MOBİL UYGULAMALAR
(Dart & Flutter)

HAFTA 1 : MOBİL UYGULAMALAR VE FLUTTER

Ayhan AKKAYA

Flutter Developer and Trainer

Bilgisayar Yüksek Mühendisi

akkayasoft

Mobil Uygulama Ekosistemi

- Mobil uygulamalar, akıllı telefonlar, tabletler ve akıllı saatler gibi küçük, taşınabilir ve kablosuz cihazlar için özel üretilmiş yazılımlardır. Mobil uygulama geliştirme süreci bu cihazlara özgü uygulamaların yazılması, denenmesi ve iyileştirilmesini kapsar.
- Diğer uygulama geliştirme süreçlerinde olduğu gibi, mobil uygulama geliştirme de temel yazılım geliştirme süreçlerinden ve mekaniklerinden beslenir. Bununla birlikte, mobil uygulama geliştirme sürecinde bu cihazların avantajları ve dezavantajları dikkate alınır.

Mobil Uygulama Ekosistemi

- Örnek vermek gerekirse, masaüstü bilgisayarların işlem güçleri çoğunlukla mobil cihazlara oranla çok daha yüksektir. Bunun en önemli nedeni mobil cihazlardaki işlemcilerin kısıtlı batarya kapasitesini etkin kullanma amacıdır.

- Bununla birlikte, masaüstü bilgisayarlar genellikle mobil cihazlardaki kablosuz bağlantı özellikleri ya da parmak izi okuyucu ve kamera gibi yerleşik algılayıcılara sahip değildir. Bu nedenle mobil uygulama geliştiricilerin yüksek işlemci gücü ve pil tüketiminden uzak durmaları gerekir. Öte yandan, mobil cihazların yetenekleri sayesinde mobil bir uygulamada kullanıcı yetkilendirmesi parmak izi okuma ya da yüz tanıma yöntemleri ile yapılabilir.

Sektör Eğilimleri

- Mobil uygulamalar genellikle hedeflenen işletim sistemlerine özgü paketler (Ör: apk, app) haline getirilerek dağıtılır. Bu işlem, mobil işletim sistemlerine özgü uygulama mağazaları üzerinden gerçekleştirilir.
- Günümüzde mobil uygulama sektöründe baskın iki işletim sistemi ve buna bağlı olarak da iki uygulama geliştirme mağazası bulunmaktadır.
- Güncel verilere göre mobil uygulama piyasasının lideri yaklaşık %72'lik pazar payı ile Google firmasının Android işletim sistemi ve bu işletim sistemine özgü uygulama mağazası Play Store'dur.

Sektör Eğilimleri

- Bu sektörün ikinci büyük temsilcisi ise yaklaşık %26'ık pazar payı ile Apple firmasının iOS işletim sistemi ve App Store uygulama mağazasıdır.

- Bu iki firma dışında büyük firmalarca (ör: Microsoft, Samsung) pek çok girişim başlatılsa da bu girişimler piyasadan beklediği ilgiyi görememiştir. Kalan %2'lik pazar payının bu girişimler arasında dağıldığı düşünülebilir.

Mobil Uygulama Geliştirme Yaklaşımları

- **Yerel (Native) Uygulamalar:** Yerel uygulama olarak ifade edilebilecek bu yaklaşımda mobil uygulama hedeflenen platform özgü araçlar ve mekaniklerle tasarlanır. Örneğin Apple'ın iPhone marka telefonlarında koşturulması beklenen bir uygulama, XCode yazılım geliştirme platformunda Objective-C/Swift dili ile hazırlanarak Apple Store uygulama mağazasından yayınlanır.
- Benzer şekilde, Google'ın Android işletim sistemini hedefleyen bir uygulamanın ise Java dilinde Android Studio ile hazırlanması ve Google Play Store üzerinden yayınlanması beklenir.

Mobil Uygulama Geliştirme Yaklaşımları

- **Web Uygulamaları:** Bu uygulamalar HTML5 ve çevresindeki teknolojiler (Ör: CSS ve JS) kullanılarak geliştirilir. Özünde bu uygulamalar mobil uygulama gibi görünen, bazı mobil uygulama yetenekleri kazandırılmış web siteleridir. Kullanıcıların bu sitelere giderek uygulamaları indirmeyi kabul etmeleri beklenir. İndirme işleminden sonra bu uygulama kullanıcının ana ekranına bir ikon olarak yerleştirilir. Web teknolojilerindeki gelişmeler paralelinde bu uygulamalar gün geçtikçe daha çok mobil uygulama görünümü kazanmaktadır.

Mobil Uygulama Geliştirme Yaklaşımları

- **Hibrit Uygulamalar:** Güncel mobil cihaz pazarı düşünüldüğünde farklı donanım kapasiteleri ve özelliklerinde binlerce Android ve iOS cihaz bulunmaktadır. Her iki pazarın da ciddi kâr kapasiteleri bulunduğundan, geliştiriciler genellikle tüm cihazlara erişme eğilimindedir. Bu durum mobil uygulama geliştirme sürecine ciddi bir zaman ve maliyet yükü getirmektedir. Öncelikle, Android gibi hedeflenen bir platformun pek çok versiyonu bulunmaktadır. Bunun yanında geliştiriciler her sene yeni bir versiyon ekleme eğilimindedir. Ayrıca uygulamanın çok çeşitli donanım özelliklerinde çalışabilir olması beklenir.

Flutter Nedir?

- Flutter, tek bir programlama dilinde yazılan kodlarla, farklı mobil platformlarda (Ör: iOS ve Android) çalışabilen yerel uygulamalar üretebilen bir araçtır. Bu kısa tanımda oldukça önemli ifadeler ve teknolojilerden bahsedilmektedir
- Öncelikle, Flutter projelerinde hedeflenen platformlara özgü yerel uygulamalar üretilir. Bu uygulamaların koşturulması (çalıştırılması) için çevirici/ara uygulamalara (Ör: AppInventor Companion, Webview) gerek yoktur. Flutter uygulamaları doğrudan işletim sistemi tarafından çalıştırıldığından diğer hibrit uygulama geliştirme alternatiflerine oranla çok daha hızlıdırlar.

Flutter Nedir?

- Phonegap gibi alternatiflerde yazılan kodlar aradaki bir katman tarafından (Ör: Webview), üzerinde çalışılan işlemcinin anlayabileceği makine kodlarına çevrilir. Bu ara çevirme nedeniyle uygulamalar performans kaybı yaşar.

- Buna karşın Flutter uygulamaları doğrudan üzerinde çalışılan işlemcinin anlayacağı makine kodlarına çevrilmiş olarak dağıtılır. Bu sayede ara çevirme işleminden kaynaklanan performans kaybından etkilenmezler.

Flutter Nedir?

- Flutter uygulamalarının üretilmesi için Dart programlama dili kullanılır. Dart kodları ile geliştirilen projelerden hedeflenen platformlara özgün yerel uygulamalar üretilerek dağıtılır.
- Buna karşın, iOS işletim sistemine özgün yerel uygulama üretmek için Objective C ya da Swift dillerinden birini kullanarak XCode ortamında proje üretilmesi gerekir.
- Benzer şekilde Android platformuna özgün yerel uygulama üretmek için ise Java ya da Kotlin programlama dilleri ile Android Studio ortamında proje üretilmesi gerekir. Flutter sayesinde tüm bu araçların kullanılmasına, farklı programlama dilleri ve çalışma mekanizmalarının öğrenilmesine gerek kalmaz.

Flutter Nedir?

- Yazılım Geliştirme Kütüphaneleri Dart dili ile kullanılabilecek kütüphaneleri ve Widget yapılarını sağlar. Dart dilini kullanarak mobil cihaz üzerinde işlem gerçekleştirmek için kütüphaneler kullanılır.
- Örneğin, mobil cihazdaki kamerayı kullanarak görüntü elde edilmesi, bu görüntünün işlenmesi ve cihazdaki disk alanına resim formatında kaydedilmesi için pek çok kütüphaneye başvurulur.
- Bunun yanında mobil uygulamanın kullanıcı arayüzünü oluşturmak için sayfalar, düğmeler ve menüleri oluşturan geniş bir bileşen (Widget) kütüphanesi bulunur.

Flutter Nedir?

- Flutter uygulamaları Dart programlama dili ile yazılır. Dart programlama dilinin amacı kullanıcıların gördüğü önyüzleri (front-end) tasarlamaktır.
- Dart, Google tarafından tasarlanan ve geliştirilmekte olan bir programlama dilidir.
- Dart nesneye yönelik ve sıkı veri tipi kontrollü (strongly-typed) bir dildir. Bu nedenle değişkenlerin veri tipleri ya da fonksiyonların geri döndürdüğü değerler nedeniyle gerçekleşebilecek uyumsuzluklar derleme sırasında yakalanabilmektedir.
- Dart ve Flutter, Google firması tarafından yürütülen iki ayrı projedir.

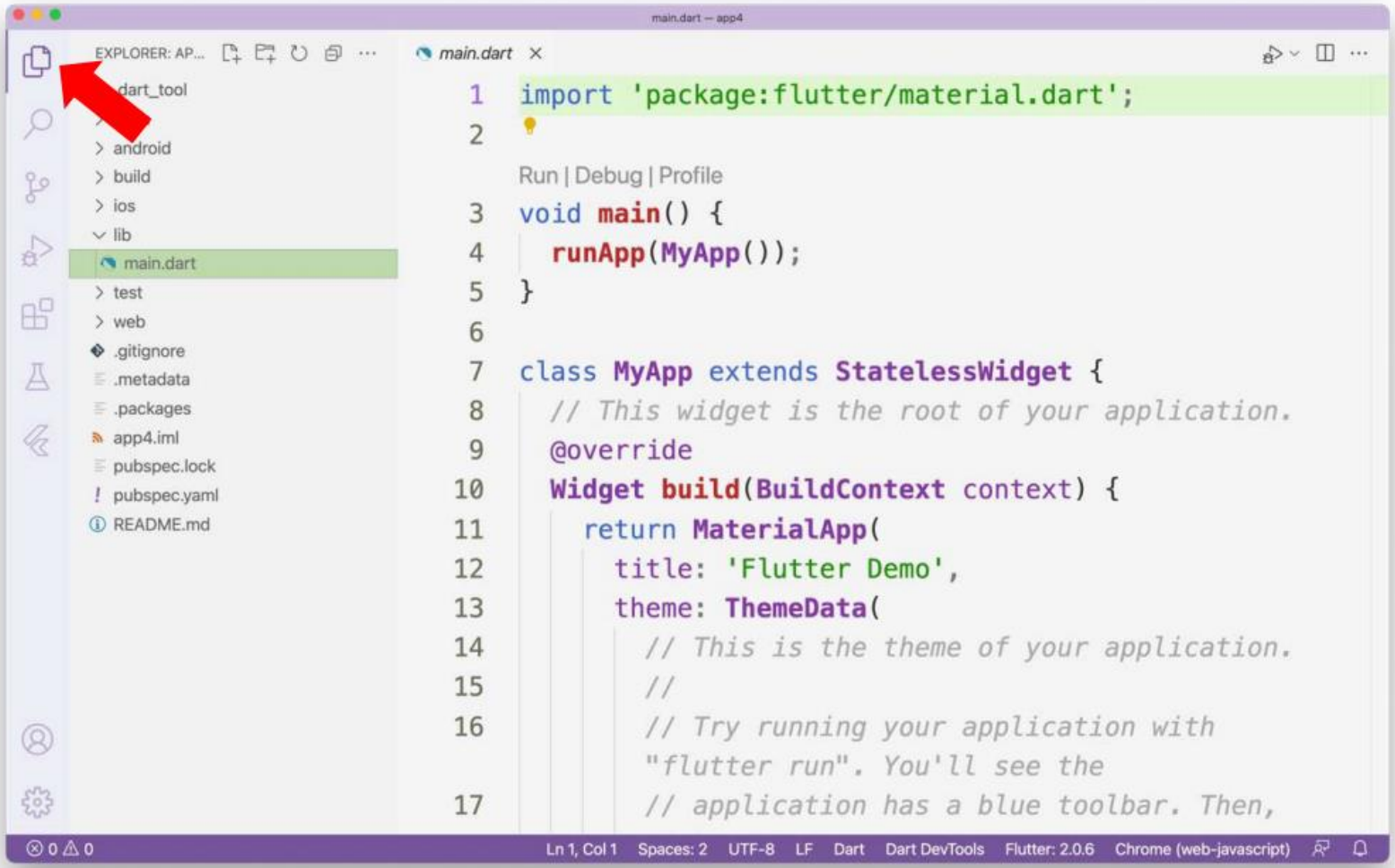
Flutter’ın Hedeflediği Platformlar

- **Android:** Google firması tasafından üretilen Android dünya genelinde en çok kullanılan mobil işletim sistemidir. Güncel verilere göre Android’in pazar payı %72 civarındadır.
- **iOS:** Apple firmasının iPhone model telefonlarında kullanılan işletim sistemidir. Güncel verilere göre iOS’un pazar payı %27 civarındadır.
- **MacOS:** Apple firmasının Mac modeli dizüstü ve masaüstü bilgisayarlarında kullanılan işletim sistemidir. Bu uygulamalar firmanın uygulama mağazası üzerinden dağıtılabilir.

Flutter’ın Hedeflediği Platformlar

- **Linux:** Dünya genelinde kullanılabilen özgür bir işletim sistemidir. Pek çok Linux türevi bulunmaktadır. Türkiye’de geliştirilen Pardus da bir Linux türevidir. Flutter, bu türevler üzerinde çalışabilecek snap paketleri oluşturabilmektedir. Bu paketler masaüstü ve dizüstü bilgisayarlarda kullanılabilir.
- **WebApp:** Temelde bir internet sitesi olarak ele alınabilir. Chrome, Safari, Edge ve Firefox gibi internet tarayıcılarının masaüstü ve mobil sürümleri üzerinde çalıştırılabilen uygulamalardır.

Flutter Uygulama Yapısı



Flutter Uygulamalarındaki Önemli Dosyalar

- **android:** Anroid klasörü bütün bir Android projesi tutar. Flutter yazılım geliştirme kiti, oluşturulan proje ile bu proje içindeki Android projesini birleştirerek bir Android uygulaması üretir. Bu klasör içinde herhangi bir değişiklik yapılmasına gerek yoktur. Oldukça önemli bir klasör olmasına rağmen, bu klasör pasif bir klasör gibi görülmelidir. Yalnızca projenin çıktısının alınacağı durumlarda üretilen APK dosyasına bu klasör içinden ulaşılacaktır.
- **ios:** Android klasörünün iOS işletim sistemine özgün karşılığı olarak düşünülebilir. Bu klasör içinde de bütün bir iOS projesi bulunur. Proje derlenirken yazılan kodlar bu proje ile birleştirilerek sonuç iOS projesi üretilecektir. Bu klasör de pasif bir klasör olarak görülebilir. Apple şirketi kendi araçları dışında iOS uygulamaları üretilmesine izin vermediğinden Windows çalışma ortamında ios klasörü görülemez.

Flutter Uygulamalarındaki Önemli Dosyalar

- **build:** build klasörü Flutter YGK tarafından derlenmiş kodların tutulduğu klasördür. Bu klasör YGK tarafından düzenlenir. Bu klasöre müdahale edilmemelidir.
- **lib:** lib (Library) klasörü Flutter uygulamalarında en çok kullanılan klasördür. Dart kodları bu klasördeki dosyalara eklenir. Flutter projelerinin geliştirilmesi sürecinde en önemli klasör lib klasörüdür.
- **test:** Test klasörü, Flutter projelerini test etmek için üretilen otomatikleştirilmiş rutinleri tutar. Daha çok İleri düzey Flutter programcılarının kullanabileceği bir klasördür.

Flutter Uygulamalarındaki Önemli Dosyalar

- **.gitignore:** Git kaynak kodu yönetim sistemi tarafından kullanılan bir dosyadır. Git, yazılan kodların günlük kopyalarını (snapshot) tutabilen bir versiyonlama sistemidir. Bu sayede kodlarda geriye dönüş gibi olanaklar sağlanır. Flutter uygulaması geliştirme sürecinde bu uygulamanın kullanılması zorunlu olmamakla birlikte, programcıların işlerini kolaylaştıran bir uygulamadır.
- **.metadata:** Flutter uygulamasının geliştirme süreciyle ilgili Flutter tarafından takip edilen üst verileri saklamakla görevli bir dosyadır. Programcıların müdahale etmesine gerek yoktur.
- **.packages:** Bu dosya Flutter projesi tarafından kullanılan kütüphanelerin takip edildiği dosyadır. Flutter YGK tarafından oluşturulur ve düzenlenir. Programcıların müdahale etmesine gerek yoktur.

Flutter Uygulamalarındaki Önemli Dosyalar

- **app1.iml:** IML dosyası projenin ismi ile başlar. Bu senaryodaki projenin adı app1 olduğundan bu dosyanını adı app1.iml olarak belirlenmiştir. Flutter Yazılım Geliştirme Kiti tarafından projenin ayarları ve gereksinimlerinin takip edilmesi amacıyla üretilir ve düzenlenir. Programcıların müdahale etmesine gerek yoktur.
- **pubspec.lock:** Bu dosya aşağıda anlatılacak pubspec.yaml dosyasının içeriğine göre otomatik olarak üretilir. Programcıların müdahale etmesine gerek yoktur.
- **pubspec.yaml:** Flutter projesinin temel ayarları ve projeye eklenecek üçüncü parti kaynakların belirtildiği belgedir. Bu belge yaml dili kullanılarak yapılandırılmıştır. Projede kullanılacak fontlar, ikon paketleri, üçüncü parti kütüphaneler ya da görseller bu belgeye işlenir. Bunun yanında projenin hangi Dart versiyonunda ve hangi paketlerle geliştirildiğine ilişkin ayarlar bu belgede tutulur.
- **README.md:** Projenin diğer geliştiricilerle paylaşılması halinde, proje hakkında bilgi sağlamak amacıyla README.md dosyasının içeriği değiştirilebilir.

main.dart Dosyasının Anatomisi

```
1   import 'package:flutter/material.dart';
2
3   void main() {
4     runApp(MyApp());
5     }
6
7   class MyApp extends StatelessWidget {
8     Widget build(BuildContext context) {
9       return MaterialApp(home: Text("Merhaba!"));
10    }
11  }
```

1.satırda bir import işlevi çağırılmaktadır. Bu import çağrısı ile Flutter tarafından sağlanan pek çok özellik kullanılabilir hale gelecektir.

Örneğin bu uygulamada kullanacağımız StatelessWidget, MaterialApp, Text gibi sınıfların tümü material.dart kütüphanesi üzerinden çekilecektir. Bu sınıflar uygulamamız içinde değil, geliştirme yapılan bilgisayardaki Flutter dosyalarında saklanmaktadır. Bu sınıflara ilişkin kodlar derleme esnasında uygulamaya enjekte edilecektir. Bu satır silindiğinde yukarıda bahsettiğimiz tüm sınıfların altına kırmızı çizgi eklendiği, yani bu sınıflarla ilgili sorun olduğu gözlenecektir. Bunun nedeni az önce bahsedildiği gibi bu sınıflara ilişkin tanımlama ve kodların material.dart kütüphanesi üzerinden projeye çekiliyor oluşudur. import işlemi pubspec dosyasında dependencies bölümünde flutter YGK'nin içerilmesi sayesinde yapılabilmektedir. Bu tanımlama ile Flutter kütüphanesi ve uygulama arasında bağlantı sağlanmaktadır.

main.dart Dosyasının Anatomisi

```
1   import 'package:flutter/material.dart';
2
3   void main() {
4     runApp(MyApp());
5     }
6
7   class MyApp extends StatelessWidget {
8     Widget build(BuildContext context) {
9       return MaterialApp(home: Text("Merhaba!"));
10    }
11  }
```

3. satırda main() fonksiyonu tanımlanmaktadır. Dart, nesne yönelik bir dil olduğundan kodlarının başlangıcının işaretlendiği bir giriş noktasına ihtiyaç vardır. Dart sınıflarında bu giriş noktası main() fonksiyonudur.

3-5. satırlar arasındaki main() fonksiyonu runApp fonksiyonunu çağırmaktadır. Bu fonksiyona 7-11. satırlar arasında tanımlanan MyApp sınıfını parametre olarak göndermektedir. Bu sayede Dart, MyApp sınıfı içinde tanımlı kodları çalıştırması gerektiğini anlamaktadır.

7. satırda bileşen ağacının kökü olan, uygulamanın kendisini tanımlayan MyApp isimli sınıfın tanımlanmasına başlanmaktadır. Bu amaçla class anahtar sözcüğü kullanılır. Ardından sınıfın adı olan MyApp verilmiştir.

9. Satırda oluşturulan MaterialApp bileşenini geri döndürdüğünden bu bileşen ekrana gönderilmiş olur. MaterialApp sınıfının oluşturulmasında çalıştırılan kodlara Text bileşeni gönderildiğinden oluşturduğumuz Merhaba! yazısı ekrana basılacaktır.