

FLUTTER

Hata Ayıklama ve Test

32. hafta • Cihaz Özellikleri ve Kalite

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
32	Hata Ayıklama ve Test	Hatalı Hesap Makinesi • Testli Not Hesabı

32. Hafta — Hata Ayıklama ve Test

Bu hafta ne yapacağız?

Çalışıyor görünen bir hesaplamanın neden yanlış sonuç üretebildiğini inceleyeceğiz. Hesaplama kodunu arayüzden ayıracak, normal değerlerin yanında sınır ve geçersiz değerleri test edeceğiz. Ardından aynı yaklaşımı ağırlıklı not hesabı yapan özgün bir uygulamada kullanacağız.

Kazanımlar

- Sözdizimi, çalışma zamanı ve mantık hatalarını ayırt etme
- Hesaplama fonksiyonunu widget kodundan ayırma
- tryParse ile kullanıcı girdisini güvenli dönüştürme
- ArgumentError ile sözleşme dışı değerleri reddetme
- Birim testi ve widget testinin görevlerini ayırt etme
- Normal, sınır ve geçersiz test verileri seçme
- Flutter golden testiyle gerçek arayüz çıktısını doğrulama

Hata türleri

Tür	Örnek	Nasıl bulunur?
Sözdizimi	Kapanmayan parantez	Derleyici mesajı
Çalışma zamanı	null değerde işlem	Hata yığını ve tekrar üretme
Mantık	Yüzdeyi 100'e bölmek	Beklenen–gerçek sonuç karşılaştırması
Sınır	100 indirimde negatif sonuç	Sınır değer testi
Doğrulama	“abc” metnini sayıya çevirmek	Geçersiz girdi testi

Çözümlü uygulama — Hata Avcısı: İndirim

Hatalı yaklaşım

```
final sonuc = fiyat - oran;
```

800 TL üründe yüzde 25 indirim, 775 TL değildir. Oran önce yüzdelik değere çevrilmelidir.

Düzeltilmiş ve doğrulanan fonksiyon

```
double indirimliTutar({
  required double fiyat,
  required double oran,
}) {
  if (fiyat < 0) throw ArgumentError('Fiyat negatif olamaz');
  if (oran < 0 || oran > 100) {
    throw ArgumentError('İndirim oranı 0-100 arasında olmalı');
  }
  return fiyat * (1 - oran / 100);
}
```

Fonksiyon arayüzden bağımsız olduğu için küçük ve hızlı bir birim testiyle denetlenebilir.

Girdiyi güvenli okuma

```
final f = double.tryParse(fiyat.text.replaceAll(',', '.'));
final o = double.tryParse(oran.text.replaceAll(',', '.'));

if (f == null || o == null) {
  setState(() {
    tutar = null;
    sonuc = 'Fiyat ve oran sayısal olmalı';
  });
  return;
}
```

`double.parse` geçersiz metinde hata fırlatırken `tryParse` null döndürür. Formlarda kullanıcıya anlaşılır geri bildirim vermek için bu davranış daha uygundur.

Birim testi

```
test('indirim formülü ve sınırlar doğru', () {
  expect(indirimliTutar(fiyat: 800, oran: 25), 600);
  expect(indirimliTutar(fiyat: 800, oran: 0), 800);
  expect(indirimliTutar(fiyat: 800, oran: 100), 0);
  expect(
    () => indirimliTutar(fiyat: 100, oran: 101),
    throwsArgumentError,
  );
});
```

Yalnız 25 gibi normal bir değer değil, 0 ve 100 sınırları ile 101 geçersiz değeri de test edilir.

Hata Avcısı: İndirim


600.00 TL
200.00 TL kazanç

Etiket fiyatı

800

TL

İndirim oranı

25

%

 Hesapla ve doğrula

Düzeltilen hata: yüzde doğrudan fiyattan çıkarılmaz;
oran önce 100'e bölünür. 0 ve 100 sınır değerleri de
test edilmelidir.

Hata Avcısı gerçek Flutter ekranı

Özgün uygulama — Testli Not Hesabı

Sonuç modeli

```
class NotSonucu {  
  const NotSonucu(this.ortalama, this.harf, this.gecti);  
  
  final double ortalama;  
  final String harf;  
  final bool gecti;  
}
```

Fonksiyon yalnız bir sayı değil, arayüzün ihtiyaç duyduğu bütün sonucu tek nesnede döndürür.

Ağırlıklı ortalama ve sınırlar

```
for (final not in [sinav, proje, uygulama]) {  
  if (not < 0 || not > 100) {  
    throw ArgumentError('Notlar 0-100 arasında olmalı');  
  }  
}  
  
final ortalama = sinav * .4 + proje * .25 + uygulama * .35;
```

Ağırlıkların toplamı 1'dir: $.40 + .25 + .35 = 1.00$.

Harf notu

```
final harf = switch (ortalama) {  
  >= 85 => 'A',  
  >= 70 => 'B',  
  >= 55 => 'C',  
  >= 45 => 'D',  
  _ => 'F',  
};
```

Koşullar yüksek değerden düşüğe doğru yazılır. Ters sıralama yapılırsa örneğin 90 değeri ilk düşük koşula takılarak yanlış harf üretebilir.

Test edilen davranışlar

```
final sonuc = notHesapla(  
  sinav: 70,  
  proje: 90,  
  uygulama: 80,  
);  
  
expect(sonuc.ortalama, 78.5);  
expect(sonuc.harf, 'B');  
expect(sonuc.gecti, isTrue);
```

Ek testler negatif notu reddeder ve tam 50 ortalamasının geçme eşiğinde kabul edildiğini doğrular.

Testli Not Hesabı

B

Ortalama 78.5
Dersi geçti

Sınav (%40)	70	/100
Proje (%25)	90	/100
Uygulama (%35)	80	/100

Hesapla ve test et

Birim testleri ağırlıkları, 0 ve 100 sınırlarını, geçme eşiğini ve geçersiz notları ayrı ayrı denetler.

Testli Not Hesabı gerçek Flutter ekranı

Birim testi ve widget testi

- Birim testi hesaplama fonksiyonunu doğrudan çağırır; hızlıdır ve hata kaynağını daraltır.
- Widget testi form alanına girilen verinin buton işlemiyle hesaplamaya ulaştığını ve sonucun ekranda gösterildiğini doğrular.
- Golden testi Flutter'ın render ettiği ekranı PNG olarak kaydeder; taşma, eksik metin ve görsel düzen sorunlarını görsel olarak denetlemeye yardım eder.

Tek bir test türü diğerlerinin yerine geçmez.

Hata ayıklama adımları

- Hatanın tekrar üretilbildiği en küçük girdiyi bulun.
- Beklenen ve gerçek sonucu açıkça yazın.
- Arayüzden bağımsız hesaplama fonksiyonunu doğrudan test edin.

4. Ara deęerleri debugger veya kontrollü loglarla inceleyin.
5. Hatanın nedenini düzeltin; yalnız görünen sonucu yamamayın.
6. Hatanın tekrarını önleyen testi kalıcı bırakın.
7. Normal, sınır ve geçersiz deęerleri yeniden çalıştırın.

Üç uygulama görevi

Kolay — KDV hesaplayıcı

Net fiyat ve KDV oranından brüt tutarı hesaplayın. 0 oranı, ondalıklı fiyat, negatif fiyat ve 100'den büyük oran için test yazın.

Orta — Devamsızlık durumu

Toplam ders saati ve devamsızlıktan yüzde hesaplayın. Tam sınırdaki geçen/kalan öğrenciyi ve sıfır toplam saat hatasını test edin.

İleri — Kademeli elektrik faturası

Farklı tüketim aralıklarında farklı birim fiyatlar uygulayın. Her kademe sınırının bir altını, kendisini ve bir üstünü parametrelili testlerle doğrulayın.

Kontrol listesi

- Hesaplama widget kodundan ayrılmış mı?
- Girdi dönüşümü `tryParse` ile güvenli mi?
- Fonksiyon geçersiz deęerleri açıkça reddediyor mu?
- Normal deęer için beklenen sonuç test edilmiş mi?
- 0, 50 ve 100 gibi sınır deęerleri kapsanmış mı?
- Hata düzeltmesini koruyan kalıcı test var mı?
- Widget testi kullanıcı akışını doğruluyor mu?
- Ekran görüntüsü testten geçen gerçek uygulamaya mı ait?

Geri bildirim ve çözüm erişimi

Öğrenci çözümünü gönderdikten sonra İpucunu göster, hatayı tekrar üretme ve sınır deęer seçme yaklaşımını anlatır; tam fonksiyonu vermez. Örnek çözümü göster, gerçek hesaplama, doğrulama, birim ve widget testi kodlarını açar. İndirme bağlantıları testleriyle birlikte çalışan iki Flutter projesini sunar.

Kaynak projeler

- Konu uygulaması: 08 - Cihaz Özellikleri ve Kalite/Uygulamalar/Uygulama-32-Hatalı Hesap Makinesi
- Özgün uygulama: 08 - Cihaz Özellikleri ve Kalite/Uygulamalar/Ozgun-32-Testli Not Hesap