

FLUTTER

Widget Ağacı ve MaterialApp

2. hafta • Flutter'a Başlangıç

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
2	Widget Ağacı ve MaterialApp	Tanıtım Ekranı • Günlük Hedef

2. Hafta - Widget Ağacı ve MaterialApp

Uygulama hedefi

Bu derste Tanıtım Ekranı uygulamasını sıfırdan kuracağız. Ekranı tek parça olarak düşünmek yerine kök, sayfa, yerleşim ve içerik widget'larına ayıracağız. Ders sonunda öğrenci gördüğü bir ekranı widget ağacı biçiminde okuyabilecek.

Kazanımlar

- runApp ile kök widget arasındaki ilişkiyi açıkla.
- MaterialApp, Scaffold, AppBar ve body görevlerini ayırt eder.
- İçeriği ListView, Card, Column, Wrap ve Chip ile düzenler.
- Sabit ekran için StatelessWidget kullanır.
- Uzun içeriğin küçük ekranda taşmasını ListView ile önler.

Widget ağacını okumak

Tanıtım Ekranı'nın sadeleştirilmiş ağacı şöyledir:



Girinti arttıkça widget'ın başka bir widget'ın çocuğu olduğu anlaşılır. MaterialApp uygulama düzeyindeki tema ve temel davranışları, Scaffold ise tek sayfanın görsel çatısını yönetir.

1. Uygulamayı başlatma

```

void main() => runApp(const TanitimApp());

class TanitimApp extends StatelessWidget {
  const TanitimApp({super.key});

  @override
  Widget build(BuildContext context) => MaterialApp(
    debugShowCheckedModeBanner: false,
    theme: ThemeData(
      colorScheme: ColorScheme.fromSeed(
        seedColor: const Color(0xff00695c),
      ),
      useMaterial3: true,
    ),
    home: const TanitimPage(),
  );
}

```

Bu ekranda kullanıcı tarafından değiştirilen bir veri olmadığı için StatelessWidget yeterlidir.

2. Kaydırılabilir sayfa oluşturma

```

Scaffold(
  appBar: AppBar(title: const Text('Tanıtım Ekranı')),
  body: ListView(
    padding: const EdgeInsets.all(22),
    children: const [
      CircleAvatar(
        radius: 52,
        child: Icon(Icons.person, size: 62),
      ),
      SizedBox(height: 14),
      Text('Ece Yılmaz', textAlign: TextAlign.center),
    ],
  ),
)

```

ListView, ekran yüksekliği yetersiz olduğunda içeriği kaydırır. Sabit bir Column kullanılsaydı küçük ekranda sarı-siyah taşma uyarısı oluşabilirdi.

3. Beceri etiketlerini yerleştirme

```

const Wrap(
  spacing: 8,
  runSpacing: 8,
  children: [
    Chip(avatar: Icon(Icons.flutter_dash), label: Text('Flutter')),
    Chip(avatar: Icon(Icons.code), label: Text('Dart')),
    Chip(avatar: Icon(Icons.palette), label: Text('Tasarım')),
  ],
)

```

Wrap, yatay alan bittiğinde sonraki Chip'i yeni satıra taşır. Bu nedenle farklı telefon genişliklerinde daha güvenlidir.

Çalışan uygulama

Uygulama gerçek Flutter widget testinde 430 × 760 ekran boyutunda çalıştırıldı. Test; Hakkımda bölümünü, Flutter beceri etiketini ve widget ağacının oluşturulmasını doğruladı.

Tanıtım Ekranı



Ece Yılmaz

11. Sınıf Bilişim Teknolojileri

Hakkımda

Mobil uygulamalar geliştiriyor, okul projelerinde tasarım ve kodlama görevleri alıyorum.



Flutter



Dart



Tasarım



İletişim: ece@okul.edu.tr

Tanıtım Ekranı çalışan Flutter uygulaması

Proje klasörü: Uygulamalar/Uygulama-02-Tanıtım Ekranı

Uygulama çalışması

- Öğrenci adını ve sınıf bilgisini değiştirin.
- En az dört beceri Chip'i ekleyin.
- Wrap yerine Row kullanıp dar ekranda oluşan farkı gözlemleyin.
- Hakkımda kartını ayrı bir widget sınıfına çıkarın.
- Ekranı 320 ve 430 piksel genişliklerde test edin.

Özgün proje - Günlük Hedef

Bu projede widget ağacına durum yönetimi eklenir. Üç günlük hedef Checkbox ile işaretlenir; tamamlanan hedef sayısı ve LinearProgressIndicator değeri aynı listeden hesaplanır. "Tümünü tamamla" düğmesi bütün hedefleri gerçek zamanlı günceller.

Günlük Hedeflerim

3 / 3 hedef tamamlandı



20 dakika kitap oku



Flutter dersini tekrar et



30 dakika yürüyüş yap



Tümünü tamamla

Günlük Hedef çalışan özgün Flutter uygulaması

Öğrenci görevi

En az üç hedef içeren kendi takip ekranınızı oluşturun. Checkbox değiştiğinde sayaç ve ilerleme çubuğu birlikte güncellenmelidir. Çözümünüzü gönderdikten sonra sistemde Günlük Hedef gerçek Dart çözümü ve indirilebilir çalışan proje açılacaktır.

Geri bildirim ölçütleri

- Hedef metni ile tamamlanma durumu aynı sıra numarasıyla eşleşiyor.
- Tamamlanan sayı listeden hesaplanıyor; elle sabitlenmiyor.
- Sıfıra bölme ihtimali engelleniyor.
- Checkbox ve toplu tamamlama işlemleri setState içinde çalışıyor.
- Son görüntü checkbox animasyonu tamamlandıktan sonra alınıyor.

Ders sonu kontrolü

- Widget ağacını yukarıdan aşağı okuyabiliyorum.
- MaterialApp ve Scaffold görevlerini ayırt edebiliyorum.
- Dar ekranda ListView ve Wrap kullanma nedenini açıklayabiliyorum.
- Stateless ve Stateful ekran seçimini gerekçelendirebiliyorum.
- Çalışan Flutter uygulamasını test ederek doğrulayabiliyorum.