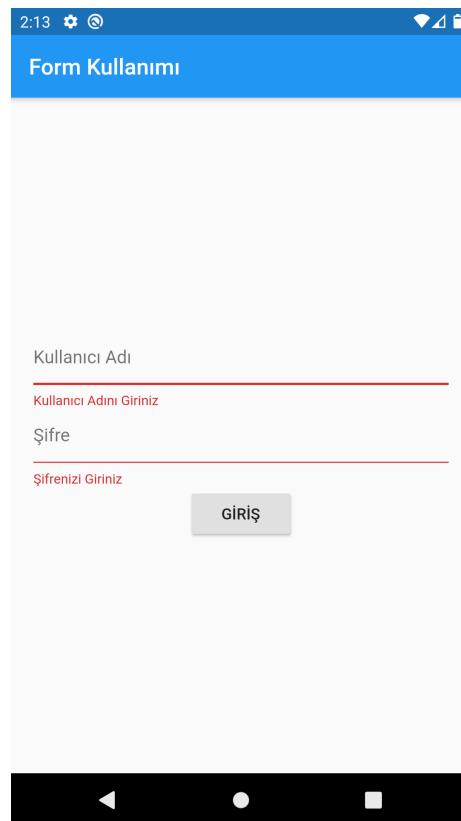
Eğitim İçeriği

- Forms
- AppBar
- Card
- ListView & GridView
- FutureBuilder ile Listeleme
- Tabs
- BottomNavigationBar
- Drawer

Forms

Form

- Flutter yapısında yer alan bu yapı sayesinde girdi kontrolleri yapabiliriz.



```

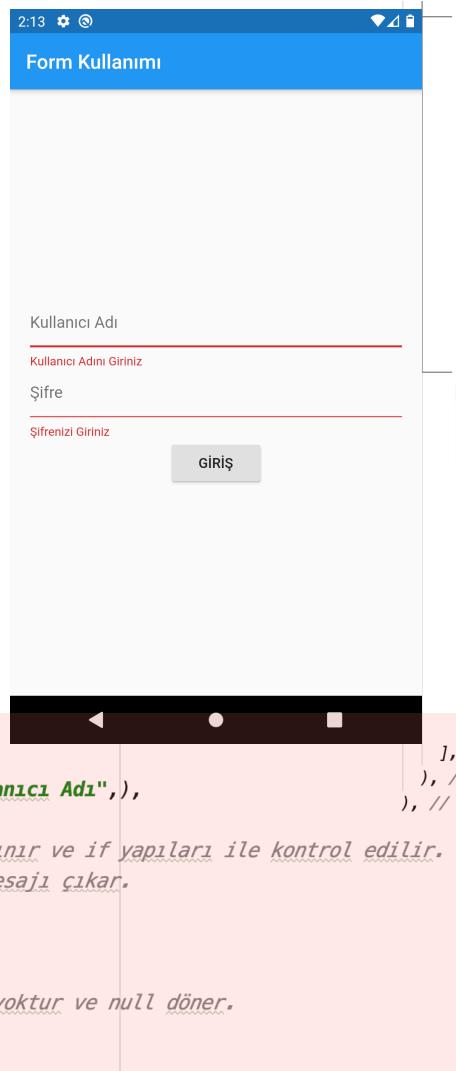
class _MyHomePageState extends State<MyHomePage> {

  var formKey = GlobalKey<FormState>(); //Form için gerekli key
  var tfKullaniciAdi = TextEditingController();
  var tfSifre = TextEditingController();

  @override
  Widget build(BuildContext context) {

    body: Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          Padding(
            padding: const EdgeInsets.all(20.0),
            child: Form(//Form yapısı
              key: formKey, //Key
              child: Column(
                children: <Widget>[
                  TextFormField(//Form için textfield
                    controller: tfKullaniciAdi,
                    decoration: InputDecoration(hintText: "Kullanıcı Adı"),
                    validator: (tfGirdisi) { //Girdi kontrolü
                      //tfGirdisi validate() metodu çalıştığında alınır ve if yapıları ile kontrol edilir.
                      //if yapıları çalışırsa return olarak uyarı mesajı çıkar.
                      if(tfGirdisi!.isEmpty){
                        return "Kullanıcı adı giriniz";
                      }
                      return null; //if yapıları çalışmadıysa sorun yoktur ve null döner.
                    },
                  ), // TextFormField
                ],
              ),
            ),
          ),
        ],
      ),
    );
  }
}

```



```

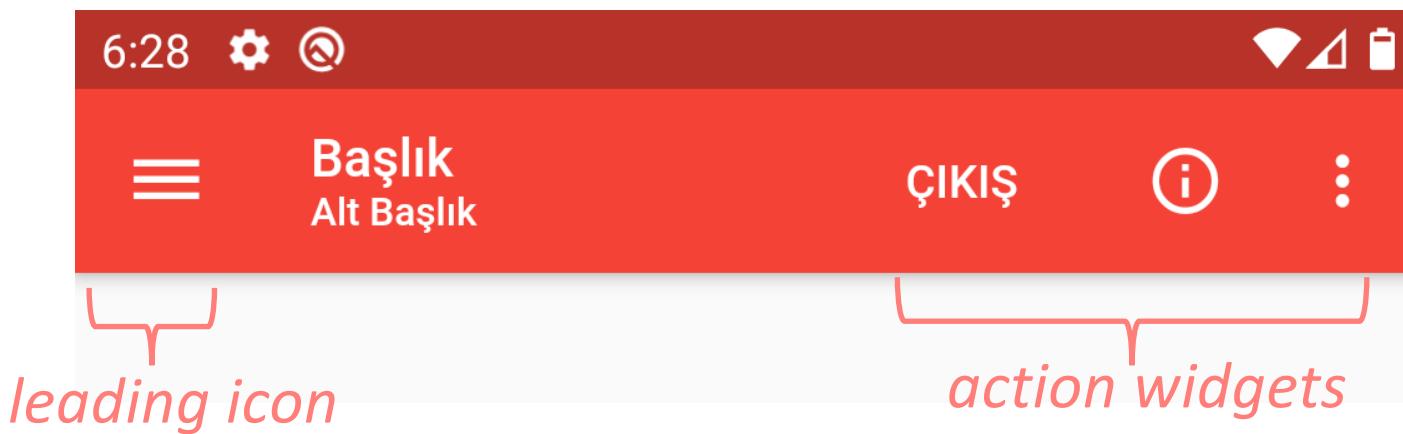
TextFormField(
  controller: tfSifre,
  obscureText: true,
  decoration: InputDecoration(hintText: "Şifre"),
  validator: (tfGirdisi) {
    if(tfGirdisi!.isEmpty){
      return "Şifre giriniz";
    }
    if(tfGirdisi!.length < 6){
      return "Şifreniz en az 6 haneli olmalıdır";
    }
    return null;
  },
), // TextFormField
ElevatedButton(
  child: Text("GİRİŞ"),
  onPressed: () {
    bool kontrolSonucu = formKey.currentState!.validate();
    //validate() metodу çalıştığında tüm textfield validator metodları çalışır.
    //validator metodlarının hepsiin sonuçları null ise
    //validate() metoduna true cevabı gelir ve yapmak istenen işlem yapılır.
    //validator metodlarından herhangi biri null değilse false cevabı gelir.
    if (kontrolSonucu) { //validate() cevabı true ise işlem yapılır.
      String ka = tfKullaniciAdi.text;
      String s = tfSifre.text;
      print("Kullanıcı Adı : $ka - Şifre : $s");
    }
  },
), // ElevatedButton
), // <Widget>[]
), // Column
), // Form

```

AppBar

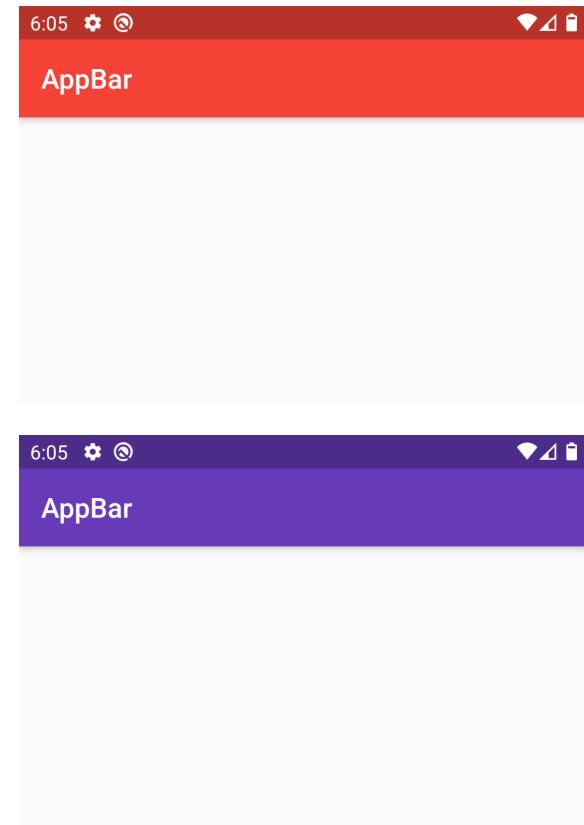
AppBar

- AppBar, sayfanın en üstünde bulunan bir widgettir.
 - Varsayılan olarak her sayfada gelmektedir.

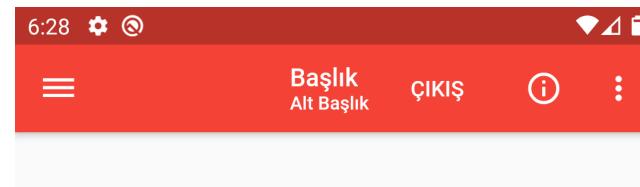
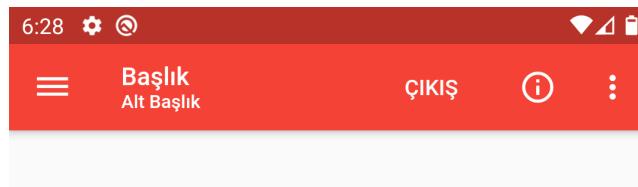


AppBar Renk Değişimi

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      title: 'Flutter Demo',  
      debugShowCheckedModeBanner: false,  
      theme: ThemeData(  
        primarySwatch: Colors.red, //AppBar renk değişimi  
        visualDensity: VisualDensity.adaptivePlatformDensity,  
      ), // ThemeData  
      home: MyHomePage(title: 'AppBar'),  
    ); // MaterialApp  
  }  
}
```



AppBar Özelleştirme



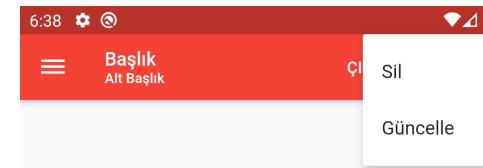
```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text("Başlık", style: TextStyle(color: Colors.white, fontSize: 16.0),),
          Text("Alt Başlık", style: TextStyle(color: Colors.white, fontSize: 12.0),),
        ],
      ), // Column
      centerTitle: false, // Başlık ortada görünür
      leading: IconButton(
        tooltip: 'Menu Icon', // Uzun basılınlca çıkan yazı
        icon: Icon(Icons.dehaze), // Görünecek icon
        onPressed: () {
          //Icona tıklanılma işlemi
          print("Menu Icon Tıklandı");
        },
      ), // IconButton
    ),
  );
}
```

```
actions: <Widget>[] // Sağ tarafta görenecek widgetlar
  TextButton(
    child: Text("ÇIKIŞ", style: TextStyle(color: Colors.white),),
    onPressed: () {
      print("Çıkış Tıklandı");
    },
  ), // TextButton
  IconButton(
    icon: Icon(Icons.info_outline),
    tooltip: 'Bilgi',
    onPressed: () {
      print("Bilgi Tıklandı");
    },
  ), // IconButton
  IconButton(
    icon: Icon(Icons.more_vert),
    tooltip: 'Popup Menu',
    onPressed: () {
      print("Popup Menu Tıklandı");
    },
  ), // IconButton
], // <Widget>[]
), // AppBar
```

AppBar Popup Menu Ekleme

- Actions içine PopupMenuItem ekleyebiliriz.

```
 IconButton(  
   icon: Icon(Icons.info_outline),  
   tooltip: 'Bilgi',  
   onPressed: () {  
     print("Bilgi Tıkllandı");  
   },  
, // IconButton  
 PopupMenuItem(  
   child: Icon(Icons.more_vert),  
   itemBuilder: (context) => [  
     PopupMenuItem(  
       value: 1,  
       child: Text("Sil"),  
     ), // PopupMenuItem  
     PopupMenuItem(  
       value: 2,  
       child: Text("Güncelle"),  
     ), // PopupMenuItem  
   ],  
   onSelected: (menuItemValue){  
     if(menuItemValue == 1){  
       print("Sil Tıkllandı");  
     }  
  
     if(menuItemValue == 2){  
       print("Güncelle Tıkllandı");  
     }  
   },  
, // PopupMenuItem  
, // <Widget>[]  
, // AppBar
```

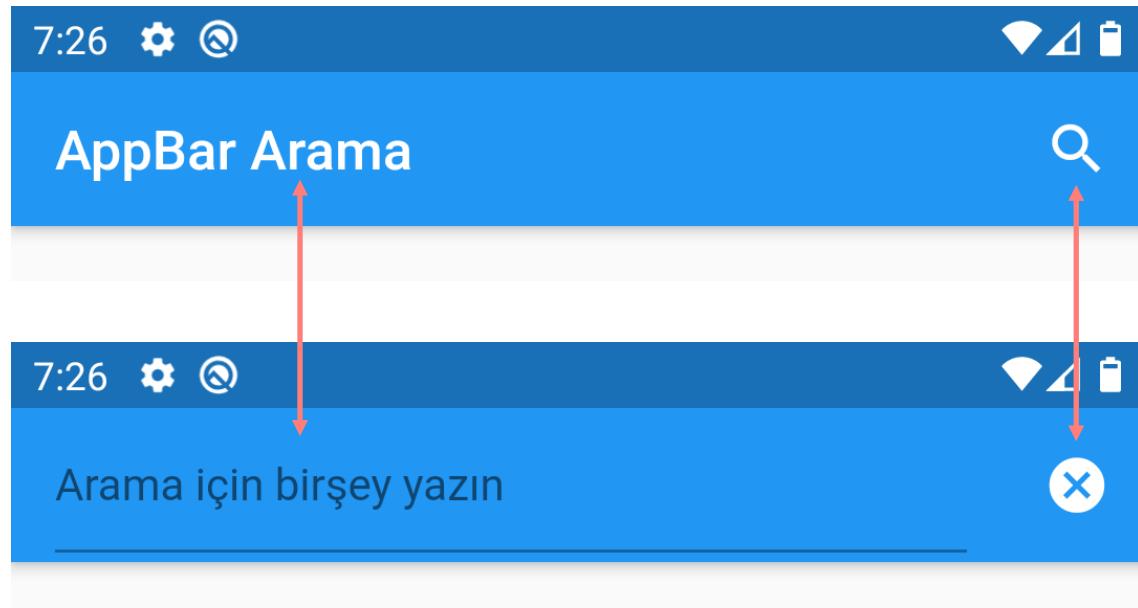


AppBar Arama Özelliği

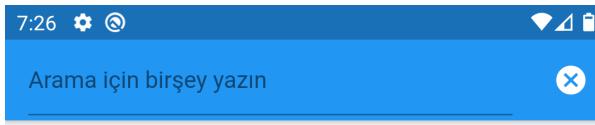
- Toolbar üzerinde arama işlemi yapmak için textfield kullanılabilir.
- İki tasarım oluşturulur arama yapıldığı durumda için ve arama yapılmadığı durum için.
 - Arama yapıldığını state özelliği olan bir değişken ile takip edebiliriz.

```
setState(() {  
    //Arama iconuna tıklanınca arama durumu true olur  
    aramaYapiliyorMu = true;  
});
```

```
setState(() {  
    //İptal iconuna tıklanınca arama durumu false olur  
    aramaYapiliyorMu = false;  
});
```



AppBar Arama Özelliği



```
class _MyHomePageState extends State<MyHomePage> {  
  
  bool aramaYapiliyorMu = false;  
  
  @override  
  Widget build(BuildContext context) {  
    return Scaffold(  
      appBar: AppBar(  
        //Başlık koşulu aramaYapiliyorMu : true ise textField  
        //false ise Başlık görünecek.  
        title: aramaYapiliyorMu ? TextField(  
          decoration: InputDecoration(hintText: "Arama için birşey yazın"),  
          onChanged: (aramaSonucu) {  
            //textfield onChanged metodu ile her harf girildiğinde  
            //veya silindiğinde çalışır.  
            print("Arama Sonucu : $aramaSonucu");  
          },  
        ) // TextField  
        : Text('AppBar Arama'),  
        //Action button koşulu aramaYapiliyorMu : true ise iptal iconu  
        //false ise arama iconu görünecek.  
      ),  
    );  
  }  
}
```

```
actions: <Widget>[  
  aramaYapiliyorMu  
  ? IconButton(icon: Icon(Icons.cancel),  
  onPressed: () {  
    setState(() {  
      //İptal iconuna tıklanınca arama durumu false olur  
      aramaYapiliyorMu = false;  
    });  
  }, // IconButton  
  : IconButton(icon: Icon(Icons.search),  
  onPressed: () {  
    setState(() {  
      //Arama iconuna tıklanınca arama durumu true olur  
      aramaYapiliyorMu = true;  
    });  
  }, // IconButton  
], // <Widget>[]  
>, // AppBar
```

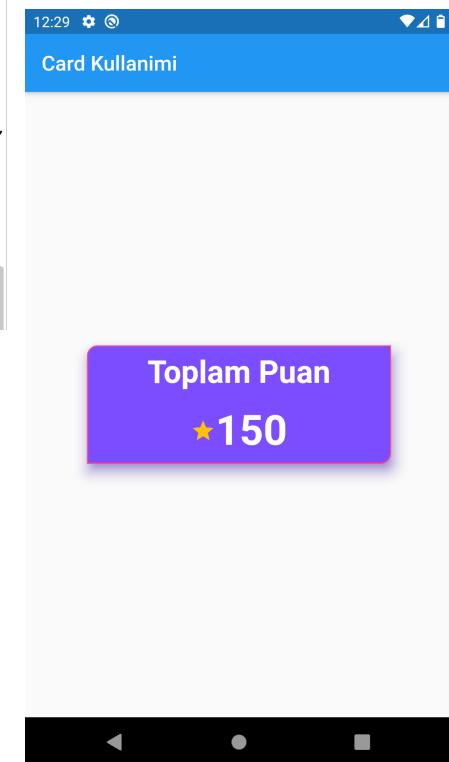
Card

Card

- Tasarım üzerinde kullanabileceğimiz kart etkisi yaratan widgettir.
 - Listeleme ve sayfa üzerinde sabit tasarımlarda kullanılabilir.

```
body: Center(  
  child: Column(  
    mainAxisAlignment: MainAxisAlignment.center,  
    children: <Widget>[  
      SizedBox(  
        width: 300,  
        child: Card(  
          color: Colors.deepPurpleAccent, //Arkaplan  
          elevation: 10.0, //Gölge miktarı  
          shadowColor: Colors.deepPurpleAccent, //Gölge rengi  
          shape: RoundedRectangleBorder(  
            borderRadius: BorderRadius.only(//İstenilen köseyi yuvarlama  
              bottomRight: Radius.circular(10),  
              topLeft: Radius.circular(10)), // BorderRadius.only  
              side: BorderSide(width: 1, color: Colors.pinkAccent) //Çerçeve eklemeye  
            ), // RoundedRectangleBorder  
          child: Center(  
            child: Column(  
              children: <Widget>[  
                Padding(  
                  padding: const EdgeInsets.all(8.0),  
                  child: Text("Toplam Puan",  
                    style: TextStyle(color: Colors.white, fontSize: 30.0, fontWeight: FontWeight.bold),  
                  ), // Text  
                ), // Padding  
              ], // Column  
            ), // Center  
          ), // Card  
        ), // SizedBox[]  
      ], // Column  
    ], // Column  
  ), // MainAxisAlignment  
); // Center
```

```
  padding: const EdgeInsets.all(8.0),  
  child: Row(  
    mainAxisAlignment: MainAxisAlignment.center,  
    children: <Widget>[  
      Icon(Icons.star,color: Colors.amber),  
      Text("150",  
        style: TextStyle(color: Colors.white,fontSize: 40.0,fontWeight: FontWeight.bold),  
      ), // Text  
    ], // Row  
  ), // Padding  
), // Column  
), // Center  
, // Card  
, // SizedBox[]  
, // Column  
], // <Widget>[]  
, // Column  
), // MainAxisAlignment  
); // Center
```



Listeleme

ListView Sabit Liste

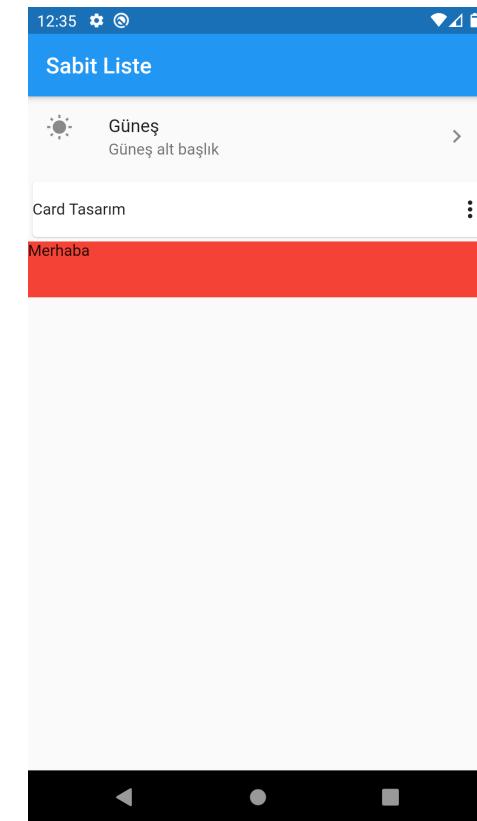
ListView : Sabit Liste

- ListTile dışında farklı satır tasarımları yapılabilir.

```
body: ListView(//Liste işlemi yapar.
  children: <Widget>[
    ListTile(//Listenin satırıdır.
      leading: Icon(Icons.wb_sunny), //Sol taraf iconu
      title: Text('Güneş'), //Başlık
      subtitle: Text("Güneş alt başlık"), //Alt başlık
      trailing: Icon(Icons.keyboard_arrow_right), //Sağ taraf iconu
      onTap: (){//Satırın tıklanılma metodudur
        print("Güneş tıklandı");
      },
    ), // ListTile
    GestureDetector(//Tıklanılma özelliği
      onTap: (){
        print("Card tıklandı");
      },
      child: Card(//Card tasarımını
        child: SizedBox(
          height: 50,
          child: Row(
            children: <Widget>[
              Text("Card Tasarımı"),
              Spacer(),
              Icon(Icons.more_vert),
            ], // <Widget>[]
          ), // Row
        ), // SizedBox
      ), // Card
    ), // GestureDetector
  ],
)
```

```
GestureDetector(
    onTap: () {
        print("Container tıklandı");
    },
    child: Container(
        height: 50,
        color: Colors.red,
        child: Text("Merhaba"),
    ), // Container
), // GestureDetector
], // <Widget>[]
) // ListView

```

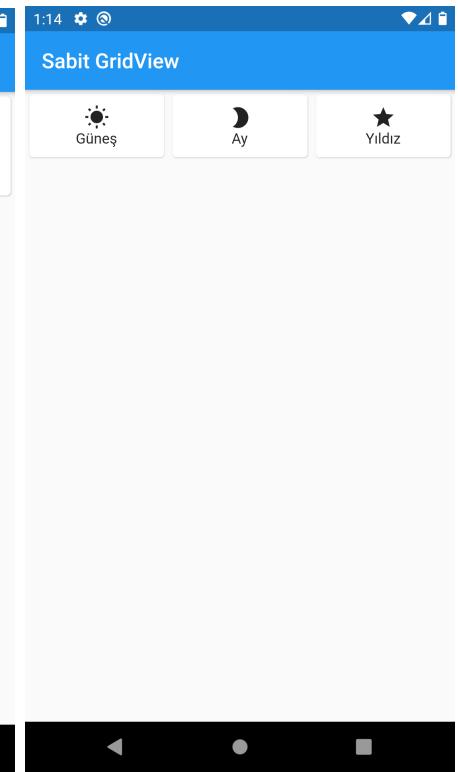
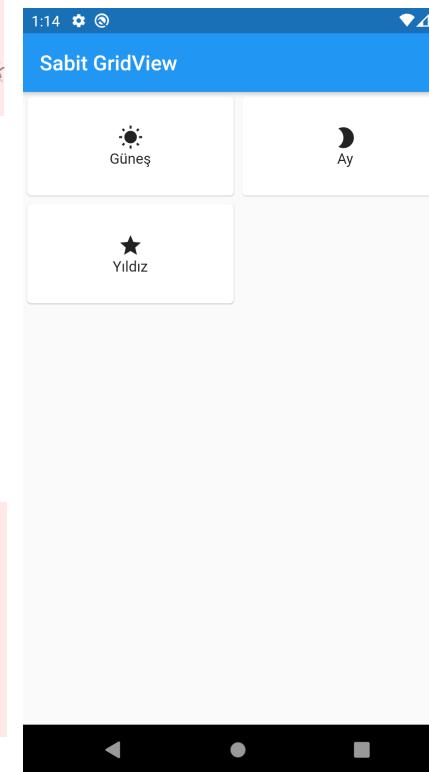


GridView Sabit Liste

GridView : Sabit Liste

- Satırlar tek tek yan yana veya alt alta oluşturduğumuz liste türündür.
 - İstenilen türde satır oluşturulabilir.

```
body: GridView.count(  
  crossAxisCount: 2, // Satırda gösterilecek item sayısı  
  childAspectRatio: 2 / 1, // Oranlama : Her bir satırın boyutunu belirler 2 : Genişlik , 1: Yükseklik  
  children: <Widget>[  
    GestureDetector(// Tıklanılma takibi  
      onTap: (){  
        print("Güneş Tıklandı");  
      },  
      child: Card(  
        child: Column(  
          mainAxisAlignment: MainAxisAlignment.center,  
          children: <Widget>[  
            Icon(Icons.wb_sunny),  
            Text("Güneş")  
          ], // <Widget>[]  
        ), // Column  
      ), // Card  
    ), // GestureDetector  
    Card(  
      child: Column(  
        mainAxisAlignment: MainAxisAlignment.center,  
        children: <Widget>[  
          Icon(Icons.brightness_2),  
          Text("Ay")  
        ], // <Widget>[]  
      ), // Column  
    ), // Card  
  ] // GridView.count
```

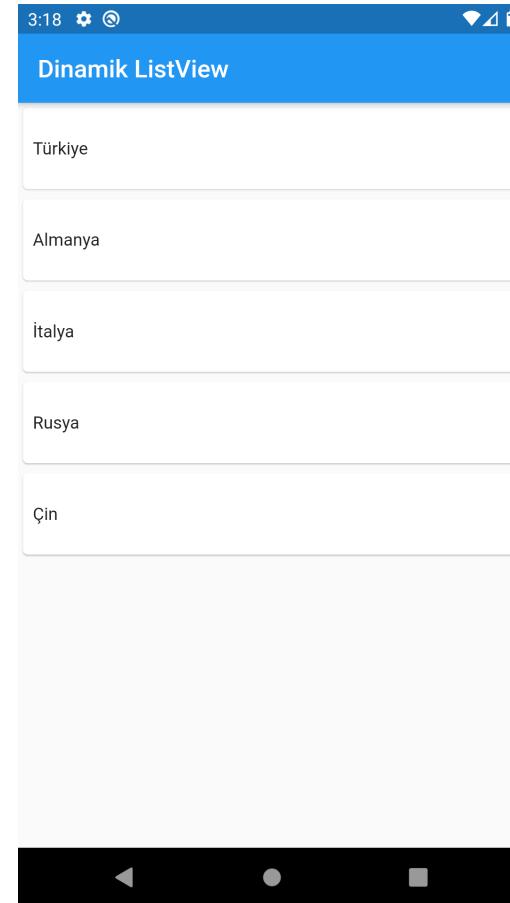


ListView Dinamik Liste

ListView.builder : Dinamik Liste

```
class _MyHomePageState extends State<MyHomePage> {
  var ulkeler = ["Türkiye", "Almanya", "İtalya", "Rusya", "Çin"];

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(widget.title),
      ), // AppBar
      body: ListView.builder(
        itemCount: ulkeler.length, // Gösterilecek veri boyutu
        itemBuilder: (context, indeks) {
          // Kaç tane veri varsa o kadar çalışır.
          // Sırayla dizinin indeks bilgisi gelir.
          return Card(
            child: Padding(
              padding: const EdgeInsets.all(8.0),
              child: SizedBox(
                height: 50,
                child: Row(
                  children: <Widget>[
                    Text(ulkeler[indeks]),
                    // İndeks bilgisini kullanarak dizi içine erişip
                    // dinamik olarak verileri diziden alabiliriz.
                  ],
                ),
              ),
            ),
          );
        },
      ), // ListView.builder
    ); // Scaffold
  }
}
```



Satır Tasarımına Tıklama

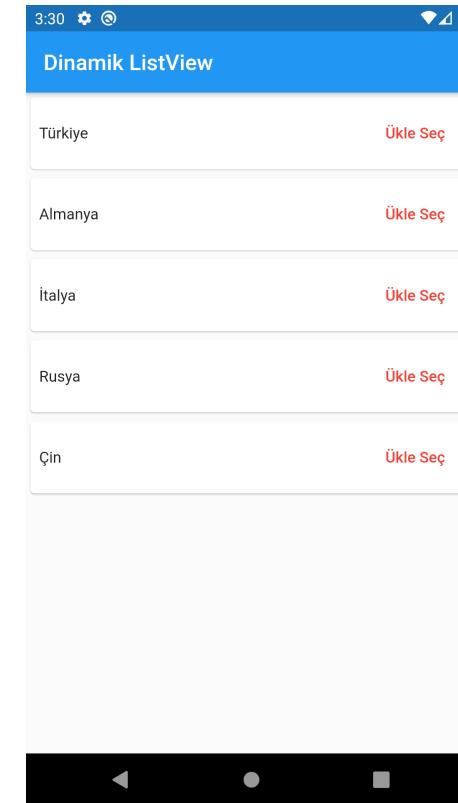
```
body: ListView.builder(  
    itemCount: ulkeler.length, // Gösterilecek veri boyutu  
    itemBuilder: (context, indeks){  
        // Kaç tane veri varsa o kadar çalışır.  
        // Sırayla dizinin indeks bilgisi gelir.  
        return GestureDetector(  
            onTap: (){  
                print("${ulkeler[indeks]} seçildi");  
            },  
            child: Card(  
                child: Padding(  
                    padding: const EdgeInsets.all(8.0),  
                    child: SizedBox(  
                        height: 50,  
                        child: Row(  
                            children: <Widget>[  
                                Text(ulkeler[indeks]),  
                                // İndeks bilgisini kullanarak dizi içine erişip  
                                // dinamik olarak verileri diziden alabiliriz.  
                            ], // <Widget>[]  
                        ), // Row  
                    ), // SizedBox  
                ), // Padding  
            ), // Card  
        ); // GestureDetector  
    },  
, // ListView.builder
```

Card'a tıklamak için
GestureDetector özelliği
kullanmalıyız.

Satır Üzerindeki Widgetlara Tıklama

- Varsayılan olarak tıklama özelliği olan widgetların *onPressed* özelliği kullanılabilir.
 - Varsayılan olarak tıklama özelliği yoksa *GestureDetector* kullanılabilir.

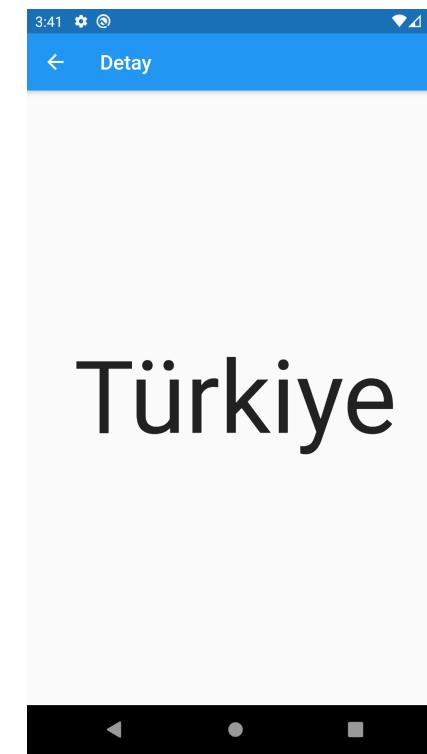
```
return GestureDetector(  
    onTap: (){  
        print("${ulkeler[indeks]} seçildi");  
    },  
    child: Card(  
        child: Padding(  
            padding: const EdgeInsets.all(8.0),  
            child: SizedBox(  
                height: 50,  
                child: Row(  
                    children: <Widget>[  
                        GestureDetector(  
                            onTap: (){  
                                print("Text ile ${ulkeler[indeks]} seçildi");  
                            },  
                            child: Text(ulkeler[indeks])  
                        ), // GestureDetector  
                        Spacer(),  
                        TextButton(  
                            child: Text("Ülke Seç",style: TextStyle(color: Colors.red)),  
                            onPressed: (){  
                                print("Button ile ${ulkeler[indeks]} seçildi");  
                            },  
                        ), // TextButton  
                    ], // <Widget>[]  
                ), // Row  
            ), // SizedBox  
        ), // Padding  
    ), // Card  
>; // GestureDetector
```



Satırı Tıklayıp Sayfa Geçişi

- Geçiş yapılacak sayfa oluşturulur.

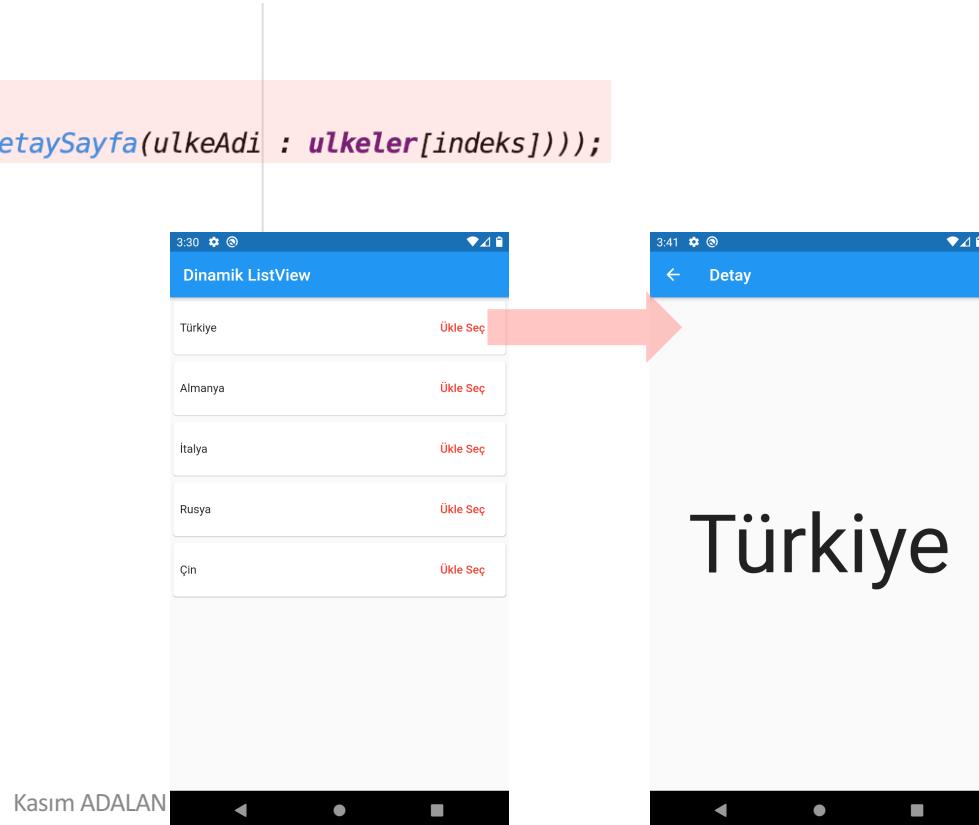
```
class DetaySayfa extends StatelessWidget {  
  
    String ulkeAdi;  
    DetaySayfa({required this.ulkeAdi});  
  
    @override  
    Widget build(BuildContext context) {  
        return Scaffold(  
            appBar: AppBar(  
                title: Text("Detay"),  
            ), // AppBar  
            body: Center(  
                child: Text("$ulkeAdi", style: TextStyle(fontSize: 100),),  
            ) // Center  
        ); // Scaffold  
    }  
}
```



Satırı Tıklayıp Sayfa Geçişi

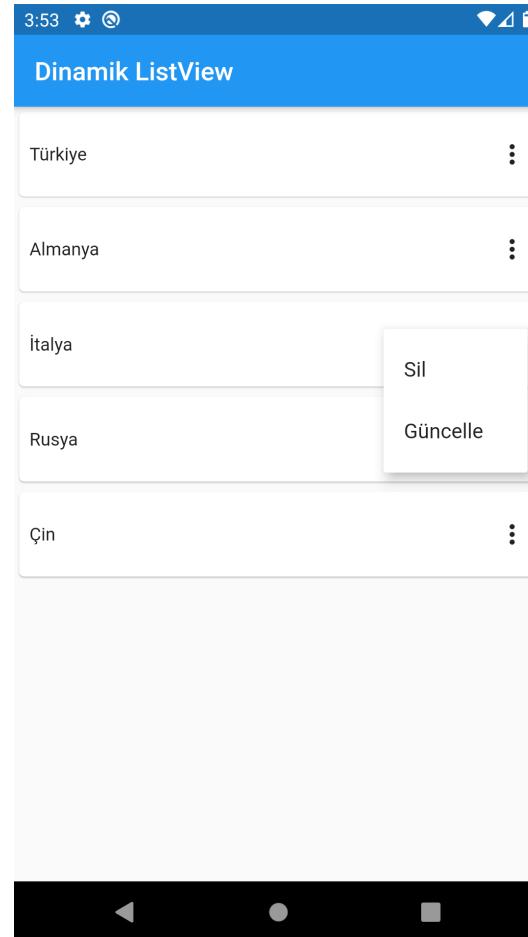
- Satırı tıklama işlemi

```
return GestureDetector(  
  onTap: (){  
    Navigator.push(context,  
      MaterialPageRoute(builder: (context) => DetaySayfa(ulkeAdi : ulkeler[indeks])));  
  },  
  child: Card(  
    child: Padding(  
      padding: const EdgeInsets.all(8.0),  
      child: SizedBox(  
        height: 50,  
        child: Row(  
          children: <Widget>[
```



Widgeta Tıklanınca Popup Menu Oluşturma

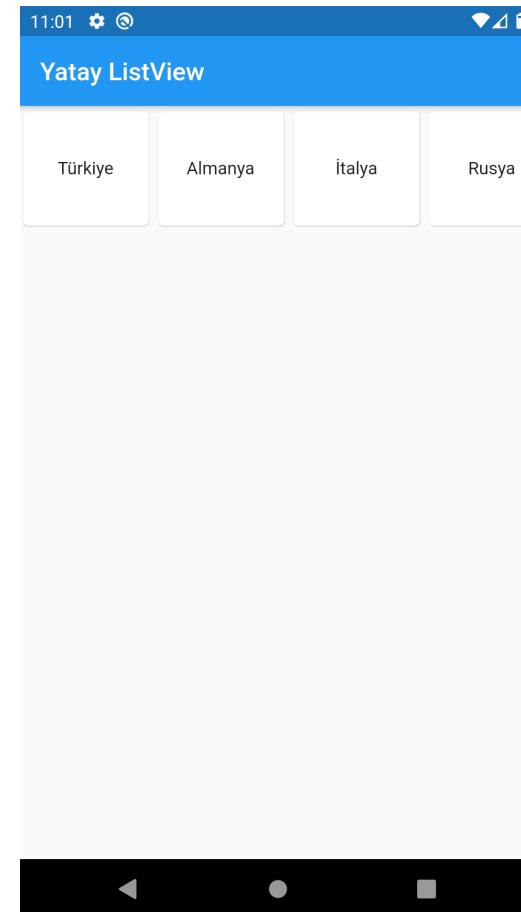
```
child: Card(  
  child: Padding(  
    padding: const EdgeInsets.all(8.0),  
    child: SizedBox(  
      height: 50,  
      child: Row(  
        children: <Widget>[  
          GestureDetector(  
            onTap: (){  
              print("Text ile ${ulkeler[indeks]} seçildi");  
            },  
            child: Text(ulkeler[indeks])  
          ), // GestureDetector  
          Spacer(),  
          PopupMenuButton(  
            child: Icon(Icons.more_vert),  
            itemBuilder: (context) => [  
              PopupMenuItem(value: 1,child: Text("Sil",),),  
              PopupMenuItem(value: 2,child: Text("Güncelle",),),  
            ],  
            onSelected: (menuItemValue){  
              if(menuItemValue == 1){  
                print("${ulkeler[indeks]} silindi");  
              }  
              if(menuItemValue == 2){  
                print("${ulkeler[indeks]} güncellendi");  
              }  
            },  
            // PopupMenuButton  
          ], // <Widget>[]  
        ), // Row  
      ), // SizedBox
```



Yatay ListView

- Yatay scroll edilebilir liste oluşturabiliriz.

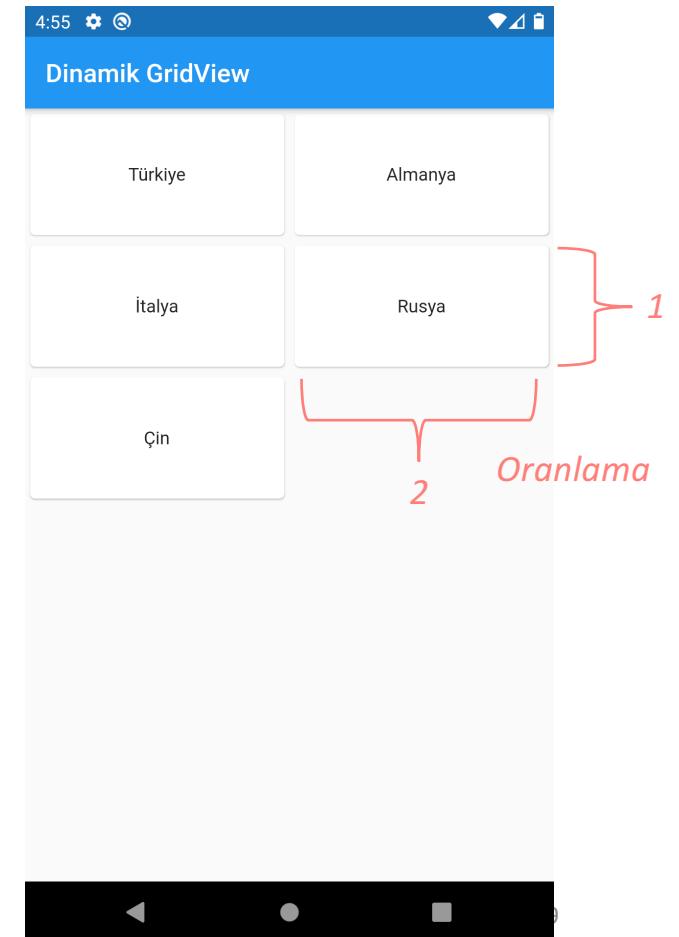
```
class _MyHomePageState extends State<MyHomePage> {  
  
  var ulkeler = ["Türkiye", "Almanya", "İtalya", "Rusya", "Çin";  
  
  @override  
  Widget build(BuildContext context) {  
    return Scaffold(  
      appBar: AppBar(  
        title: Text(widget.title),  
      ), // AppBar  
      body: SizedBox(//Listviewin yükseklikte kapladığı alanı sınırlamalıyız.  
        height: 100, //Aksi halde dikeyde tam sayfayı kaplar.  
        child: ListView.builder(  
          scrollDirection: Axis.horizontal, //Scroll yönü  
          itemCount: ulkeler.length,  
          itemBuilder: (context, indeks){  
            return Card(  
              child: SizedBox(  
                width: 100, //İçerik genişliği  
                child: Center(  
                  child: Column(  
                    mainAxisAlignment: MainAxisAlignment.center,  
                    children: <Widget>[  
                      Text(ulkeler[indeks]),  
                      ], // <Widget>[]  
                    ), // Column  
                  ), // Center  
                ), // SizedBox  
              ); // Card  
          },  
        ), // ListView.builder  
      ), // SizedBox  
    ); // Scaffold
```



GridView Dinamik Liste

GridView.builder : Dinamik Liste

```
class _MyHomePageState extends State<MyHomePage> {
  var ulkeler = ["Türkiye", "Almanya", "İtalya", "Rusya", "Çin"];
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(widget.title),
      ), // AppBar
      body: GridView.builder(
        gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
          crossAxisCount: 2 ,//Satırda item sayısı
          childAspectRatio: 2 / 1
          //item boyutu oranı , 2 : Genişlik , 1 : Yükseklik
          //crossAxisCount ile satırda item sayısına göre item genişliği değişir.
          //Örneğin : crossAxisCount : 2 olursa item genişliği ekran genişliğinin yarısı kadar olur
          //Oranlama 2 / 1 olduğu için yükseklikte ekran genişliğinin yarısının yarısı olur.
        ), // SliverGridDelegateWithFixedCrossAxisCount
        itemCount: ulkeler.length,
        itemBuilder: (context,indeks){
          //Kaç tane veri varsa o kadar çalışır.
          //Sırayla dizinin indeks bilgisi gelir.
          return Card(                                Satır tasarıımı
            child: Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: <Widget>[
                Text(ulkeler[indeks]),
              ], // <Widget>[]
            ), // Row
          ); // Card
        },
      ), // GridView.builder
    ); // Scaffold
}
```



Tasarıma Tıklama

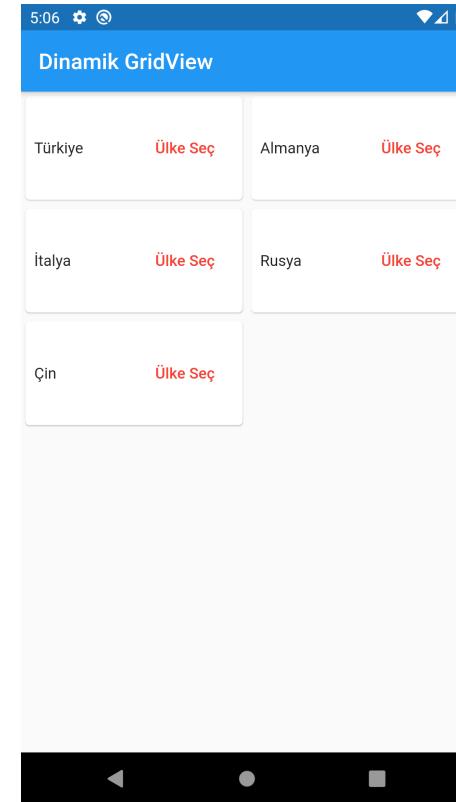
```
itemCount: ulkeler.length,  
 itemBuilder: (context, indeks){  
   //Kaç tane veri varsa o kadar çalışır.  
   //Sırayla dizinin indeks bilgisi gelir.  
   return GestureDetector(  
     onTap: (){  
       print("${ulkeler[indeks]} seçildi");  
     },  
     child: Card(  
       child: Row(  
         mainAxisAlignment: MainAxisAlignment.center,  
         children: <Widget>[  
           Text(ulkeler[indeks]),  
         ], // <Widget>[]  
       ), // Row  
     ), // Card  
   ); // GestureDetector  
},
```

Card'a tıklamak için
GestureDetector özelliği
kullanmalıyız.

Satır Üzerindeki Widgetlara Tıklama

- Varsayılan olarak tıklama özelliği olan widgetların *onPressed* özelliği kullanılabilir.
 - Varsayılan olarak tıklama özelliği yoksa *GestureDetector* kullanılabilir.

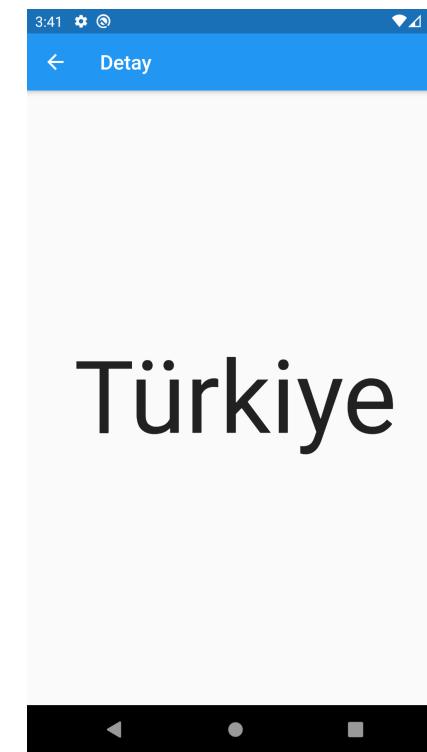
```
return GestureDetector(
  onTap: (){
    print("${ulkeler[indeks]} seçildi");
  },
  child: Card(
    child: Padding(
      padding: const EdgeInsets.all(8.0),
      child: Row(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          GestureDetector(
            onTap: (){
              print("Text ile ${ulkeler[indeks]} seçildi");
            },
            child: Text(ulkeler[indeks])), // GestureDetector
          Spacer(),
          TextButton(
            child: Text("Ülke Seç",style: TextStyle(color: Colors.red),),
            onPressed: (){
              print("Button ile ${ulkeler[indeks]} seçildi");
            },
          ), // TextButton
        ], // <Widget>[]
      ), // Row
    ), // Padding
  ), // Card
); // GestureDetector
```



Satırı Tıklayıp Sayfa Geçişi

- Geçiş yapılacak sayfa oluşturulur.

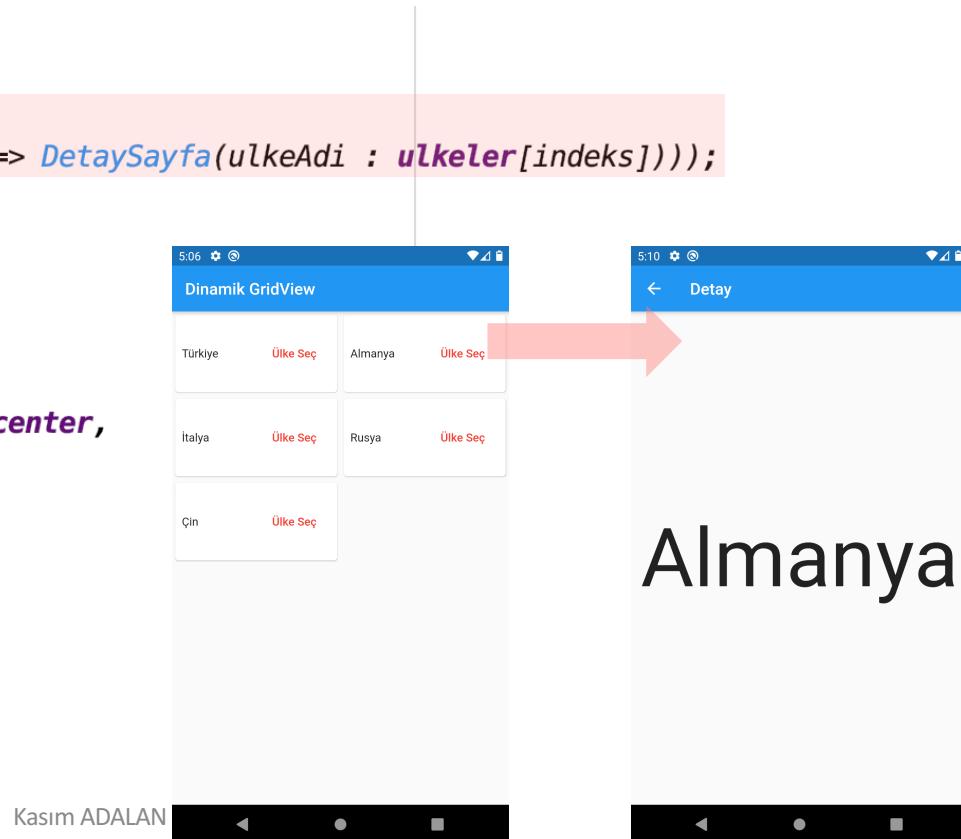
```
class DetaySayfa extends StatelessWidget {  
  
    String ulkeAdi;  
    DetaySayfa({required this.ulkeAdi});  
  
    @override  
    Widget build(BuildContext context) {  
        return Scaffold(  
            appBar: AppBar(  
                title: Text("Detay"),  
            ), // AppBar  
            body: Center(  
                child: Text("$ulkeAdi", style: TextStyle(fontSize: 100),),  
            ) // Center  
        ); // Scaffold  
    }  
}
```



Tasarıma Tıklayıp Sayfa Geçişi

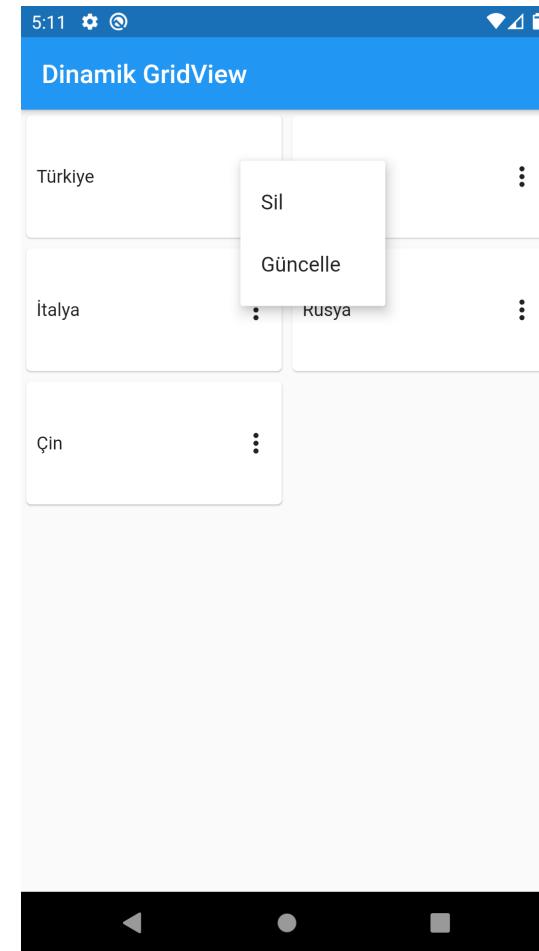
- Tasarıma tıklama işlemi

```
return GestureDetector(  
  onTap: (){  
    Navigator.push(context,  
      MaterialPageRoute(builder: (context) => DetaySayfa(ülkeAdı : ulkeler[indeks])));  
  },  
  child: Card(  
    child: Padding(  
      padding: const EdgeInsets.all(8.0),  
      child: Row(  
        mainAxisAlignment: MainAxisAlignment.center,  
        children: <Widget>[
```



Widget'a Tıklanınca Popup Menu Oluşturma

```
child: Card(  
  child: Padding(  
    padding: const EdgeInsets.all(8.0),  
    child: Row(  
      mainAxisAlignment: MainAxisAlignment.center,  
      children: <Widget>[  
        GestureDetector(  
          onTap: (){  
            print("Text ile ${ulkeler[indeks]} seçildi");  
          },  
          child: Text(ulkeler[indeks])), // GestureDetector  
        Spacer(),  
        PopupMenuButton(  
          child: Icon(Icons.more_vert),  
          itemBuilder: (context) => [  
            PopupMenuItem(value: 1,child: Text("Sil")),  
            PopupMenuItem(value: 2,child: Text("Güncelle")),  
          ],  
          onSelected: (menuItemValue){  
            if(menuItemValue == 1){  
              print("${ulkeler[indeks]} silindi");  
            }  
            if(menuItemValue == 2){  
              print("${ulkeler[indeks]} güncellendi");  
            }  
          },  
        ), // PopupMenuButton  
      ], // <Widget>[]  
    ), // Row  
  ), // Padding  
, // Card
```



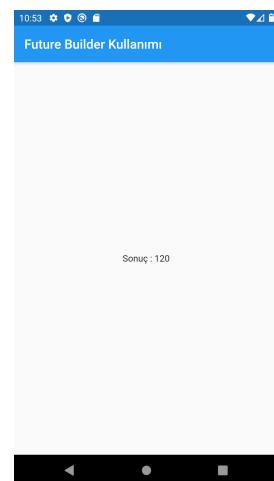
FutureBuilder ile Listeleme

FutureBuilder

- Asenkron işlemler için kullanılan bir yapıdır.
- async özelliği olan fonksiyonu kullanırken await özelliği ile sadece yapması gereken işlemi bitirene kadar çalışmasını sağlarız.
- Fakat await kullanmak için async özelliği olan fonksiyon içinde olmamız gereklidir.
- async özelliği olan fonksiyonu widget içinde kullanmak istediğimizde async özelliği olması gerekmektedir.Bu özellik widgetlarda yoktur.
- **Widget içinde async özelliğini kullanmak için FutureBuilder yapısı gereklidir.**

FutureBuilder Kullanımı

```
Future<int> faktoriyelHesapla(int sayi) async{  
  
    var sonuc = 1;  
  
    for(var i=1;i<=sayi;i++){  
        sonuc = sonuc * i ;  
    }  
  
    return sonuc;  
}
```



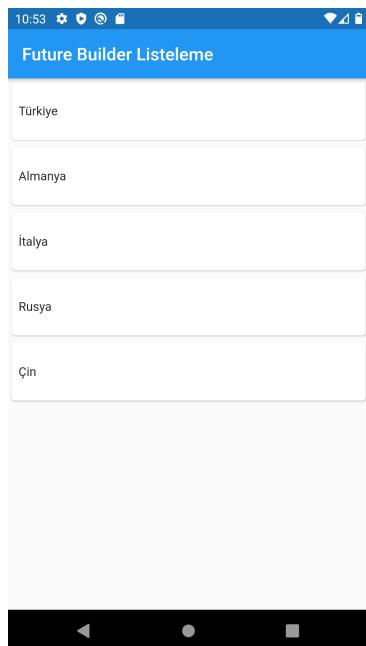
```
@override  
Widget build(BuildContext context) {  
    return Scaffold(  
        appBar: AppBar(  
            title: Text(widget.title),  
        ), // AppBar  
        body: Center(  
            child: Column(  
                mainAxisAlignment: MainAxisAlignment.center,  
                children: <Widget>[  
                    FutureBuilder<int>(  
                        future: faktoriyelHesapla(5),  
                        builder: (context,snapshot){  
                            if(snapshot.hasError){  
                                print("Hata sonucu : ${snapshot.error}");  
                            }  
  
                            if(snapshot.hasData){  
                                return Text("Sonuç : ${snapshot.data}");  
                            }else{  
                                return Center(child: Text("Gösterilecek Veri Yok"),);  
                            }  
                        },  
                    ), // FutureBuilder  
                ], // <Widget>[]  
            ), // Column  
        ), // Center  
    ); // Scaffold  
}
```

Annotations on the right side of the code:

- 'Çalıştıracağı fonksiyonun geri dönüş türü.' points to the return type of the FutureBuilder's future parameter.
- 'Çalıştırılacak fonksiyon' points to the function passed to the future parameter.
- 'Fonksiyonun çalışma sonucunu temsil eden değişken.' points to the snapshot variable used to access the result.
- 'Fonksiyonun sonuçına erişim.' points to the snapshot.data part of the code.
- 'Eğer gelen veri boş ise yani null ise burası çalışır ve tasarımda istediğimizi gösterebiliriz.' points to the conditional logic where an empty result leads to a specific UI.

FutureBuilder ile Listeleme Çalışması

```
Future<List<String>> verileriGetir() async{  
  
    var ulkeListesi = ["Türkiye", "Almanya", "İtalya", "Rusya", "Çin"];  
  
    //Bu metodun lokal veritabanı veya internet üzerindeki veritabanından  
    //bu listeyi aldığımızı düşünelim.  
    return ulkeListesi;  
}
```

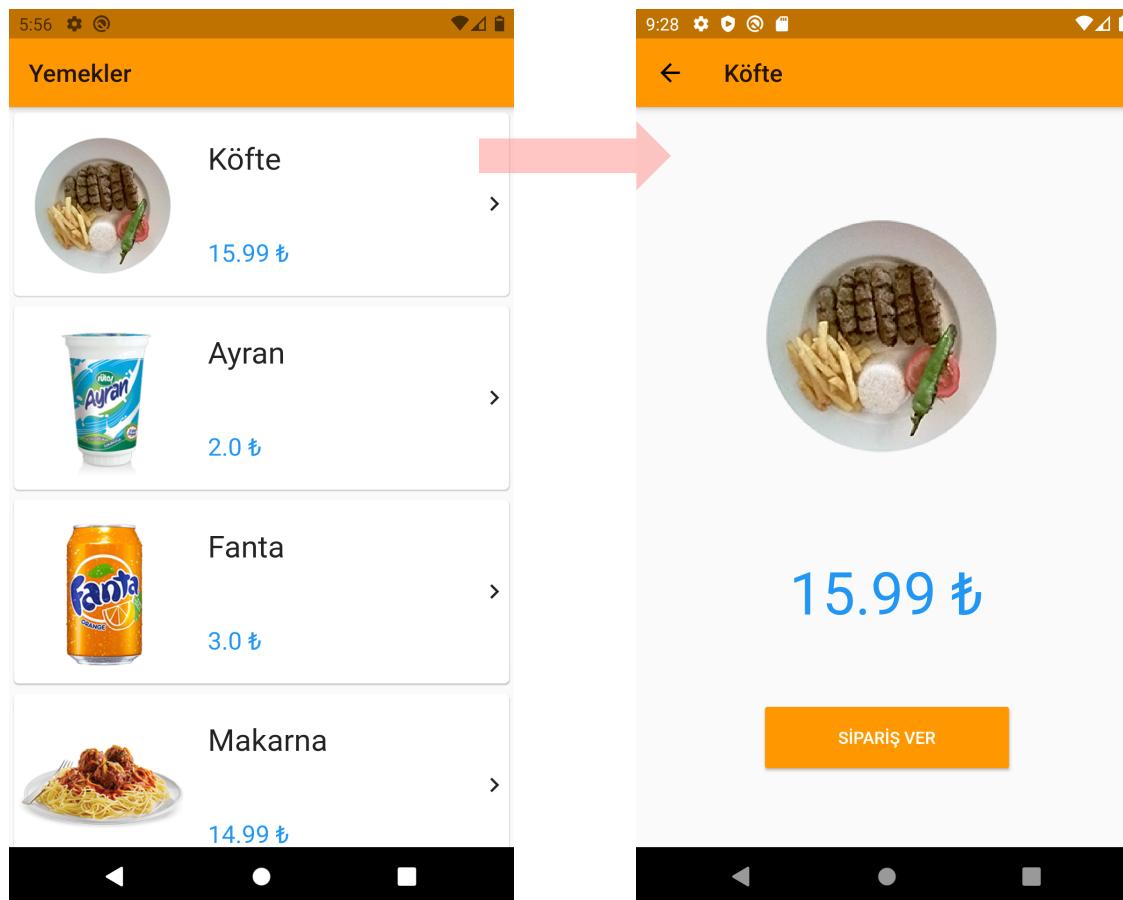


```
@override  
Widget build(BuildContext context) {  
    return Scaffold(  
        appBar: AppBar(  
            title: Text("")),  
        body: FutureBuilder<List<String>>(  
            future: verileriGetir(),  
            builder: (context,snapshot){  
                if(snapshot.hasData){  
                    var ulkeListesi = snapshot.data;  
  
                    return ListView.builder(  
                        itemCount: ulkeListesi!.length,  
                        itemBuilder: (context,indeks){  
                            var ülke = ulkeListesi[indeks];  
                            return Card(  
                                child: Padding(padding: const EdgeInsets.all(8.0),  
                                    child: SizedBox(height: 50,  
                                        child: Row( children: [Text(ulke),]),  
                                    ), // SizedBox  
                                ), // Padding  
                            ); // Card  
                        },  
                    ); // ListView.builder  
                }else{  
                    return Center();  
                }  
            },  
        ), // FutureBuilder  
    ); // Scaffold
```

Eğer gelen veri boş ise
yani null ise tasarım
boş görünsün

Düzenli ListView.builder

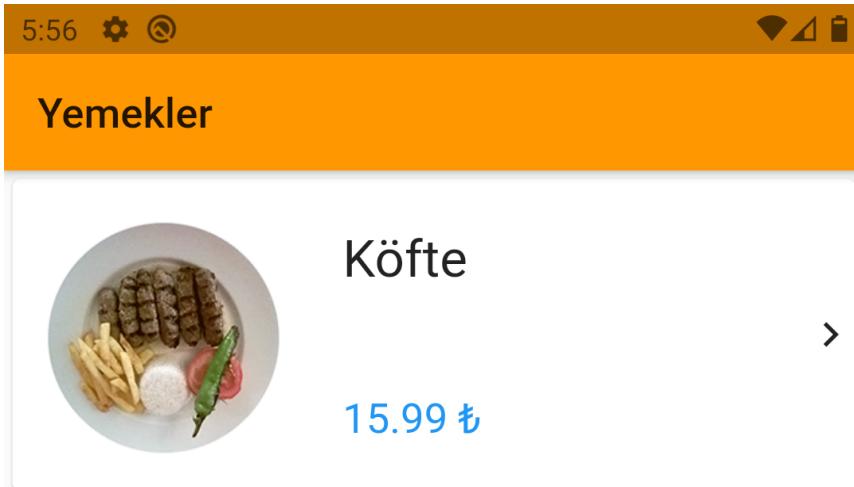
Detaylı ListView.builder



Nesne Tabanlı Çalışma

- Nesne tabanlı çalışmak için satır tasarıımına model olacak bir sınıf oluşturmalıyız.

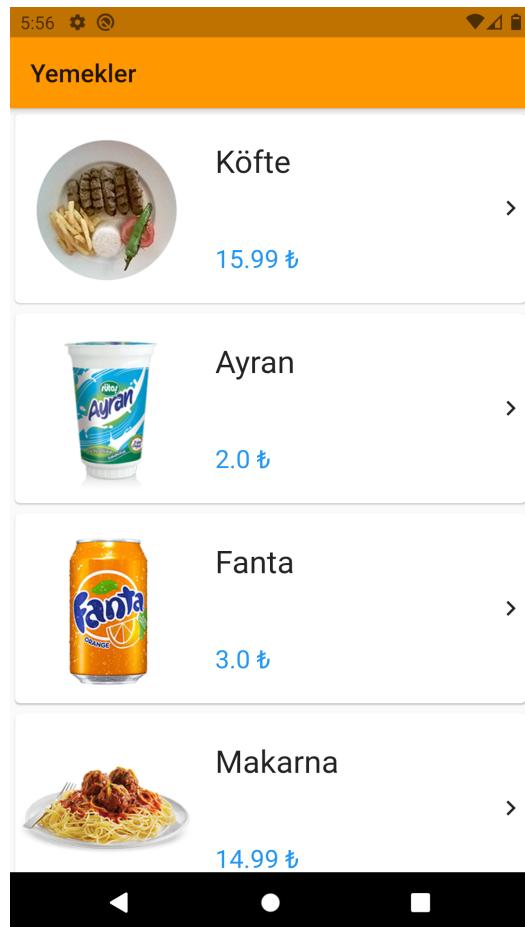
```
class Yemekler{  
    int yemek_id;  
    String yemek_adi;  
    String yemek_resim_adi;  
    double yemek_fiyat;  
  
    Yemekler(  
        this.yemek_id, this.yemek_adi, this.yemek_resim_adi, this.yemek_fiyat);  
}
```



Veri Kümesi Oluşturulur.

```
class _MyHomePageState extends State<MyHomePage> {  
  
  Future<List<Yemekler>> yemekleriGetir() async {  
    var yemekListesi = <Yemekler>[];  
  
    var y1 = Yemekler(1, "Köfte", "kofte.png", 15.99);  
    var y2 = Yemekler(2, "Ayran", "ayran.png", 2.0);  
    var y3 = Yemekler(3, "Fanta", "fanta.png", 3.0);  
    var y4 = Yemekler(4, "Makarna", "makarna.png", 14.99);  
    var y5 = Yemekler(5, "Kadayıf", "kadayıf.png", 8.50);  
    var y6 = Yemekler(6, "Baklava", "baklava.png", 15.99);  
  
    yemekListesi.add(y1);  
    yemekListesi.add(y2);  
    yemekListesi.add(y3);  
    yemekListesi.add(y4);  
    yemekListesi.add(y5);  
    yemekListesi.add(y6);  
  
    return yemekListesi;  
  }  
}
```

Satır Tasarımı



```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text(widget.title),
    ), // AppBar
    body: FutureBuilder<List<Yemekler>>(
      future: yemekleriGetir(),
      builder: (context,snapshot){
        if(snapshot.hasData){
          var yemekListesi = snapshot.data;
          return ListView.builder(
            itemCount: yemekListesi!.length,
            itemBuilder: (context,indeks){
              var yemek = yemekListesi[indeks];
              //Tasarım içinde kullanmak için Yemek nesnesi alındı.
              return GestureDetector(
                onTap: (){
                  Navigator.push(context,
                    MaterialPageRoute(builder: (context) => DetaySayfa(yemek : yemek)));
                },
                child: Card(
                  child: Row(
                    children: <Widget>[
                      SizedBox(width:150, height: 150, child: Image.asset("resimler/${yemek.yemek_resim_adi}")),
                      Padding(padding: const EdgeInsets.all(8.0),
                        child: Column(
                          crossAxisAlignment: CrossAxisAlignment.start,
                          children: <Widget>[
                            Text("${yemek.yemek_adi}",style: TextStyle(fontSize: 25),),
                            SizedBox(height: 50,)
```

Sayfa Geçişi



```
class _DetaySayfaState extends State<DetaySayfa> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("${widget.yemek.yemek_adi}"),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: <Widget>[
            Image.asset("resimler/${widget.yemek.yemek_resim_adi}"),
            Text("${widget.yemek.yemek_fiyat} \u20ac", style: TextStyle(color: Colors.blue, fontSize: 48)),
            Padding(padding: const EdgeInsets.all(8.0),
              child: SizedBox(
                width: 200,
                height: 50,
                child: RaisedButton(
                  color: Colors.orange,
                  textColor: Colors.white,
                  child: Text("SIPARIŞ VER"),
                  onPressed: () {
                    print("${widget.yemek.yemek_adi} sipariş verildi");
                  },
                ),
              ),
            ),
          ],
        ),
      );
    );
  }
}
```

Anasayfa

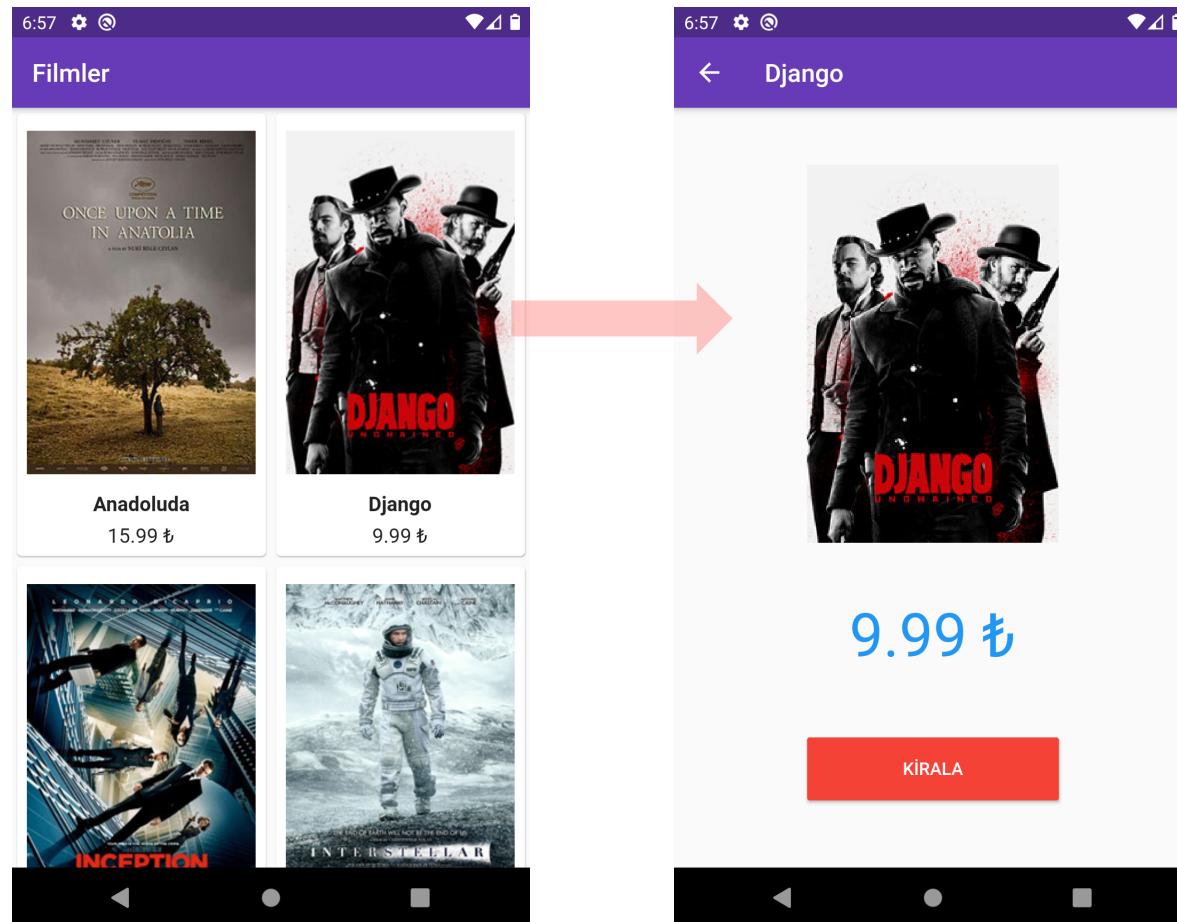
```
body: ListView.builder(
  itemCount: yemekListesi.length,
  itemBuilder: (context,indeks){
    var yemek = yemekListesi[indeks];
    //Tasarım içinde kullanmak için Yemek nesnesi alındı.
    return GestureDetector(
      onTap: (){
        Navigator.push(context,
          MaterialPageRoute(builder: (context) => DetaySayfa(yemek : yemek)));
      },
      child: Card(
        child: Row(
```

```
class DetaySayfa extends StatefulWidget {
  Yemekler yemek;
  DetaySayfa({required this.yemek});

  @override
  _DetaySayfaState createState() => _DetaySayfaState();
}
```

Detaylı GridView.builder

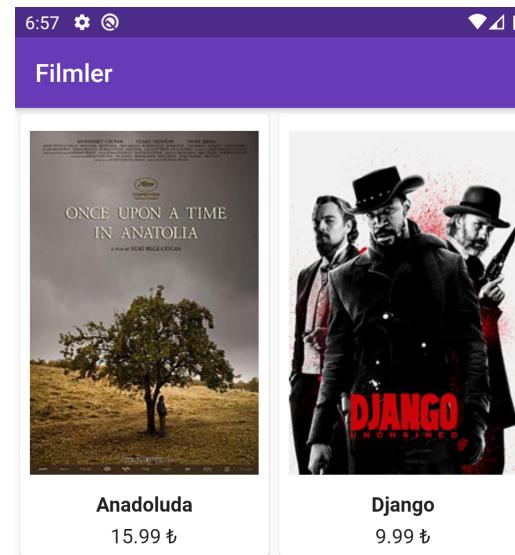
Detaylı GridView.builder



Nesne Tabanlı Çalışma

- Nesne tabanlı çalışmak için satır tasarıımına model olacak bir sınıf oluşturmalıyız.

```
class Filmler{  
    int film_id;  
    String film_adi;  
    String film_resim_adi;  
    double film_fiyat;  
  
    Filmler(this.film_id, this.film_adi, this.film_resim_adi, this.film_fiyat);  
}
```



Veri Kümesi Oluşturulur.

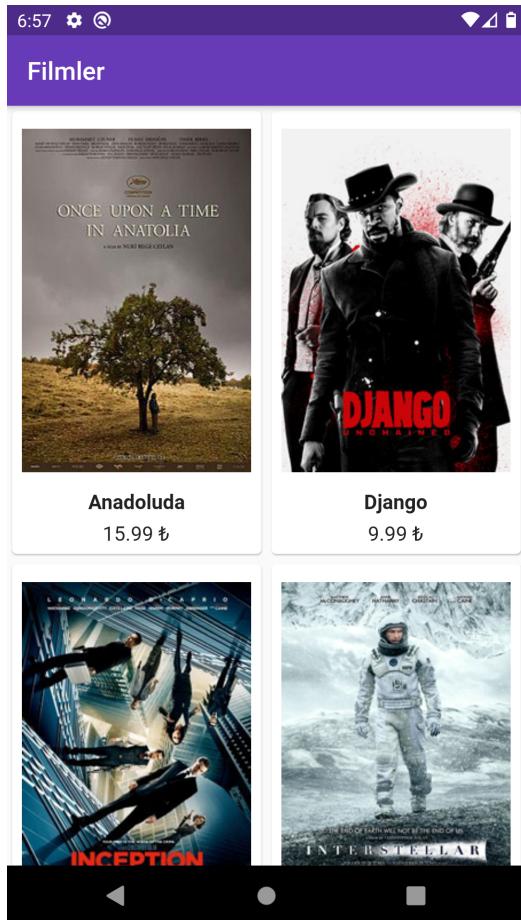
```
Future<List<Filmler>> filmleriGetir() async {
    var filmllerListesi = <Filmler>[];

    var f1 = Filmller(1,"Anadoluda","anadoluda.png",15.99);
    var f2 = Filmller(2,"Django","django.png",9.99);
    var f3 = Filmller(3,"Inception","inception.png",7.99);
    var f4 = Filmller(4,"Interstellar","interstellar.png",21.99);
    var f5 = Filmller(5,"The Hateful Eight","thehatefuleight.png",5.99);
    var f6 = Filmller(6,"The Pianist","thepianist.png",17.99);

    filmllerListesi.add(f1);
    filmllerListesi.add(f2);
    filmllerListesi.add(f3);
    filmllerListesi.add(f4);
    filmllerListesi.add(f5);
    filmllerListesi.add(f6);

    return filmllerListesi;
}
```

Satır Tasarımı



```
@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text(widget.title),
    ),
    body: FutureBuilder<List<Filmler>>(
      future: filmleriGetir(),
      builder: (context,snapshot){
        if(snapshot.hasData){
          var filmlerListesi = snapshot.data;
        }
      }
    )
  );
}

return GridView.builder(
  gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
    crossAxisCount: 2 ,//Satırda item sayısı
    childAspectRatio: 2 / 3.5
  ),
  itemCount: filmlerListesi!.length,
  itemBuilder: (context,indeks){
    var film = filmlerListesi[indeks];
    return GestureDetector(
      onTap: (){
        Navigator.push(context,
          MaterialPageRoute(builder: (context) => DetaySayfa(film : film)));
      },
      child: Card(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: <Widget>[
            Padding(
              padding: const EdgeInsets.all(8.0),
              child: Image.asset("resimler/${film.film_resim_adi}"),
            ), // Padding
            Text("${film.film_adi}",style: TextStyle(fontSize: 16,fontWeight: FontWeight.bold),),
            Text("${film.film_fiyat} \u20ba",style: TextStyle(fontSize: 16),),
          ],
        ),
      );
    );
  }
);
```

Sayfa Geçisi



```
class _DetaySayfaState extends State<DetaySayfa> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("${widget.film.film_adi}"),
      ), // AppBar
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: <Widget>[
            Image.asset("resimler/${widget.film.film_resim_adi}"),
            Text("${widget.film.film_fiyat} \u20ba", style: TextStyle(color: Colors.blue, fontSize: 48),),
            Padding(padding: const EdgeInsets.all(8.0),
              child: SizedBox(width: 200, height: 50,
                child: RaisedButton(
                  color: Colors.red,
                  textColor: Colors.white,
                  child: Text("KIRALA"),
                  onPressed: (){
                    print("${widget.film.film_adi} kiralandi");
                  },
                ), // RaisedButton
              ), // SizedBox
            ), // Padding
          ], // <Widget>[]
        ), // Column
      ), // Center
    ); // Scaffold
  }
}
```

```
body: GridView.builder(
  gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
    crossAxisCount: 2 ,//Satırda item sayısı
    childAspectRatio: 2 / 3.5
  ), // SliverGridDelegateWithFixedCrossAxisCount
  itemCount: filmListe.length,
  itemBuilder: (context,indeks){
    var film = filmListe[indeks];
    return GestureDetector(
      onTap: (){
        Navigator.push(context,
          MaterialPageRoute(builder: (context) => DetaySayfa(film : film)));
      },
    );
  }
);
```

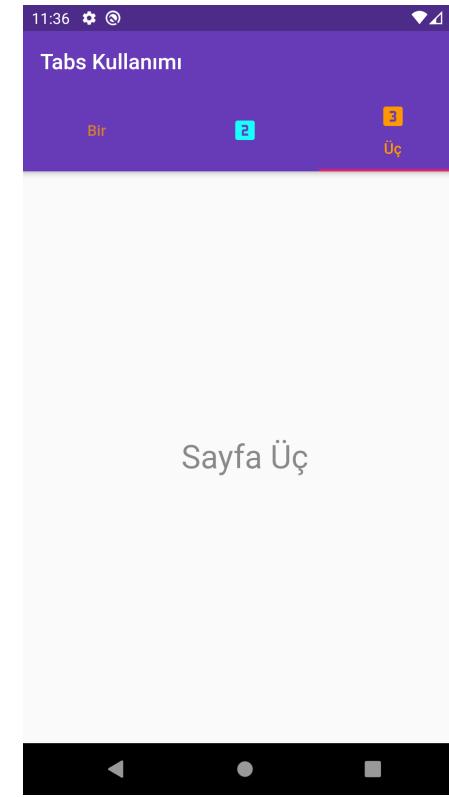
```
class DetaySayfa extends StatefulWidget {
  Filmler film;
  DetaySayfa({required this.film});
  @override
  _DetaySayfaState createState() => _DetaySayfaState();
}
```

Tabs

Tabs

- Tabs kullanarak aynı ekranda birden fazla sayfa gösterebiliriz.
- Tablara tıklayarak veya parmak hareketi ile sağa-sola kaydırarak sayfalar arasında geçiş yapabiliriz.

```
class _MyHomePageState extends State<MyHomePage> {  
  
  @override  
  Widget build(BuildContext context) {  
    return DefaultTabController(//Tab ile Sayfa açılmasını organize eden yapı  
      length: 3,//Tab sayısı  
      child: Scaffold(  
        appBar: AppBar(  
          title: Text(widget.title),  
          bottom: TabBar(//Appbar altında yer alan tabbar alanı  
            tabs: [  
              Tab(text: "Bir"),  
              Tab(icon: Icon(Icons.looks_two,color: Colors.cyanAccent,)),  
              //Bireysel olarak Icon rengi özelleştirme  
              Tab(text: "Üç",icon: Icon(Icons.looks_3)),  
            ],  
            indicatorColor: Colors.pink,//Belirteç rengi  
            labelColor: Colors.orange,//Yazı ve icon rengi  
          ), // TabBar  
        ), // AppBar  
        body: TabBarView(  
          //Tablara tıklanınca açılacak sayfalar  
          children: [  
            SayfaBir(),  
            SayfaIki(),  
            SayfaUc(),  
          ],  
        ), // TabBarView  
      ), // Scaffold  
    ); // DefaultTabController  
  }  
}
```



Sayfalar

```
class SayfaBir extends StatefulWidget {
  @override
  _SayfaBirState createState() => _SayfaBirState();
}

class _SayfaBirState extends State<SayfaBir> {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text("Sayfa Bir",style: TextStyle(color: Colors.black45,fontSize: 30),),
    ); // Center
  }
}

class SayfaIki extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text("Sayfa İki",style: TextStyle(color: Colors.black45,fontSize: 30),),
    ); // Center
  }
}

class SayfaUc extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text("Sayfa Üç",style: TextStyle(color: Colors.black45,fontSize: 30),),
    ); // Center
  }
}
```

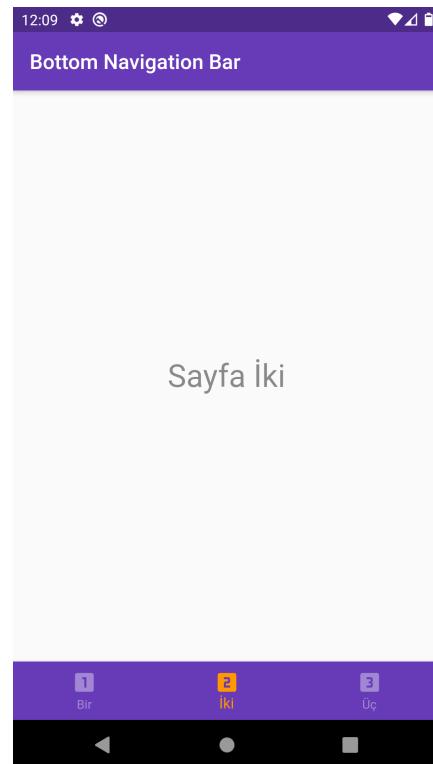
Bottom Navigation Bar

Bottom Navigation Bar

Bottom navigation bar, tabs gibi çalışan bir yapıdır.

Sayfanın altında çalışacak şekilde tasarlanmıştır.

Tabs gibi parmak hareketi ile yapılan kaydırma işlemine duyarlı değildir.



```

class _MyHomePageState extends State<MyHomePage> {

int secilenIndexs = 0;
//State ózellikleri olan secilen BottomNavigationBar indeksini takip için deðiþken
var sayfaListesi = [SayfaBir(),SayfaIki(),SayfaÜc()];
//Ekranada göstermek istedigimiz sayfaların listesi

@Override
Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
            title: Text(widget.title),
        ), // AppBar
        body: sayfaListesi[secilenIndexs],//Secilen indekse göre anlık olarak sayfa görüntülenir.
        //BottomNavigationBar fab gibi body dışında yer alır.
        bottomNavigationBar: BottomNavigationBar(
            items: [
                BottomNavigationBarItem(
                    icon: Icon(Icons.looks_one),
                    label: "Bir",
                ), // BottomNavigationBarItem
                BottomNavigationBarItem(
                    icon: Icon(Icons.looks_two),
                    label: "İki",
                ), // BottomNavigationBarItem
                BottomNavigationBarItem(
                    icon: Icon(Icons.looks_3),
                    label: "Üç",
                ), // BottomNavigationBarItem
            ],
        ),
    );
}

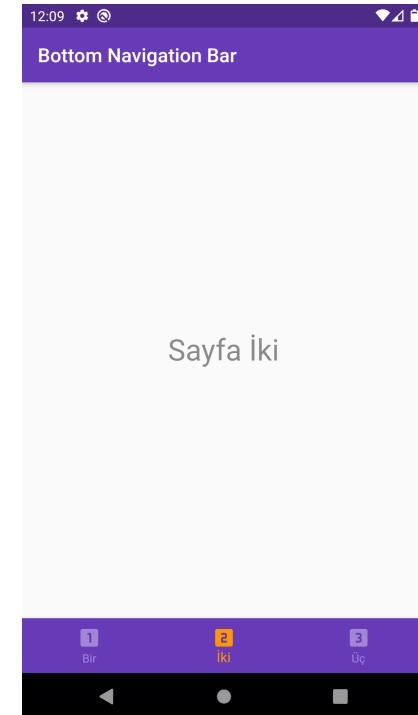
```

Kasım ADALAN

```

currentIndex: secilenIndexs,
//En son seçili olan indeks, secilenIndexs her seçimde degistiði için
//bar üzerinde seçim animasyonu olusur.
selectedItemColor: Colors.orange,//Secili durumda yazı ve icoñ rengi
unselectedItemColor: Colors.white38,//Secili olmadığı durumda yazı ve icoñ rengi
backgroundColor: Colors.deepPurple,//Arkaplan rengi
onTap: (indeks){//Seçim yapıldığında secilen indeks alınır.
    setState(() {
        secilenIndexs = indeks;
    });
},
), // BottomNavigationBar
); // Scaffold
}

```



56

Sayfalar

```
class SayfaBir extends StatefulWidget {
  @override
  _SayfaBirState createState() => _SayfaBirState();
}

class _SayfaBirState extends State<SayfaBir> {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text("Sayfa Bir",style: TextStyle(color: Colors.black45,fontSize: 30),),
    ); // Center
  }
}

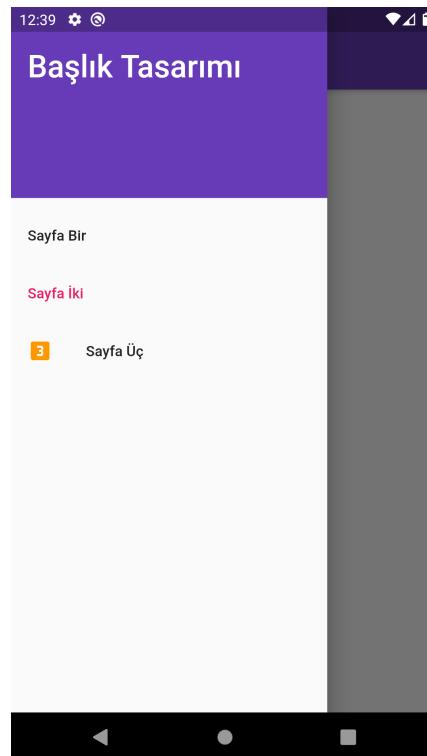
class SayfaIki extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text("Sayfa İki",style: TextStyle(color: Colors.black45,fontSize: 30),),
    ); // Center
  }
}
```

```
class SayfaÜç extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text("Sayfa Üç",style: TextStyle(color: Colors.black45,fontSize: 30),),
    ); // Center
  }
}
```

Drawer

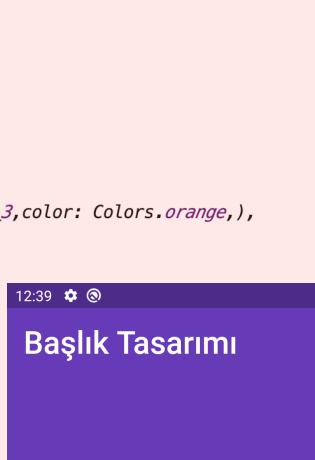
Drawer

- Drawer aynı ekran üzerinde birden fazla sayfa göstermek için kullandığımız bir yapıdır.
 - Parmak hareketine duyarlı şekilde açılır ve kapanabilir.
 - Geri tuşu ile kapatılabilir.

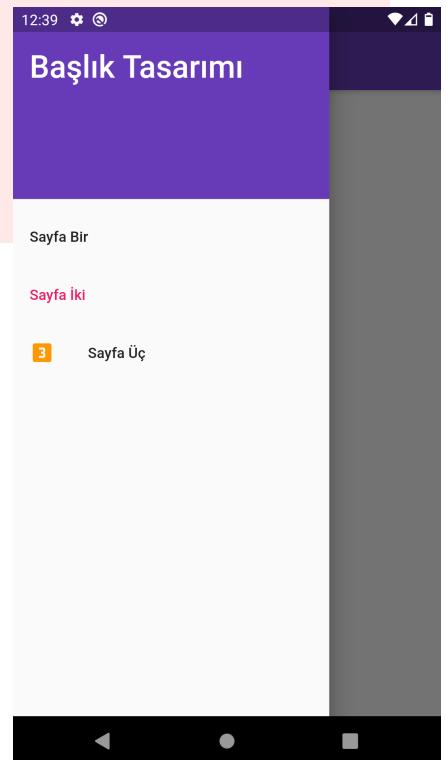


```
class _MyHomePageState extends State<MyHomePage> {
int secilenIndexs = 0;
//State özelliği olan seçilen BottomNavigationBar indeksini takip için değişken
var sayfaListesi = [SayfaBir(), SayfaIki(), SayfaUc()];
//Ekranда göstermek istediğimiz sayfaların listesi
@Override
Widget build(BuildContext context) {
return Scaffold(
  appBar: AppBar(
    title: Text(widget.title),
  ), // AppBar
  body: sayfaListesi[secilenIndexs], //Secilen indekse göre anlık olarak sayfa görüntülenir.
  //Drawer fab gibi body dışında yer alır.
  drawer: Drawer(
    child: ListView( //Drawer içindeki list görünümü
      padding: EdgeInsets.zero, //Başlığın ekranın üstüne sıfıra sıfır olmasını sağlar.
      children: <Widget>[
        DrawerHeader( //Başlık Alanı
          child: Text('Başlık Tasarımı', style: TextStyle(color: Colors.white, fontSize: 30),),
          decoration: BoxDecoration(
            color: Colors.deepPurple, //Başlık arkaplanı
          ), // BoxDecoration
        ), // DrawerHeader
        ListTile( //İtem
          title: Text('Sayfa Bir'), //İtem tasarımlı
          onTap: () { //İtema tıklanılma
            setState(() { //İtema tıklanılınca secilenIndexs değişkeni sayesinde sayfa değişimi.
              secilenIndexs = 0;
            });
            Navigator.pop(context); //Seçim sonrasında Drawer'ın kapanmasını sağlar.
          },
        ), // ListTile
      ],
    ),
  ),
)
```

```
  ListTile(
    title: Text('Sayfa İki', style: TextStyle(color: Colors.pink),),
    onTap: () {
      setState(() {
        secilenIndex = 1;
      });
      Navigator.pop(context);
    },
  ), // ListTile
  ListTile(
    leading: Icon(Icons.looks_3, color: Colors.orange,),
    title: Text('Sayfa Üç'),
    onTap: () {
      setState(() {
        secilenIndex = 2;
      });
      Navigator.pop(context);
    },
  ), // ListTile
], // <Widget>[]
), // ListView
// Drawer
// Scaffold
```



Sayfa Bir



Kasım ADALAN

Sayfalar

```
class SayfaBir extends StatefulWidget {
  @override
  _SayfaBirState createState() => _SayfaBirState();
}

class _SayfaBirState extends State<SayfaBir> {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text("Sayfa Bir",style: TextStyle(color: Colors.black45,fontSize: 30),),
    ); // Center
  }
}

class SayfaIki extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text("Sayfa İki",style: TextStyle(color: Colors.black45,fontSize: 30),),
    ); // Center
  }
}
```

```
class SayfaUc extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Center(
      child: Text("Sayfa Üç",style: TextStyle(color: Colors.black45,fontSize: 30),),
    ); // Center
  }
}
```

Teşekkürler...



kasim-adalan



kasimadalan@gmail.com



kasimadalan