

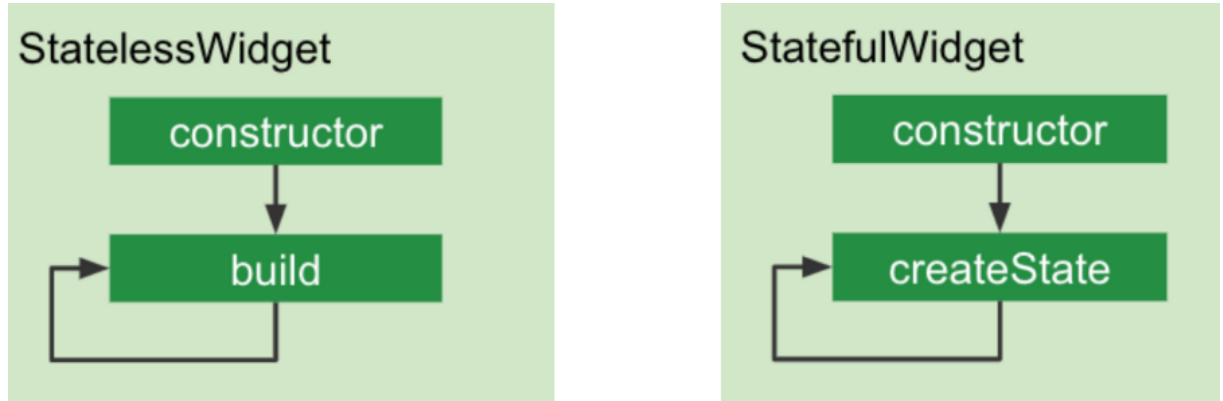
## Stateless ve Stateful Widgetlar

Flutter'da iki tür widget bulunur: Stateless ve Stateful. Bu ders notunda, her ikisinin de ne olduğunu ve ne zaman kullanılacağını öğreneceğiz.

[ <https://api.flutter.dev/flutter/widgets/StatelessWidget-class.html> ]

### State Kavramı

State uygulamamızın o anki görüntüsünü, durumunu temsil eder. Bir widget oluştuğunda eşzamanlı olarak durumu hakkında bilgi alınabilmesi, değiştiğinde bilginin eşzamanlı olarak yansması state yönetimi ile mümkün olmaktadır. State yenilendiğinde görüntümüzde yinelenir. Burada bir veri değişimi söz konusu olduğunda, state ekrana tekrar bastırılarak güncel görüntü elde edilir. Durum widgetları Stateless Widget ve Stateful Widget olarak ikiye ayrılır.



### Flutter Stateless Widget Nedir?

Stateless widget'lar statik olan yani herhangi bir etkileşim içermeyen widget'ımızdır. Bir kere oluşturulup görüntüledikten sonra içeriği hep sabit kalır değişmez. Bunlara metin, resim, sabit simgeler gibi örnekler verebiliriz. Stateless Widget kullanıldığında, kullanıcı arayüzü ekrana bir kere basılır ve tekrar yenilenmez. Sabit bir sayfa oluşturacaksa, sadece bilgi amaçlı ve interaktif bir sayfa olmayacaksa bu widget'ın kullanılması yerinde olur. Performans açısından daha başarılı sonuçlar elde edilebilir. Ancak günümüz itibari ile uygulamalar, sayfalar oldukça interaktif olduğu için bu pek mümkün görünmemektedir. Stateless Widget içerisine zorunlu olarak override edilen build metodu ile içerisindeki widgetı ekrana bastırır ve olay sonlanır. Sonraki bir veri değişiminde ekran yinelenmesi yapmaz.

Aşağıdaki görselde gözüktüğü üzere Stateless Widget build metodu ile ekrana görüntüyü bir kereliğine bastırır. Stateful Widget ise createState metodu ile bir state oluşturur ve oluşan bu state'i yenileme kabiliyetine sahiptir. Böylelikle sayfamızda bulunan dinamik veriler de güncellenmiş olur.

State bir widget'ın verilerini veya durumunu ifade eder. Örneğin, bir sayaç uygulamasında sayacın değeri bir state dir. Bir checkbox uygulamasında checkbox'un işaretli olup olmadığı bir state dir.

State'i daha iyi anlamak için bir analogi yapalım. Bir restorana gittiğinizde, garson siparişinizi (state) alır ve mutfaka iletir. Mutfak sipariş durumunu (state) değiştirir ve yemeği hazırlar. Garson daha sonra yemeği size getirir, yani güncellenmiş state'i size sunar. Stateless widget, menüdeki sabit yemekler gibidir; ne sipariş ederseniz edin, aldığınız şey değişmez. Stateful widget ise siparişinize özel istekler

ekleyebileceğiniz bir yemektir; nasıl pişirileceğini, hangi malzemelerin kullanılacağını belirleyebilir ve siparişiniz masanıza geldiğinde bu özel istekler yansıtılmış olur.

## Stateless Widgetlar

Stateless widgetlar, durumları (state) değişmeyen widgetlardır. Bir kez oluşturulur ve durumları sabit kalır. Bir Stateless widget'ın build metodu sadece bir kez çağrılır.

Örneğin, bir metin widget'ı Stateless bir widgettır, çünkü metni değişmez.

```
class MyText extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return Text("Merhaba Dünya!");  
  }  
}
```

## Stateful Widgetlar

Stateful widgetlar, durumları (state) değişebilen widgetlardır. Örneğin, bir sayaç uygulaması veya bir checkbox Stateful bir widget olmalıdır, çünkü durumları kullanıcı etkileşimi ile değişebilir.

Stateful widgetlar iki sınıftan oluşur: Widget sınıfı ve State sınıfı. Widget sınıfı kullanıcı arayüzünü oluşturur ve State sınıfı widget'ın durumunu (state) tutar.

## Uygulama 1: Sayaç Uygulaması

Bu uygulama, bir Stateful widget olan bir sayaç içerir. Kullanıcı bir butona bastığında sayaç değeri artar.

[ <https://flutlab.io/editor/26f853e6-188f-48b5-a178-b8fb956d10b4> ]

```
class MyCounterWidget extends StatefulWidget {  
  @override  
  _MyCounterWidgetState createState() => _MyCounterWidgetState();  
}
```

```

class _MyCounterWidgetState extends State<MyCounterWidget> {
  int count = 0;

  void incrementCounter() {
    setState(() {
      count++;
    });
  }

  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        Text("Count: $count"),
        ElevatedButton(
          onPressed: incrementCounter,
          child: Text("Arttır"),
        ),
      ],
    );
  }
}

```

Bu widget bir sayacı temsil eder ve sayacın değerini arttırmak için bir düğme içerir. “createState” ile bir sınıf oluşturulur ve bu sınıf state’i (durumu) yönetir. Ardından count yaratılır ve setState ile bu count her seferinde +1 artacak şekilde fonksiyona tanımlanır.

## Uygulama 2: Checkbox Uygulaması

Bu uygulama, bir Stateful widget olan bir checkbox içerir. Kullanıcı checkbox’u işaretlediğinde veya kaldırdığında, checkbox’un durumu değişir.

```

class CheckboxApp extends StatefulWidget {
  @override
  _CheckboxAppState createState() => _CheckboxAppState();
}

class _CheckboxAppState extends State<CheckboxApp> {
  bool _isChecked = false;

  void _toggleCheckbox(bool? value) {
    setState(() {
      _isChecked = value ?? !_isChecked;
    });
  }

  @override

```

```

Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text('Checkbox Uygulaması'),
    ),
    body: Center(
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          Checkbox(
            value: _isChecked,
            onChanged: _toggleCheckbox,
          ),
          Text(
            _isChecked ? 'Checkbox işaretli' : 'Checkbox
işaretsiz',
            style: Theme.of(context).textTheme.headline6,
          ),
        ],
      ),
    ),
  );
}

```

Bu örneklerde, Stateful widgetların nasıl oluşturulduğunu ve durumlarının nasıl değiştirildiğini gördük. setState metodunun, Flutter'a widget'ın durumunun değiştiğini ve yeniden build edilmesi gerektiğini bildirdiğine dikkat edin.

Genel olarak, bir widget'ın durumu değişmezse Stateless widget kullanılır. Ancak, bir widget'ın durumu kullanıcı etkileşimi veya diğer etkenlerle değişebiliyorsa, Stateful widget kullanılmalıdır.