

FLUTTER

HTTP ve JSON

26. hafta • İnternet ve Uzak Veri

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
26	HTTP ve JSON	Kullanıcı Listesi • Döviz Bilgi Kartı

26. Hafta — HTTP ve JSON

Bu hafta ne yapacağız?

Bir REST API'ye gerçek GET isteği gönderecek, HTTP durum kodunu kontrol edecek, JSON yanıtını Dart modellerine dönüştürecek ve yükleniyor–başarı–hata ekranlarını yöneteceğiz. Konu uygulaması uzak kullanıcı listesini; özgün uygulama günlük döviz referans kurlarını gösterecek.

Kazanımlar

- URI ve HTTP GET isteği oluşturma
- Durum kodu 200 dışındaki yanıtları hata olarak ele alma
- İsteğe zaman aşımı ekleme
- JSON nesnesi, liste ve iç içe alanları çözümleme
- Ağ verisini tür güvenli Dart modeline dönüştürme
- Servise `http.Client` enjekte ederek kontrollü test yazma
- Hata sonrasında yeniden deneme ve yenileme davranışı oluşturma

Kurulum

```
dependencies:  
  flutter:  
    sdk: flutter  
  http: ^1.6.0
```

Çözümlü uygulama — HTTP Kullanıcı Listesi

1. Kullanıcı modeli

```
class Kullanici {  
  const Kullanici({  
    required this.id,  
    required this.ad,  
    required this.eposta,  
    required this.sehir,  
  });  
  
  final int id;  
  final String ad;  
  final String eposta;  
  final String sehir;  
  
  factory Kullanici.fromJson(Map<String, dynamic> json) => Kullanici(  
    id: json['id'] as int,  
    ad: json['name'] as String,  
    eposta: json['email'] as String,  
    sehir: (json['address'] as Map<String, dynamic>)['city'] as String,  
  );  
}
```

`address.city` iç içe bir JSON alanıdır. Önce `address` nesnesi Map'e dönüştürülür, sonra `city` okunur.

2. Gerçek GET isteği

```
final uri = Uri.parse('https://jsonplaceholder.typicode.com/users');

final yanıt = await client
  .get(uri)
  .timeout(const Duration(seconds: 8));
```

`Uri.parse`, adresi güvenli URI nesnesine dönüştürür. `timeout`, bağlantı hiç tamamlanmazsa arayüzün sonsuza kadar beklemesini engeller.

3. Durum kodunu denetleme

```
if (yanıt.statusCode != 200) {
  throw http.ClientException(
    'HTTP ${yanıt.statusCode}',
    uri,
  );
}
```

Sunucunun yanıt vermesi işlemin başarılı olduğu anlamına gelmez. Örneğin 404 ve 500 yanıtları da bir HTTP yanıtıdır fakat başarı verisi olarak çözümlenmemelidir.

4. JSON listesini modellere dönüştürme

```
final data = jsonDecode(yanıt.body) as List;

return data
  .map((e) => Kullanici.fromJson(
    Map<String, dynamic>.from(e as Map),
  ))
  .toList();
```

Test edilebilir servis

Servis kendi içinde doğrudan yeni istemci üretmek yerine `http.Client` alır:

```
class KullaniciServisi {
  KullaniciServisi(this.client);
  final http.Client client;
}
```

Canlı uygulama `http.Client()` kullanır. Test ise önce 500, ikinci istekte kontrollü JSON döndüren `MockClient` verir. Böylece hata ve yeniden deneme akışı gerçek internet durumundan bağımsız doğrulanır.

Gerçek uygulama ekranı

Test ilk HTTP 500 yanıtında hata ekranını, “Yeniden dene” sonrasında üç kullanıcının JSON'dan çözümlenmesini doğrulamıştır.

3 kullanıcı JSON'dan çözümlendi

- 1** Ece Yılmaz
ece@okul.edu.tr
Ankara >
- 2** Mert Kaya
mert@okul.edu.tr
İzmir >
- 3** Zeynep Ak
zeynep@okul.edu.tr
Bursa >

HTTP Kullanıcı Listesi gerçek Flutter ekranı

Özgün uygulama — HTTP Döviz Bilgi Kartı

Canlı uygulama anahtarsız Frankfurter API'nin güncel kur uç noktasını kullanır:

```
final uri = Uri.parse(  
  'https://api.frankfurter.dev/v1/latest'  
  '?base=TRY&symbols=USD,EUR,GBP',  
);
```

Frankfurter'ın resmi v1 belgesine göre latest uç noktası son iş gününün kurlarını verir ve base parametresi temel para birimini değiştirir: <https://frankfurter.dev/v1/>

JSON yanıtı

```
{
  "base": "TRY",
  "date": "2026-07-21",
  "rates": {
    "USD": 0.030,
    "EUR": 0.027,
    "GBP": 0.023
  }
}
```

Bu değerler 1 TL'nin kaç birim yabancı para olduğunu gösterir. Ekranda 1 USD'nin TL karşılığını göstermek için ters oran alınır:

```
final yabancı = (rates[kod] as num).toDouble();
if (yabancı <= 0) throw const FormatException('Geçersiz kur');
final tlkarşılığı = 1 / yabancı;
```

Finansal uygulamalarda kaynak, referans tarihi ve verinin banka alıř/satıř fiyatı olmadıęı açıkça belirtilmelidir.

HTTP Döviz Bilgi Kartı



Türk lirası karşılıkları

1 birim döviz kaç TL?

Referans tarihi: 2026-07-21

U

1 USD

Frankfurter günlük referans
kuru

₺33.33

E

1 EUR

Frankfurter günlük referans
kuru

₺37.04

G

1 GBP

Frankfurter günlük referans
kuru

₺43.48

Bilgilendirme: Bu değerler eğitim amaçlı referans kurlardır;
banka alıř/satıř fiyatı değildir.

HTTP Döviz Bilgi Kartı gerçek Flutter ekranı

Hata türleri

- Ağ yok veya DNS hatası: istemci exception üretir.
- Zaman aşımı: `TimeoutException` oluşur.
- HTTP 4xx/5xx: durum kodu elle reddedilir.
- Bozuk JSON: `FormatException` oluşur.
- Beklenmeyen/eksik alan: tür dönüşümü hata verir.
- Boş sonuç: başarılıdır fakat ayrı boş durum ekranı gerektirir.

Öğrenciye teknik stack trace yerine anlaşılır mesaj ve yeniden deneme seçeneği gösterilir; geliştirici için ayrıntı log sistemine yazılır.

Uygulama görevleri

Kolay — Tek kullanıcı kartı

Bir kullanıcı uç noktasından ad, e-posta ve şehir alın. Yükleniyor ve hata ekranı ekleyin.

Orta — Ülke bilgisi

Açık bir ülkeler API'sinden ülke adı, başkent ve nüfus çekin. Boş veya eksik alanları güvenli yönetin.

İleri — Kur dönüştürücü

Kaynak ve hedef para birimini seçtin, API'den oran alın ve girilen tutarı dönüştürün. Zaman aşımı, geçersiz para kodu, yenileme ve son güncelleme tarihi ekleyin.

Test ve kontrol

- Test istemcisi istenen URI'yi doğrulamalı.
- 200 yanıtı modellere dönüşmeli.
- 4xx/5xx yanıtı başarı ekranına ulaşmamalı.
- Yeniden deneme yeni HTTP isteği göndermeli.
- İç içe JSON alanları doğru okunmalı.
- Sıfır veya eksik kur bölme hatasına yol açmamalı.
- Uygulama yükleniyor, boş, hata ve dolu durumlarını ayrı göstermeli.

Geri bildirim ve çözüm erişimi

Öğrenci HTTP servis ve model kodunu gönderdikten sonra İpucunu göster istek–durum kodu–JSON akışını açıklar. Örnek çözümü göster gerçek servis, model ve arayüz kaynaklarını; indirme bağlantısı ise HTTP bağımlılığıyla çalışan projeyi sunar.

Kaynak projeler

- Konu uygulaması: 07 - ■nternet ve Uzak Veri/Uygulamalar/Uygulama-26-Kullan■c■
Listesi
- Özgün uygulama: 07 - ■nternet ve Uzak Veri/Uygulamalar/Ozgun-26-Döviz Bilgi Kart■