

FLUTTER

Değişkenler ve Veri Türleri

5. hafta • Dart Temelleri ve Etkileşim

36 hafta, 72 bağımsız çalışan Flutter projesi, gerçek uygulama ekranları, uygulamalı dersler ve öğrenci geri bildiriminden sonra incelenebilen çözüm kaynakları.

Bu kitap güncel ders kaynaklarından otomatik üretilmiştir; özet ya da jenerik şablon değildir.

Öğrenme Haritası

Hafta	Ders	Projeler
5	Değişkenler ve Veri Türleri	Alışveriş Tutarı • Su Tüketim Takibi

5. Hafta — Değişkenler ve Veri Türleri

Bu hafta ne yapacağız?

Bir alışverişte birim fiyat ve ürün adedini kullanıcıdan alıp ödenecek tutarı hesaplayacağız. Böylece `String`, `int`, `double`, `bool`, `final` ve `const` değerlerin uygulamada neden farklı görevleri olduğunu göreceğiz. Ardından aynı bilgiyi günlük su tüketimini izleyen özgün bir projede kullanacağız.

Kazanımlar

- Dart veri türlerini gerçek bir problemde seçme
- Metin girdisini sayıya güvenli biçimde dönüştürme
- Değişebilen ve değişmeyen değerleri ayırt etme
- Ondalıkli sayılarla hesap yapıp sonucu biçimlendirme
- Geçersiz veride hesaplama yerine anlaşılır hata gösterme

Uygulama 05 — Alışveriş Tutarı

1. Hangi veri hangi türde?

Veri	Tür	Neden?
TextField içeriği	<code>String</code>	Kullanıcı girdisi önce metindir.
Ürün adedi	<code>int</code>	Tam sayı olmalıdır.
Birim fiyat ve toplam	<code>double</code>	Kuruş içerebilir.
Hata mesajı	<code>String?</code>	Hata yokken <code>null</code> olabilir.

2. Girdileri yönetme

```
final fiyatController = TextEditingController(text: '42,50');
final adetController = TextEditingController(text: '3');
double? toplam;
String? hata;
```

`final`, controller değişkeninin başka bir controller'a atanmayacağını söyler; controller içindeki metin yine değişebilir. `double?` ise sonucun hesaplama yapılmadan önce boş olabildiğini sağlar.

3. Metni sayıya dönüştürme

```
final fiyat = double.tryParse(
  fiyatController.text.replaceAll(',', '.'),
);
final adet = int.tryParse(adetController.text);
```

`tryParse`, geçersiz metinde uygulamayı durdurmaz; `null` döndürür. Türkçe ondalık virgülü, dönüşümden önce noktaya çevrilir.

4. Doğrulama ve hesaplama

```
if (fiyat == null || fiyat <= 0 || adet == null || adet <= 0) {  
  hata = 'Fiyat ve adet sifirdan büyük olmalıdır.';  
  toplam = null;  
} else {  
  hata = null;  
  toplam = fiyat * adet;  
}
```

Sonuç iki ondalık basamakla yazdırılır:

```
Text('₺${toplam!.toStringAsFixed(2)}')
```

Testte $42,50 \times 3$ işlemi yapıldı; uygulamanın 127.50 sonucunu ürettiği doğrulandı.

Alışveriş Tutarı



Sepet toplamını hesapla

Birim fiyat

₺ 42,50

Ürün adedi

 3

Σ Toplamı hesapla

Ödenecek tutar

₺127.50

3 ürün × ₺42.50

42,50 TL fiyat ve 3 adet için hesaplanan alışveriş tutarı

Özgün proje — Su Tüketim Takibi

Bu projede günlük hedef değişmeyen bir tam sayı, içilen bardak sayısı ise değişen bir tam sayıdır:

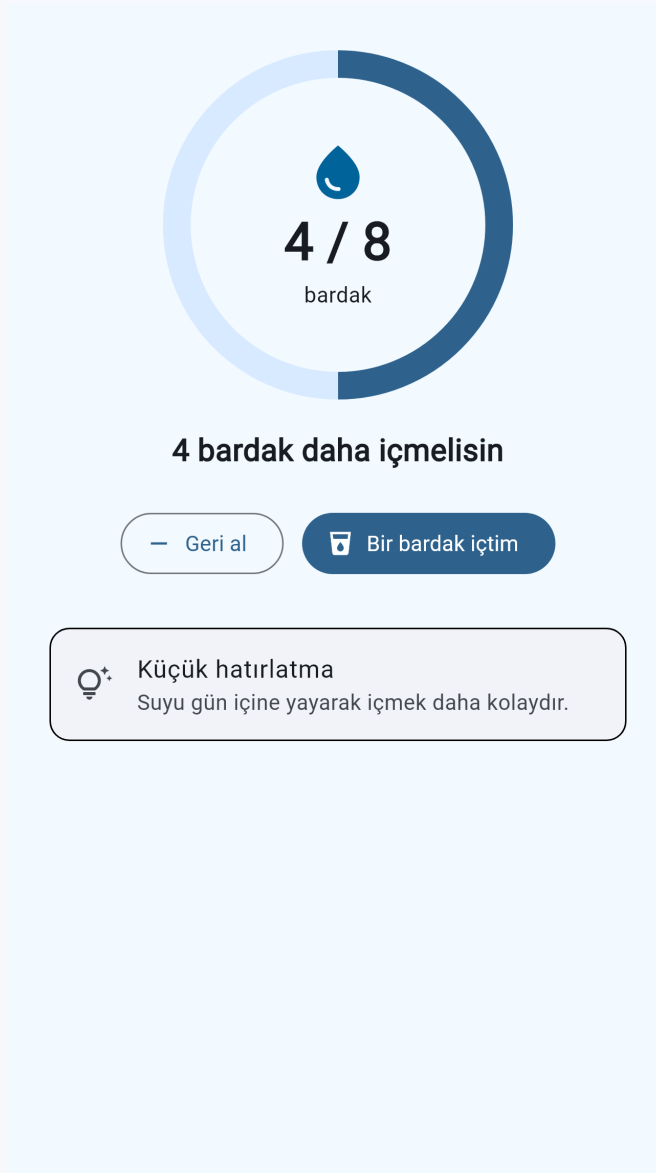
```
static const hedef = 8;  
int bardak = 3;  
final oran = bardak / hedef;
```

hedef derleme zamanından beri sabit olduğu için const, bardak kullanıcı eylemiyle değiştiği için normal int olur. Bölme işlemi double bir oran üretir ve CircularProgressIndicator bu oranı görselleştirir.

```
onPressed: bardak < hedef  
  ? () => setState(() => bardak++)  
  : null
```

Hedefe ulaşıldığında ekleme düğmesi devre dışı kalır. Testte “Bir bardak içtim” düğmesine basıldı; sayacın 4 / 8, mesajın “4 bardak daha içmelisin” olduğu doğrulandı.

Su Takibim



Dört bardak içilmiş Su Tüketim Takibi ekranı

Sıra sende

1. Kolay: Su hedefini 10 yap ve başlangıç miktarını değiştir.
2. Orta: Alışveriş hesabına yüzde indirim girdisi ekle.
3. İleri: Su miktarını bardak yerine mililitre olarak al; toplam litreyi iki ondalık basamakla göster.

Çözümünü gönderdikten ve öğretmen geri bildirimi aldıktan sonra Su Tüketim Takibi'nin gerçek Dart çözüm kodu ile çalışan proje indirme bağlantısı açılacaktır. İpucu yalnız veri türü seçimine yöneltecek, örnek çözüm eksiksiz `lib/main.dart` içeriğini gösterecektir.

Kontrol listesi

- Tam sayılar `int`, ondalıklı değerler `double` olarak tutuldu.
- Kullanıcı girdisi doğrudan zorla dönüştürülmedi; `tryParse` kullanıldı.
- Sıfır, negatif ve geçersiz değerler hesaplama alınamadı.
- Controller'lar `dispose` edildi.
- Her iki proje Flutter widget testiyle çalıştırıldı ve gerçek ekran çıktıları görsel olarak doğrulandı.