

10/7/2016

Mobile Programlama

Yrd. Doç. Dr. Burak İNNER

Kocaeli Üniversitesi

Bilgisayar Mühendisliği

Burak İNNER
[COMPANY NAME]

Mobil Programlama Ders Notları

Android nedir?

Android İşletim Sistemi

Android, akıllı telefonlar ve tablet bilgisayarlar gibi mobil cihazlar için geliştirilmiş açık kaynak kodlu ve Linux çekirdek (kernel) tabanlı bir işletim sistemidir. Android sisteminin geliştirilmesinden sorumlu proje Android Open Source Project (AOSP) olarak adlandırılır ve Google liderliğinde, diğer şirketlerle birlikte Open Handset Alliance tarafından geliştirilmektedir.

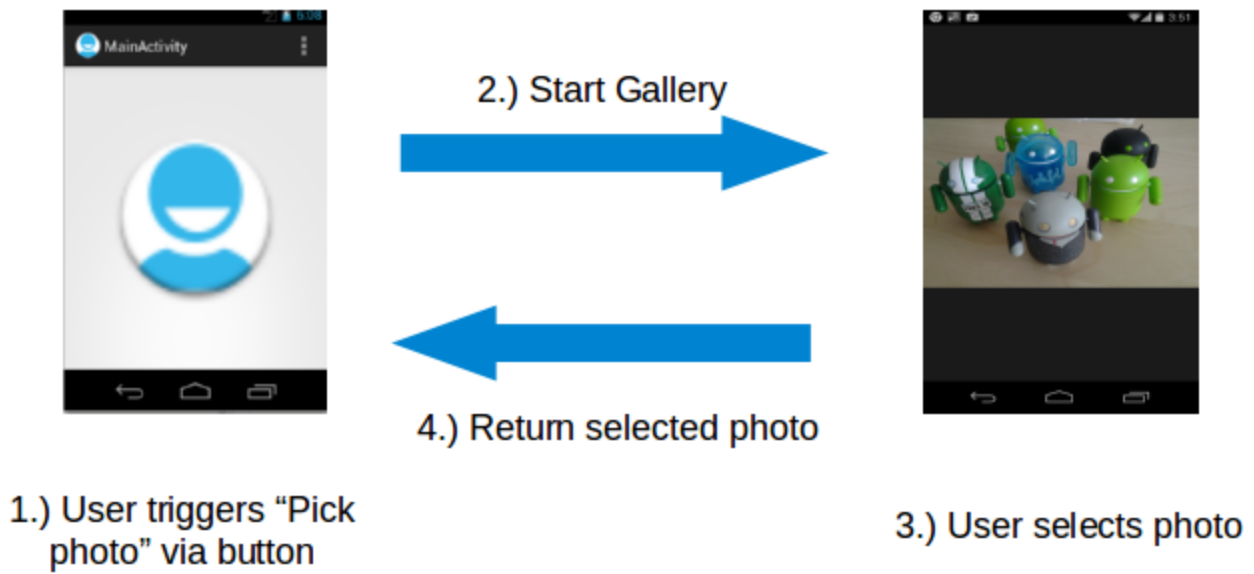
Android sistemi, arkaplan işlemlerini destekler, zengin bir kullanıcı arabirimi kütüphanesi sağlar, OpenGL ES (kısaca OpenGL olarak kullanılır) standardını kullanarak 2-D ve 3-D grafik destekler, dosya sistemine ve gömülü SQLite veritabanına erişim izinlerini denetler.

Bir Android uygulama genellikle farklı görsel ve görsel olmayan bileşenden oluşur ve diğer uygulamaların bileşenlerini yeniden kullanabilirsiniz.

Task

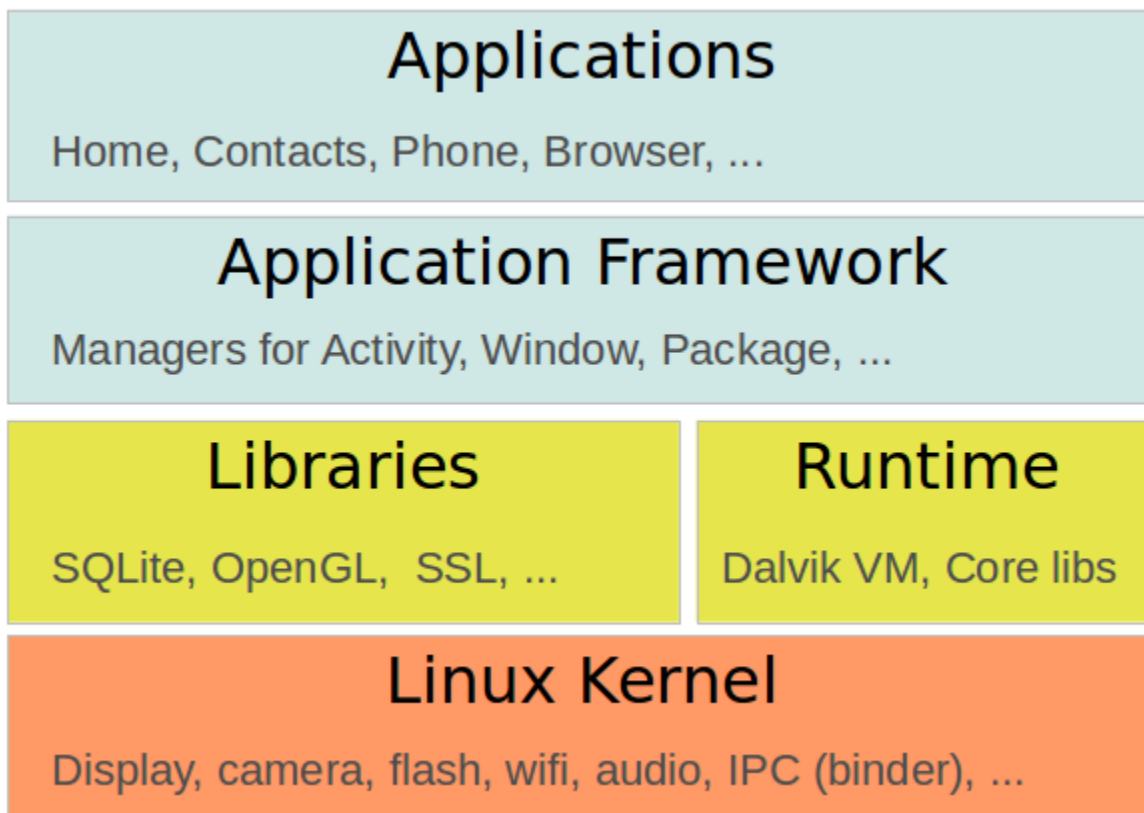
Android'de diğer uygulama bileşenlerinin yeniden kullanılması kavramına task (görev) adı verilir. Bir uygulama bir görevi başarmak için diğer Android bileşenlerine erişebilirler. Örneğin uygulamanızın bir bileşeni Android işletim sisteminin fotoğraf yönetimi yapan (bu bileşen uygulamanın parçası olmasa bile) başka bir bileşeni tetikleyebilir. Bu bileşen sizin bir fotoğraf seçmenizi sağlar ve seçilen fotoğrafı kullanmak üzere uygulamanza geri dönmenizi sağlar.

Böyle bir olayı akış, aşağıdaki grafikte gösterilmiştir.



Android platformu bileşenleri

Android sistem aşağıdaki grafikte de görüldüğü gibi, tipik dört seviyeye (alana) ayrılır.



Bu seviyeler şu şekilde tanımlanabilir:

- Uygulamalar - Android Open Source Project; Tarayıcı (Browser), Kamera, Galeri, Müzik, Telefon vb. gibi birçok varsayılan uygulamayı içerir.
- Uygulama Framework (çerçevesi) - Android uygulamalarından Android sistemi ile üst düzey etkileşimleri sağlayan bir API.
- Kütüphaneler ve Çalışma Zamanı – Uygulama çerçevesi ve Dalvik çalışma zamanının birçok ortak fonksiyonları için kütüphaneler (örneğin: Grafik Render, veri depolama, web tarama, vb.) sunmakla birlikte Android uygulamalarını çalıştırmak için core (çekirdek) Java kütüphanelerini sunar.
- Linux çekirdeği - temel donanım için iletişim katmanı.

Uygulama çerçevesi, Linux çekirdeği, kütüphaneler ve çalışma zamanını kapsamaktadır (encapsulated). Android uygulama geliştiricisi, genellikle yeni Android uygulamaları oluşturmak için üstteki iki seviye (Uygulamalar ve Uygulama Çerçevesi) ile çalışır.

1.4. Google Play

Google şirketi tarafından Google Play hizmeti ile programcıların Android kullanıcılarına kendi Android uygulamalarını sunabileceği bir pazar ortamı hazırlanmıştır. Müşteriler (mobil cihaz kullanıcıları) Google Play uygulamasını kullanarak Android uygulamalarını satın alıp yükleyebilirler.

Google Play ayrıca bir güncelleme hizmeti sunmaktadır. Eğer bir programcı Google Play'e yaptığı uygulamanın yeni sürümünü yüklerse, mevcut kullanıcılara bir güncelleştirmenin yüklenebilir olduğu bildirilir ve güncelleştirmeyi yükleyebilme imkânı sunar.

Google Play, ayrıca Android uygulama geliştiricilerinin Google tarafından sunulan hizmet ve kütüphanelere erişimini sağlar. Örneğin, Google Haritalarının gösterilmesi ve kullanılması, farklı Android yüklemeleri arasındaki uygulama durumunun eşitlemesi için hizmet sunar. Google Play üzerinden bu hizmetlerin sunulması eski Android sürümleri kullananlar için telefonlarında Android sürüm güncelleştirmelerine gerek kalmadan bu hizmetleri kullanabilmelerini sağlar.

Android Geliştirme Araçları

JDK

2.1. Android SDK

Android Yazılım Geliştirme Kiti (Android SDK), Android uygulamaları oluşturmak, derlemek ve paketlemek için gerekli araçları içerir. Bu araçların çoğu komut satırı tabanlıdır. Android uygulamaları geliştirmek için öncelikli yol Java programlama dili kullanmaktır.

2.2. Android debug bridge (adb)

Android SDK, sanal veya gerçek Android cihazlara bağlanarak cihazı yönetmeyi veya uygulama hatalarının ayıklamasının yapılmasını sağlayan Android hata ayıklama köprü (ADB) barındırmaktadır.

2.3. Gradle and the Android plug-in for Gradle

Android inşa sistemi olarak Gradle kullanılır. Android takımı Gradle plugin sayesinde Android projesinin en üst dizininde build.gradle dosyasının hazırlanmasını sağlamaktadır. Bu dosyanın içeriği genellikle aşağıdaki gibi görünmektedir fakat değişik sürümlere göre farklılıklar olabileceğini de dikkate alınız.

```
// Top-level build file where you can add configuration options common to all
// sub-projects/modules.

buildscript {
    repositories {
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:1.5.0'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        jcenter()
    }
}
```

2.4. Android Geliştirici Araçları ve Android Studio

Google Android uygulamaları oluşturmak için Android Studio adında bir bütünleşik geliştirme ortamı sunar. Bu IDE IntelliJ IDE'si temellidir.

Android araçları Android'e özel dosyaların düzenlemesi için özelleştirilmiş editörler sunmaktadır. Android'in çoğu yapılandırma dosyaları XML formatındadır. Dosyanın XML gösterimi ile yapılandırılmış bir kullanıcı arayüzü yardımıyla veri girişini sağlayan gösterim arasında geçiş yapmanızı sağlar.

2.5. Android Çalışma_zamanı (ART)

Android 5.0 ve üst sürümlerde tüm Android uygulamaları için çalışma zamanı olarak Android Runtime (ART) kullanılır.

ART (Ahead of Time) zamanın önünde derleme kullanır. Android cihaz üzerine bir uygulama yüklendiğinde uygulama kodu makina koduna çevrilir. Bu da yaklaşık olarak % 30 daha büyük derlenmiş kod üretir fakat uygulamanın başlatılmasından itibaren daha hızlı çalıştırılmasını sağlar. Derlenme işleminin bir kere yapılması sayesinde pilin daha uzun süre gitmesi de sağlamaktadır.

Derleme yalnızca uygulamanın ilk başlatma sırasında bir kez yapıldığı gibi bu da, pil ömrünü kaydeder.

Dex2oat aracı, Android araçları tarafından oluşturulan dex dosyasını alır, çalıştırılabilir ve bağlanabilir (Linkable) (ELF dosyası) formatına çevirir. Bu dosya dex kodu, derlenmiş yerel kod ve meta-veri içerir.

ART içindeki çöp toplama uygulamaların donma sürelerini azaltmak için optimize edilmiştir.

2.6. Android uygulamaları geliştirmek için nasıl

Android uygulamaları öncelikle Java programlama dilinde yazılmıştır. Gelişimi sırasında geliştiriciler Android özel yapılandırma dosyaları oluşturdular ve uygulama mantığını Java programlama dilinde yazdılar. Android araçları arkaplanda bu uygulama dosyalarını Android uygulamasına çevirmektedir. Geliştiriciler IDE içinde dağıtma (deployment) işlemini başlattığında Android uygulaması derlenir, paketlenir, dağıtılır ve başlatılır.

2.7. Android uygulama kaynak kodu Dönüşüm süreci

Java kaynak dosyaları Java derleyicisi tarafından Java sınıf dosyalarına dönüştürülür.

Android SDK içinde Java sınıf dosyalarını dex (Dalvik Executable) dosyasına dönüştüren dx ismindeki bir araç bulunmaktadır. Uygulamanın tüm sınıf dosyaları bu dex dosyasının içine konur. Bu dönüşüm sürecinde sınıf dosyalarındaki gereksiz bilgiler dex dosyasına optimize edilmiş olarak kaydedilir. Örneğin aynı string ifade farklı sınıf dosyalarında bulunursa dex dosyası içinde bu string ifadesine sadece bir referans tutulur. Bu nedenle dex dosyaların boyutu ilgili sınıf dosyalarından daha küçüktür. dex dosyası ve Android projesinin örneğin görüntü ve XML dosyaları gibi kaynakları (resources) bir .apk (Android Package) dosyası olarak paketlenir. AAPT (Android Asset Packaging Tool) programı bu adımı gerçekleştirir. Ortaya çıkan .apk dosyası Android uygulamayı çalıştırmak için gerekli tüm verileri içerir ve adb aracı ile bir Android cihaza dağıtılabilir (deploy).

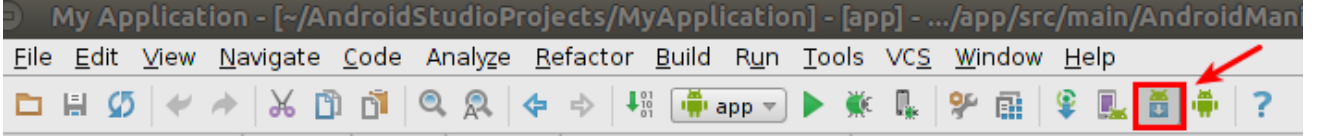
4. Android SDK Yöneticisi

4.1. Android SDK yöneticisinin kullanımı

Android SDK yöneticisi Android API'sinin belirli sürümlerini yüklemenizi sağlar. Android SDK yöneticisi Android paketlerini yüklemenizi ve silmenizi sağlar.

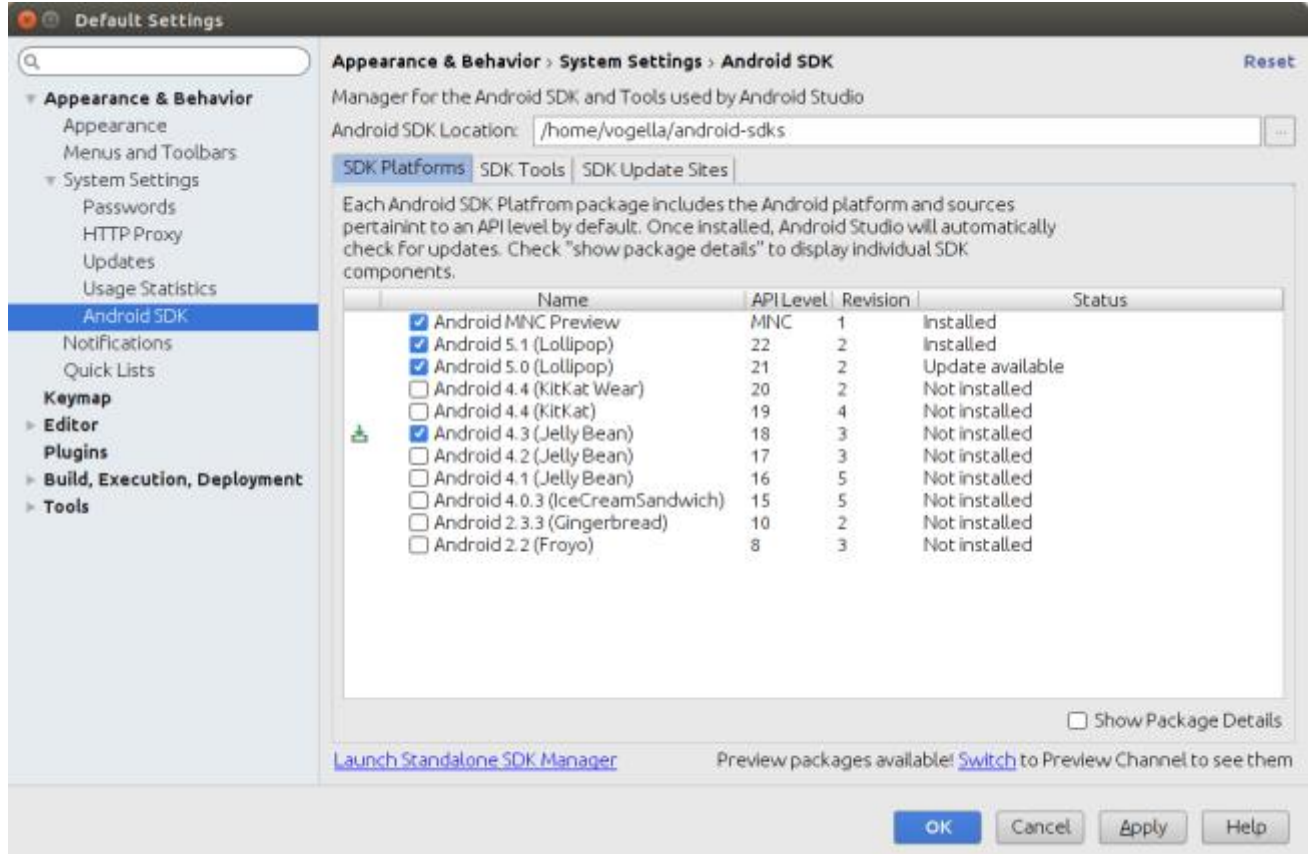
4.2. Android Studio Android SDK yöneticisini açın

Tools → Android → SDK Manager menüsünden veya resimdeki görüldüğü gibi Android Studio araç çubuğundaki SDK Manager simgesine tıklanarak Android yöneticisini açın.



4.3. Seçilen Android sürümü veya kütüphaneleri yükleyin

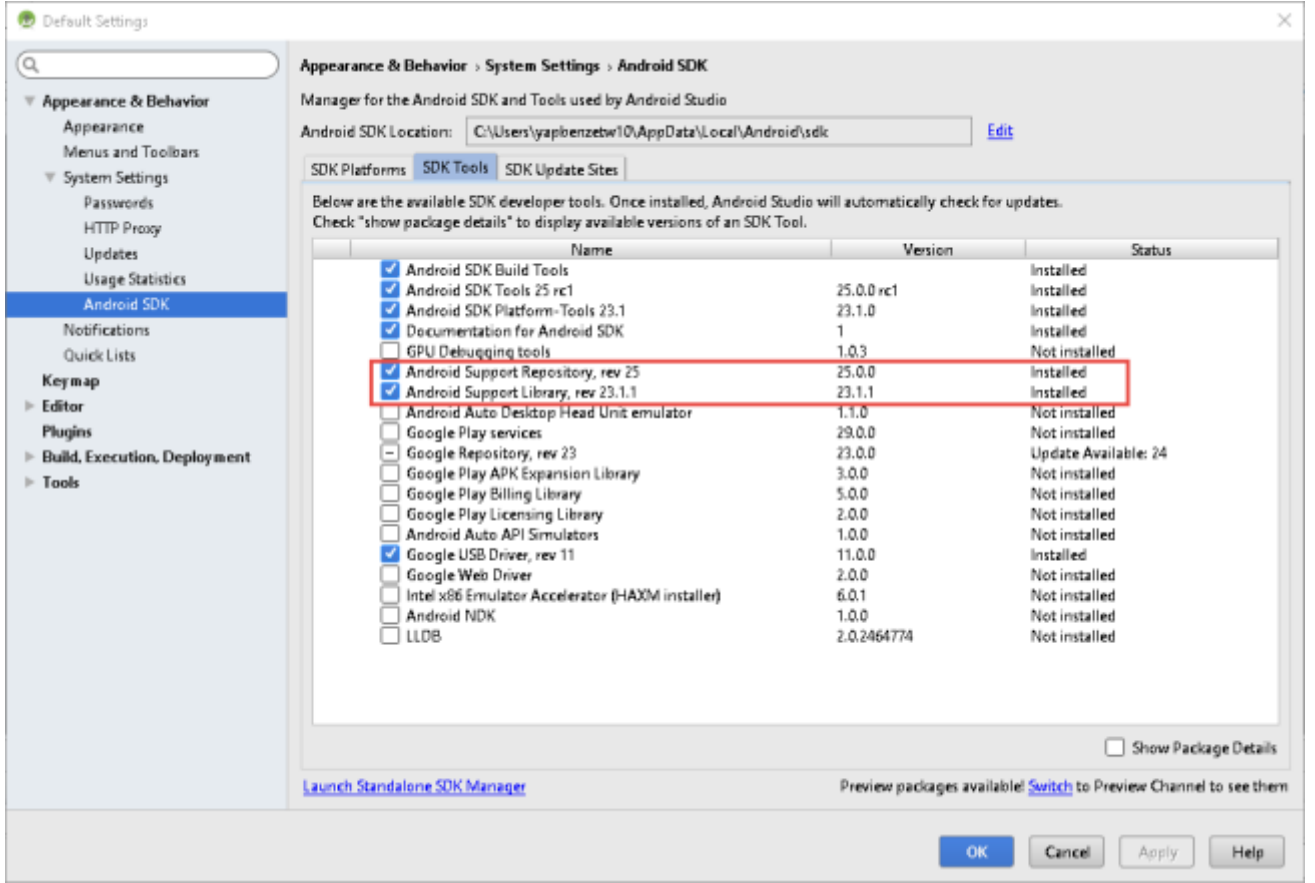
Android SDK yöneticisi açıldığında geliştirme yapmak istediğiniz Android sürümünü seçerek yükleyin düğmesine basın. Aşağıdaki ekranda Android API 18 sürümü için seçimi gösterilmiştir.



Kurulumu başlatmak için OK düğmesine basın. Kurulum tamamlandıktan sonra seçme imkanı sunulmaktadır. SDK Platformları sekmesi kullanılan API sürümlerine göre SDK geliştirme araçları yüklenmesi için kullanılır.

4.4. Destek kütüphanesinin kurulumu

Destek kütüphanesi daha üst Android sürümleri tarafından sağlanan işlevselliği alt Android sürümlerinde kullanmayı sağlar. Android SDK Yöneticisinde "Android Support Repository" seçerek yükleyiniz. "Android Support Library" ise Eclipse ADT aracı içindir.



Bu kütüphanenin v4, v7 ve V13, v15,..., v25 gibi birçok farklı sürümleri vardır. Örneğin, destek kitaplığı v7, API 7 sürümüne sahip cihazlarda çalışır. Destek kütüphanesinin yüksek sürümleri daha düşük sürümlerinin de çalışmasını gerektirir. Örneğin destek kütüphanesinin v7 sürümü v4 kütüphanesini gerektirir.

5. Android geliştirmesi için Eclipse kullanımı

Google geliştirme ekibi gelecekteki tüm geliştirme işlemlerini Android Studio üzerine odaklamıştır. Bu nedenle bu şu anda Android geliştirilmesi için en iyi geliştirme ortamı Android Studio'dur. Şu anda ADT geliştirme aracı yeni Gradle sistemi yerine özel bir Eclipse build sistemi kullanmaktadır. Bu nedenle build sistemi tutarsızlıklara neden olabilmektedir.

6. Test için Android Sanal Cihazları veya Gerçek Cihazların kullanılması

6.1. Android emülatörü ve Android Sanal Cihazı

Android geliştirme araçları içinde bir Android cihaz emülatörü bulunmaktadır. Bu Android Sanal Aygıtı (AVD) gerçek bir Android telefonunu taklit ederek bilgisayarda çalıştırılmasını sağlar. AVD gerçek donanımlara ihtiyaç olmadan farklı Android sürümleri ve yapılandırmaları üzerinde uygulamaların test edilmesini sağlar. Eğer gerçek bir Android cihazınız olsa bile, AVD oluşturulması ve kullanımına aşina olmalısınız.

AVD oluşturma sırasında sanal aygıt için yapılandırmanın hazırlanması gerekir. Örneğin çözünürlük, Android API sürümü ve ekran yoğunluğunu içeren yapılandırmalarının hazırlanması gerekmektedir.

Farklı konfigürasyona sahip birden çok AVD tanımlanabilir ve aynı anda çalıştırılabilir. Bu bir seferde birden farklı aygıt yapılandırmalarını test etmenizi sağlar.

NOT: Eğer başlatma işlemi sırasında AVD'yi durdursanız AVD'nin bozulmasına neden olabilirsiniz. AVD ilk çalıştırılması özellikle eski bir makinede 10 dakika kadar sürebilir. Modern bir makinede genellikle yeni bir AVD başlatmak 1 ile 3 dakika arasında sürer.

AVD başladıktan sonra, GUI fare ile kontrol edebilmektedir. Emülatör ayrıca sağ tarafta bulunan bir menü üzerinden telefon düğmelerine erişim imkanı sunar.

Geliştirme sırasında bir kere AVD'yi başlattıktan sonra durdurmak zorunda değilsiniz. Eğer uygulamayı değiştirdiyseniz ve yeni bir sürümünü test etmek istiyorsanız, sadece AVD üzerine uygulamayı yeniden dağıtmanız (deploy) yeterlidir.

6.2. Debug sertifikası ve geçerlilik süresi

Android cihaz üzerine bir uygulamanın yüklenebilmesi için bu uygulamanın imzalanmış olması gerekir. Geliştirme aşamasında Eclipse hata ayıklama anahtarı olarak adlandırılan bir sertifika ile otomatik olarak uygulamayı imzalar. Bu hata ayıklama sertifikası üretim tarihinden itibaren 365 gün sonra kullanım dışı kalır. Sertifika süresi dolduğunda, derlenme sırasında sertifikanın süresinin dolduğuna dair bir hata alırsınız. Bu sorunu gidermek için debug.keystore dosyasını silmeniz yeterlidir. Varsayılan depolama konumu OS X ve Linux için `~ /.android`, Windows XP için `C:\Documents and Settings\[username]\.android\` Windows Visata ve Windows 7 için `C:\[username]\.android\` klasörleridir. Bir sonraki derleme işlemi sırasında hata ayıklama anahtarı yeniden üretilecektir.

6.3. Android Cihaz emulator Kısayolları

Aşağıdaki tabloda, bir AVD ile çalışmak için kullanışlı kısayollar bulunmaktadır.

Kısayol	Açıklama
Alt+Enter	Emülatör penceresini büyütür.
Ctrl+F11	Emülatör pençesinin yönelimini sırasıyla yatay ve dikey olarak değiştirir.
F8	Ağı açıp kapatır.

6.4. Google ile Android AVD Karşılaştırması

Bir AVD cihazının oluşturulması sırasında Android cihazı mı yoksa Google cihazını kullanmak istediğinize karar vermeniz gerekir. Android için oluşturulan bir AVD Android Open Source Project programlarını içerir. Google API için oluşturulan bir AVD ek olarak Google'a özel kodlar içerir. Google API için oluşturulan AVD, yeni Google Maps API ya da yeni konum servisleri gibi Google Play hizmetleri kullanan uygulamaları test etmenizi sağlar.

6.5. GPU rendering ile hız optimizasyonu

Bir emülatör oluşturma sırasında Snapshot veya Host GPU seçeneğinden birisini kullanmayı tercih edebilirsiniz.

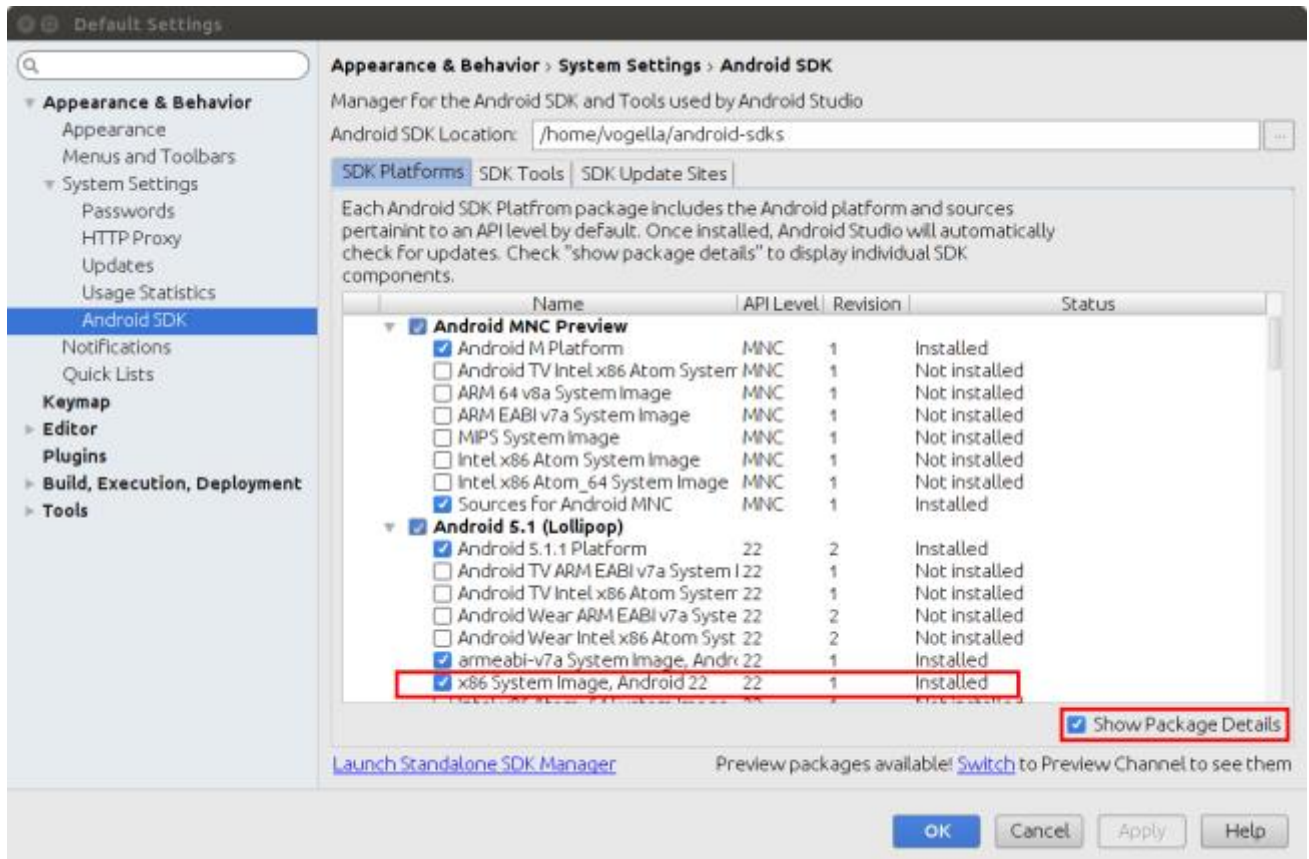
NOT: İletişim kutusundan iki seçeneği birden işaretleme imkanı sunulmuştur ancak bunu yaparsanız seçeneklerden her ikisinin seçilemeyeceğini belirten bir hata mesajı alırsınız.

Snapshot seçeneğini seçerseniz ikinci kez aygıtı başlattığınızda cihaz çok hızlı başlayacaktır çünkü AVD kapatıldığında durumunu saklayacaktır. Host GPU seçeneğini seçerseniz AVD bilgisayarın ekran kartını kullanır böylece cihazdaki render işlemleri çok daha hızlı yapılır.

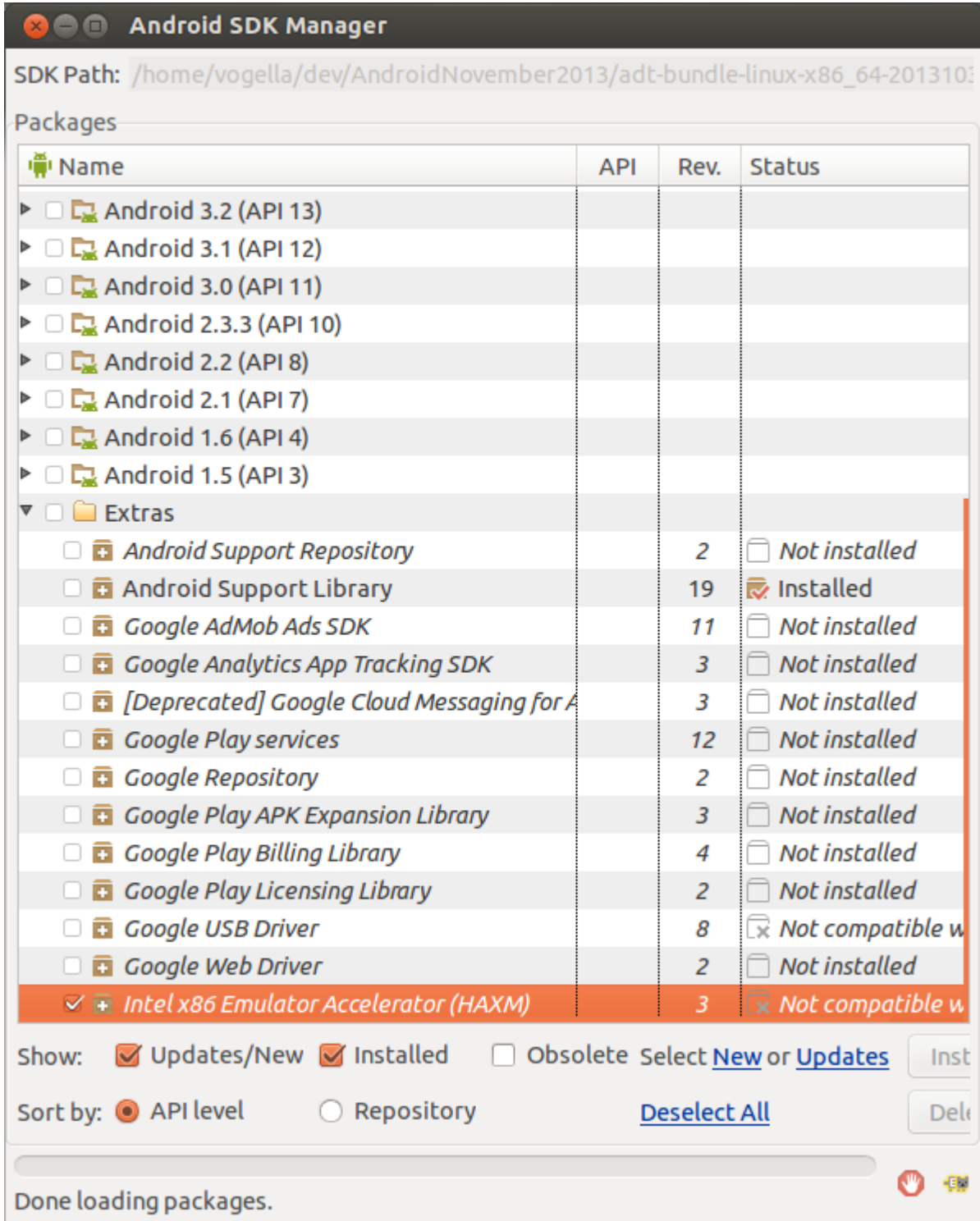


6.6. Intel sistem görüntüsü ile Hız optimizasyonu

Bir AVD sistem görüntüsünü (image) ARM işlemci mimarisine veya Intel CPI mimarisine dayalı olarak çalıştırabilmeniz mümkündür. Intel sistem görüntüsü kullanan bir Android sanal aygıt ARM tabanlı sistem görüntüye göre Intel/AMD donanım üzerinde çalıştırıldığında çok daha hızlı çalışmaktadır. Bunun nedeni Emülatörün bilgisayarınızda ARM işlemci yönergelerini Intel / AMD CPU yönergelerine çevirmesine gerek olmamasıdır. Bir API için Intel sistem görüntü Android SDK Yöneticisi ile kurulabilir. Android Studio'da yeni bir aygıt oluştursanız bu otomatik olarak gerçekleşir. Paket detayları seçeneği ile de bu yapılandırmayı yapmak mümkündür.



Not: Intel sistem görüntüsünün tüm API düzeyleri için bulunmadığını unutmayınız.



Bu yazının yazıldığı anda Windows işletim sistemi için ekstra sürücüler indirmek ve yüklemek zorunludur. İndirme işlemi tamamlandıktan sonra Android kurulum dizinindeki extras/intel klasöründe sürücüler bulabilirsiniz. Sürücüler yüklemeniz için .exe dosyasını çalıştırmamız yeterlidir. Bu ek kurulum aşaması Intel emülatörünü Windows işletim sisteminde hızlandırmak için gereklidir. Sadece Android aracılığıyla sürücüyü indirmek her hangi bir fark sağlamaz.

İndirme işleminden sonra Intel emülatör tabanlı yeni bir AVD oluşturabilirsiniz. Emülatör hızlı başlamaz fakat Android uygulamaların çalıştırılması daha hızlı olacaktır.

İpucu: Linux işletim sisteminde daha karmaşık kurulum gerekir. Ayrıntılı bir Intel emülatörü kurulum açıklaması için kurulum kılavuzuna <https://software.intel.com/en-us/android/articles/intel-hardware-accelerated-execution-manager> adresinden bakabilirsiniz. Bu sitede ayrıca Windows için de ayrıntılı kılavuz bulunmaktadır.

6.7. Alternatif emülatör olarak Genymotion

Mevcut varsayılan Android emülatörüne alternatifler bulunmaktadır. Örneğin Genymotion emülatörü Android projelerinin başlatma ve çalıştırılmasında nispeten daha hızlıdır.

<https://www.genymotion.com/> adresinden kişisel kullanım için ücretsiz sürümü indirilebilir.

6.8. Test için gerçek bir Android cihazı kullanma

Kullanılan Android cihazının ayarlarından USB hata ayıklama özelliğinin açılması gerekmektedir. Bunun için Seçenek → Geliştirici Seçenekleri menüsü seçilerek USB hata ayıklama-seçeneğinin işaretlenmesi yeterlidir.

Ayrıca cep telefonunuz için sürücü yüklemeniz gerekebilir. Linux ve Mac OS genelde ek bir işlem gerektirmeden çalışırken Windows tipik bir sürücü kurulum işlemi gerektirir.

Windows Sürücü kurulumu ile ilgili detaylar için <http://developer.android.com/tools/device.html> adresinden Google kılavuzuna bakınız.

NOT: Android uygulama için seçilen minimum Android sürümü cihazınızdaki Android sürümüne uygun olmalıdır.

Eğer bilgisayarınıza bağlı birden çok cihaz varsa, kullanmak istediğiniz birisini seçebilirsiniz. Sadece bir cihaz bağlıysa, uygulama otomatik olarak bu cihaza dağıtılır.

7. Uygulama: Android Studio ile Geliştirmeye Başlayın

7.1. Hedef

Bu uygulamada bir Android projesi oluşturma ve Android Studio üzerinden bir Android sanal cihaz oluşturulması anlatılacaktır.


7.2. Yeni Android projesi oluşturun

Android Studio File->New Project menüsü ile yeni bir proje oluşturulması işlemi başlatılır.

Projeniz için aşağıda verilen bilgileri kullanabilir, kendi belirlediğiniz verileri de kullanabilirsiniz. Projenin yeri ve paket adı girdiğiniz verilerden üretilmiştir. Başka bir paket adı istiyorsanız, Edit düğmesine basabilirsiniz. Uygulama isminin büyük harfle başlaması gerekmektedir.

Property	Value
Company Domain	yapbenzet.kocaeli.edu.tr
Package Name	Tr.edu.kocaeli.yapbenzet.uygulama1
API (Minimum, Target, Compile with)	19
Template	Latest
Application name	Uygulama1

Create New Project

 **New Project**
Android Studio

Configure your new project

Application name:

Company Domain:

Package name: [Edit](#)

Project location: ...

Previous

Next

Cancel

Finish

Create New Project

Target Android Devices

Select the form factors your app will run on

Different platforms may require separate SDKs

☒ Phone and Tablet

Minimum SDK: API 19: Android 4.4 (KitKat)

Lower API levels target more devices, but have fewer features available.
By targeting API 19 and later, your app will run on approximately 62.6% of the devices that are active on the Google Play Store.
[Help me choose](#)

☐ Wear

Minimum SDK: API 21: Android 5.0 (Lollipop)

☐ TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

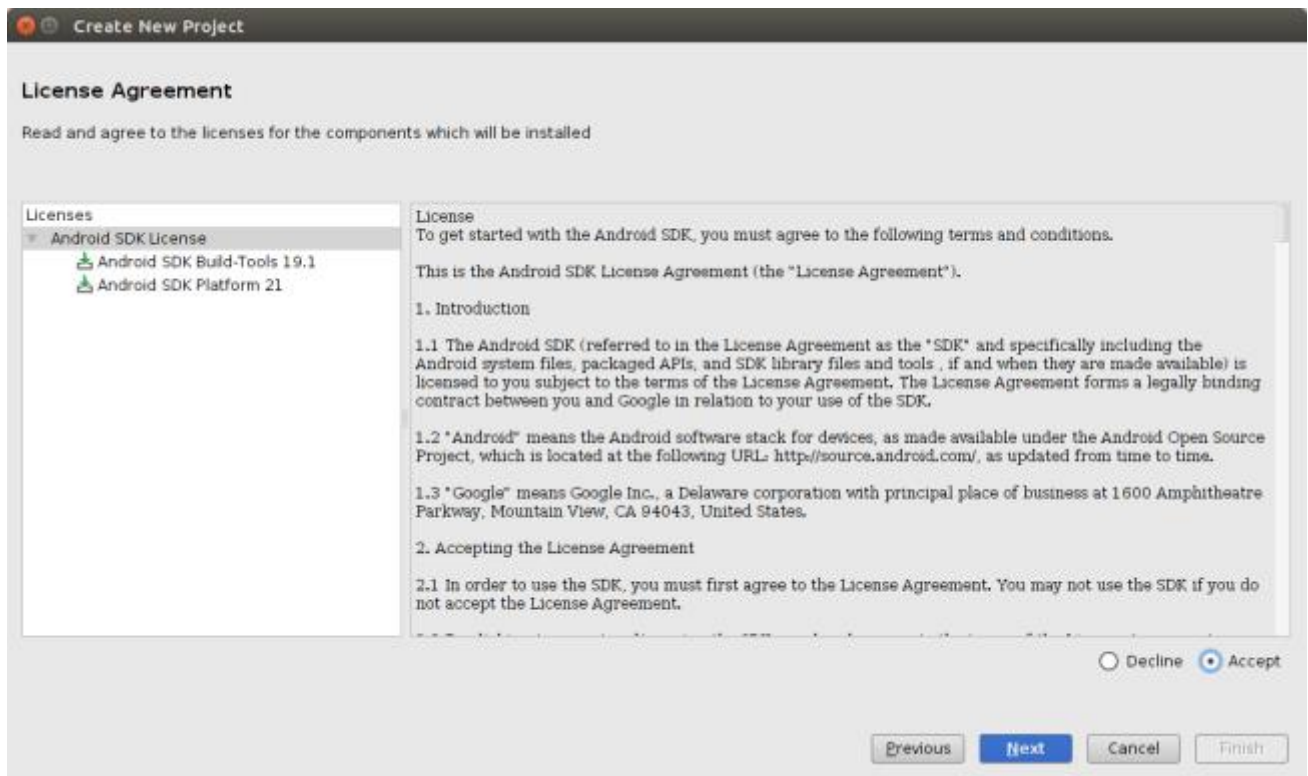
☐ Android Auto

☐ Glass

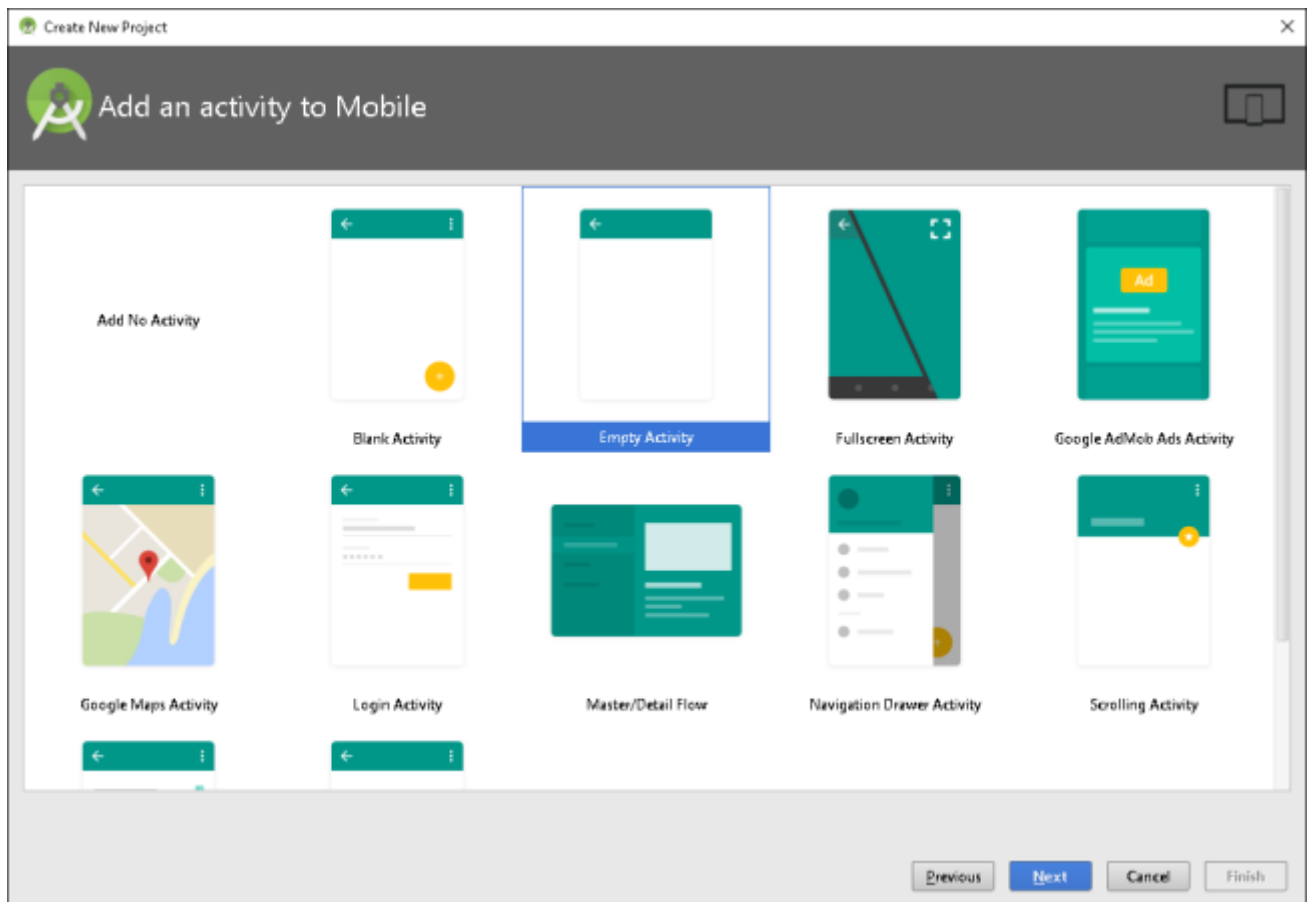
Minimum SDK: Glass Development Kit Preview

Previous Next Cancel Finish

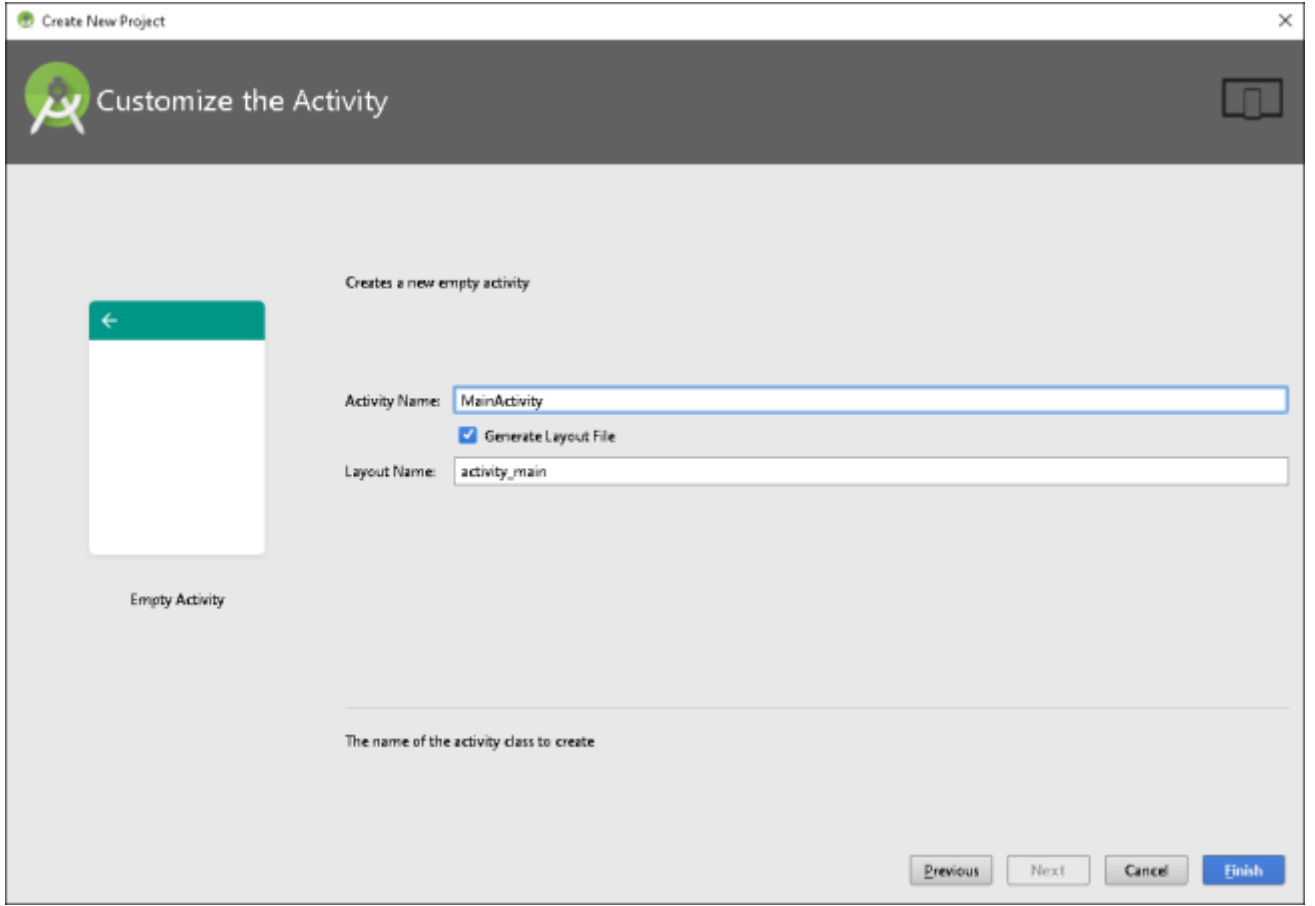
Eğer daha önce indirilmediyse Android Studio gerekli Android SDK araçlarını otomatik olarak indirir. Yani kurulum bağlı olarak bir sonraki pencere ile karşılaşmayabilirsiniz.



"Empty Activity" seçeneğini seçiniz.



En son sayfada varsayılan değerleri kabul edebilirsiniz.



Create New Project

Customize the Activity

Creates a new empty activity

Activity Name: MainActivity

☒ Generate Layout File

Layout Name: activity_main

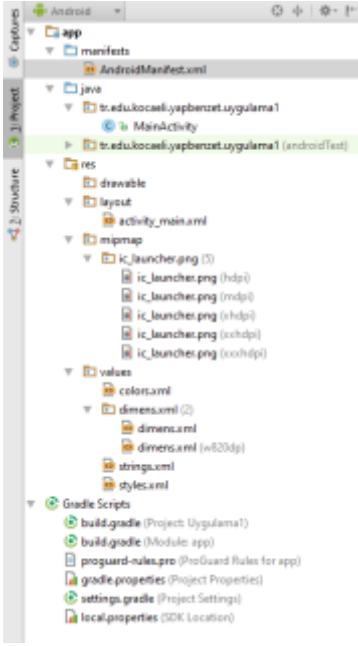
Empty Activity

The name of the activity class to create

Previous Next Cancel Finish

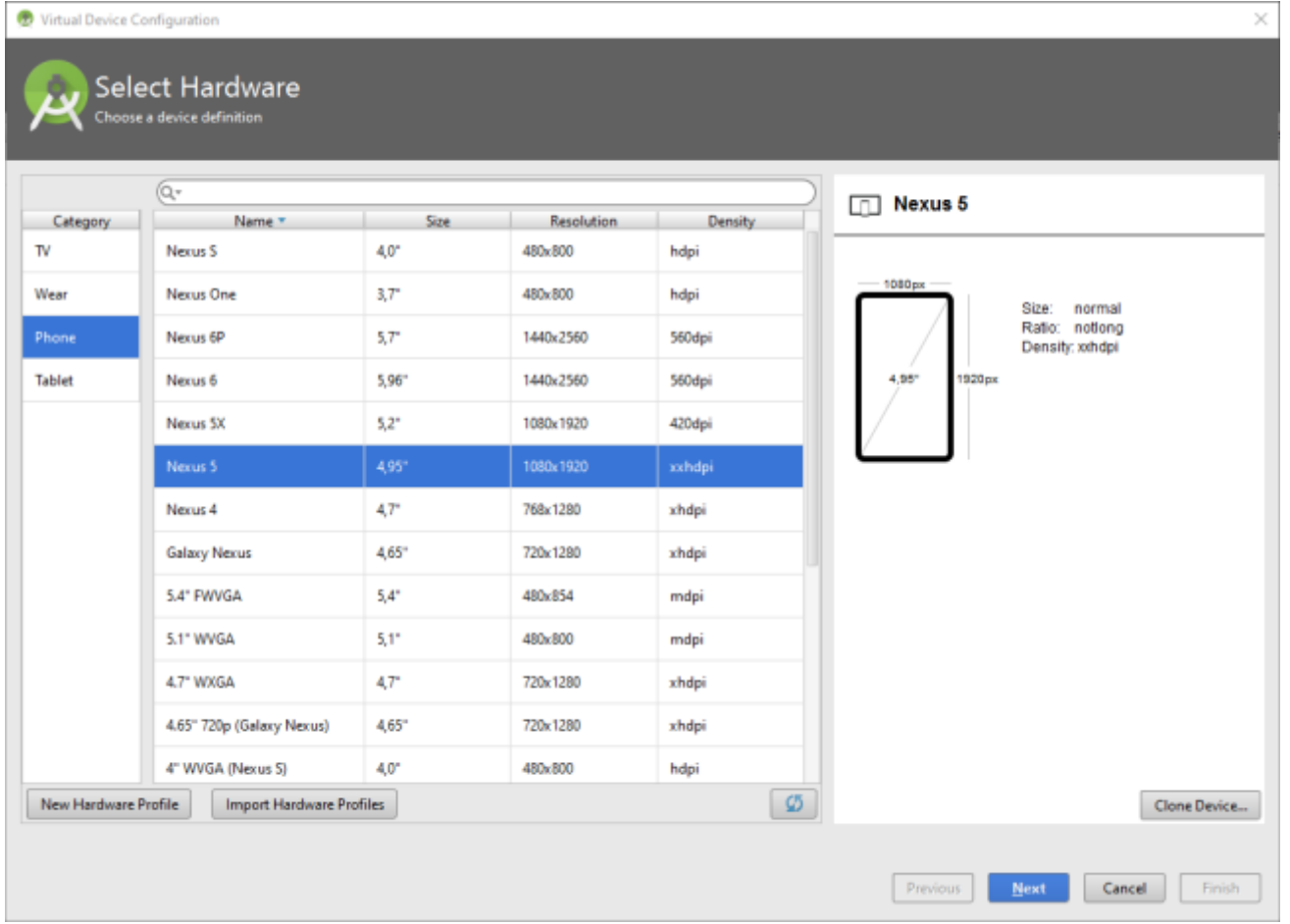
7.3. Oluşturulan proje gözden geçirin

Yukardaki Şirbaz girilen bilgilere göre bir Android projesi üretir. Oluşturulan proje yapısı ve dosyalarını gözden geçirin.

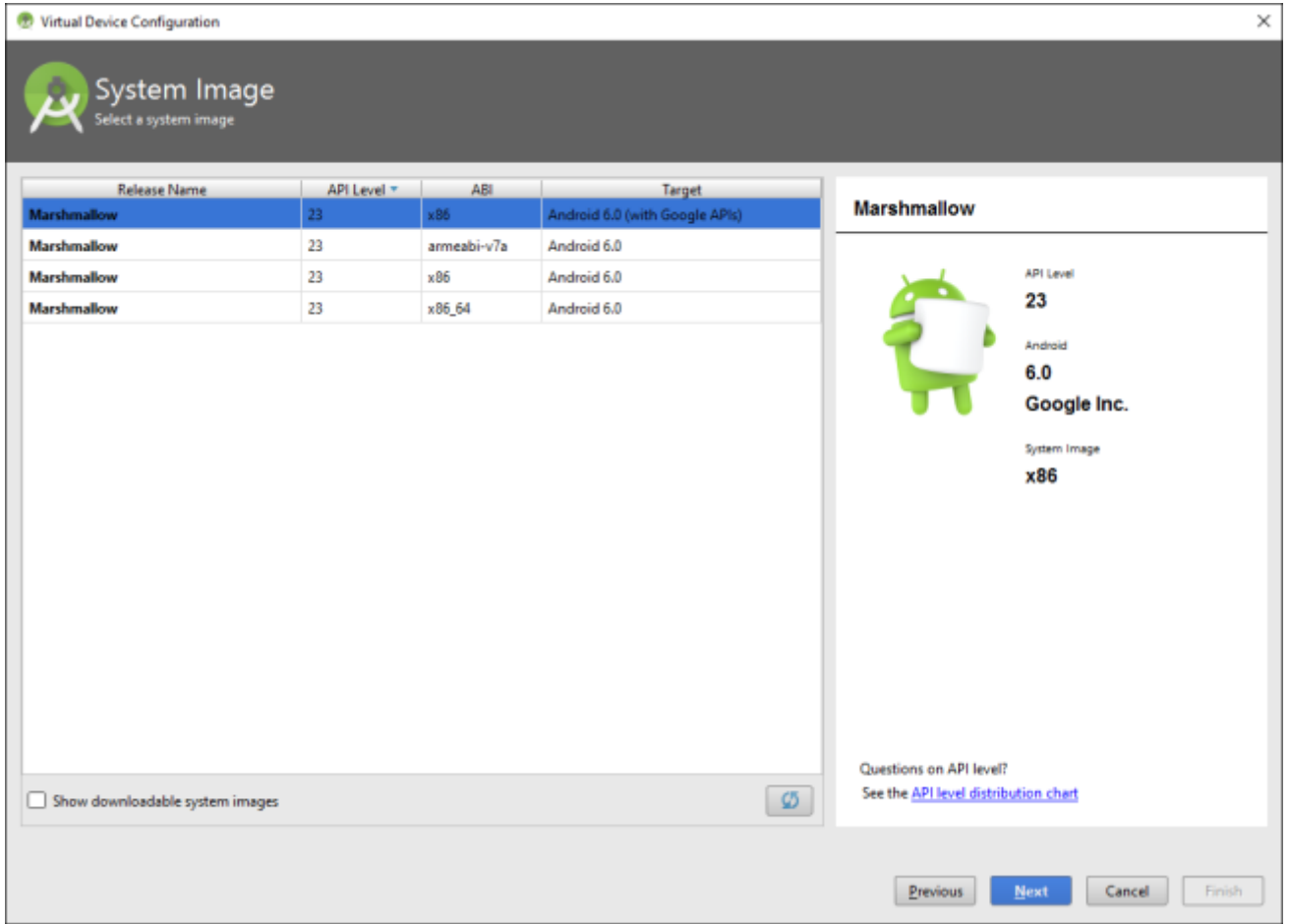


7.4. Bir sanal aygıt oluşturun (AVD)

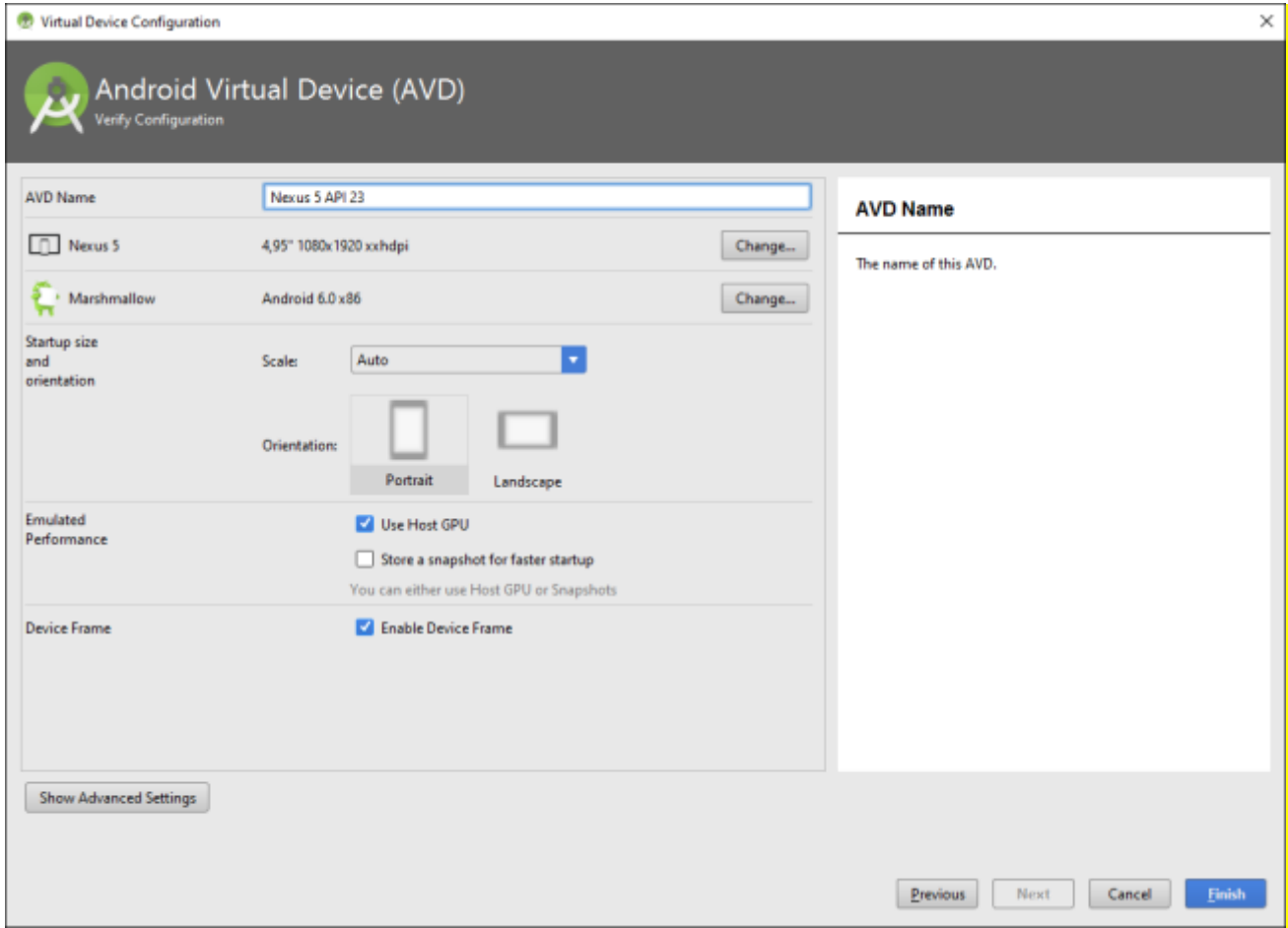
Yeni bir Android Sanal Aygıt (AVD) tanımlamak için Tools → Android → AVD Manager seçeneğini seçiniz. Daha sonra Create Virtual Device düğmesine basınız.



Next düğmesine bastığınızda AVD için en güncel API seviyesini seçebilirsiniz. "Show downloadable system images" seçeneği ile daha fazla indirilebilir sistem görüntülerini görebilirsiniz. Google API içeren sistem görüntüsü kullanmanız tavsiye edilmektedir, bu durumda GOOGLE hizmetlerini de test etme imkanınız olacaktır.

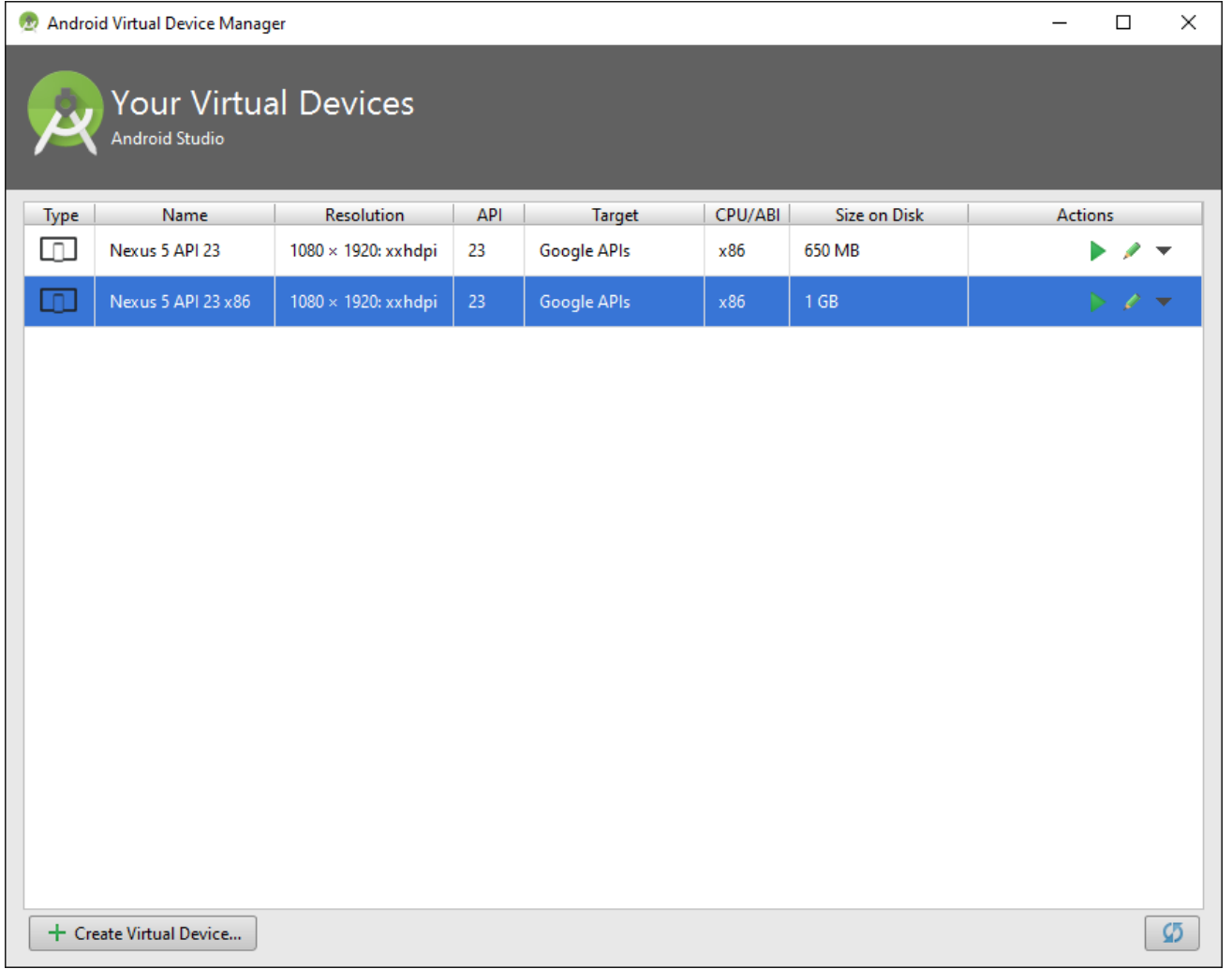


AVD için bir isim verebilirsiniz. Finish düğmesine bastığınızda AVD oluşturma işlemini tamamlamış olacaksınız.



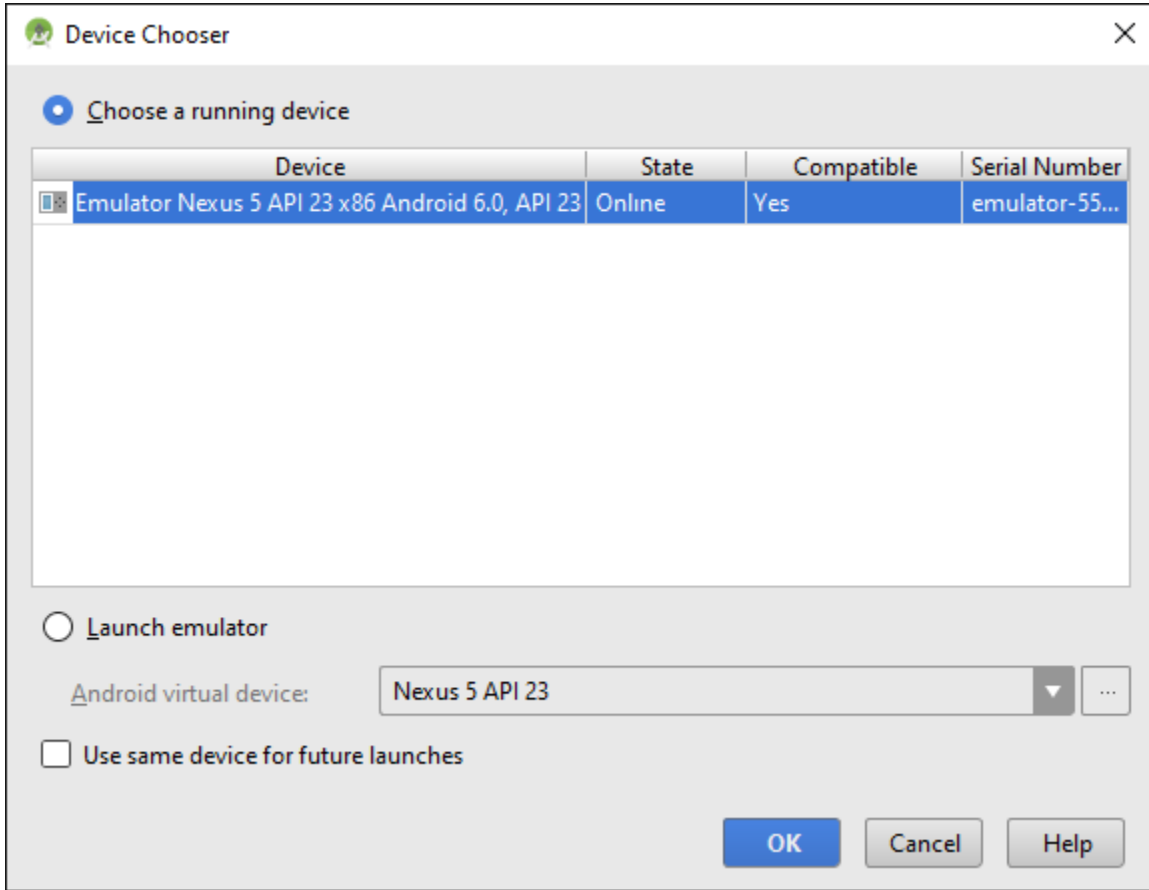
7.5. Sanal cihazınızı başlatın

Hazırladığınız AVD'lerden birisini seçerek başlatınız.



7.6. Sanal cihaz üzerinde uygulamayı başlatın

Uygulamanızı başlatmak için Run → Run 'app' menüsünü seçiniz. Uygulamanın dağıtılacağı (deploy) bir cihazı seçmenizi isteyen bir iletişim kutusu açılacaktır.



Çalışan emülatör seçildiğinde uygulamanız dağıtım yapıldıktan sonra başlatılacaktır.



8. Bir Android Uygulamasının Parçaları

8.1. Android uygulaması

Bir Android uygulaması diğer android uygulamalarından bağımsız olarak başlatılıp kullanılabilen, cihaza yüklenebilen tek bir birimdir. Bir Android uygulaması Android bileşenleri, Java kaynak kodu ve kaynak (resource) dosyalarından oluşur. Android uygulama bileşenleri Intent nesnesi ile diğer Android uygulamaları bileşenlerine bağlanabilir. Bu şekilde çapraz başvuru görevleri oluşturulabilir. Bu bileşenlerin entegrasyonu ek bileşenlerin yüklü olmasa veya farklı bileşenler aynı görevi gerçekleştirmek durumunda bile Android uygulaması kusursuz çalışacak şekilde yapılabilir.

8.2. Android yazılım bileşenleri

Aşağıdaki Android bileşenleri tanımlanabilir:

- Uygulama (Application)
- Faaliyetler (Activities)
- Hizmetler (Services)
- Yayın alıcıları (Kısaca alıcılar) (Broadcast receivers)
- İçerik sağlayıcılar (Kısaca sağlayıcılar) (Content providers)

8.3. Uygulama (Application)

Bir Android uygulaması sadece bir tane Application sınıfına sahip olabilir. Android uygulaması başlar başlamaz bu Application sınıfından nesneleştirme (instantiate) yapılır ve uygulama kapatılana kadar kalır.

Açıkça tanımlanmamışsa Android uygulamanız için varsayılan bir uygulama nesnesi oluşturur.

8.4. Etkinlik (Activity)

Bir etkinlik bir Android uygulamasının görsel kısmını temsil eder. Bir Android uygulaması pek çok etkinliğe (activity) sahip olabilir.

Kullanıcı arayüzü oluşturmak, kullanıcı ile etkileşim ve fragmanları ilerleyen bölümlerde anlatılacaktır.

8.5. Yayın alıcıları (BroadcastReceiver)

Bir yayın alıcısı (receiver) sistem mesajları ve intent'leri dinlemek için kaydedilebilir (register). Belirtilen olay meydana geldiğinde bir alıcı (receiver) Android sistemi tarafından haberdar edilir. Örneğin, Android sisteminin önyükleme işlemi bitmesi olayı veya bir çağrı gelmesi vb. telefon durumunun değişmesi olayı için bir alıcı kayıt edebilirsiniz.

8.6. Hizmet (Service)

Hizmet (service) bir kullanıcı arabirimi sağlamadan görevlerini gerçekleştirir. Örneğin Android içindeki bildirim framework yoluyla kullanıcıyı uyarabilir ve yayın alıcıları (broadcast receiver) yardımıyla diğer Android bileşenleri ile iletişim kurabilirler.

8.7. İçerik sağlayıcı (Content providers)

Bir içerik sağlayıcı (sağlayıcı) uygulama verilerine yapılandırılmış arabirim sunar. Bir uygulama içindeki verilere erişmek için sağlayıcı kullanılabilir aynı zamanda diğer uygulamalar ile veri paylaşımı için de sağlayıcı kullanılabilir.

Android bir içerik sağlayıcısı ile bağlantılı olarak yoğun bir şekilde kullanılan SQLite veritabanına sahiptir. SQLite veritabanında saklanan tüm verilere sağlayıcı üzerinden erişilebilir.

8.8. Bağlam (Context)

Çalışan bir uygulamanın Android sistemi ile bağlantısı android.content.Context sınıfının oluşumları (instance) ile sağlanır. Aynı zamanda proje kaynaklarına erişim imkanı ve uygulama ortamı hakkında küresel bilgi sağlar. Örneğin, Bağlam (Context) sayesinde kullanılan aygıtın ekranın boyutunu kontrol edebilirsiniz. Bağlam sınıfı ayrıca örneğin zaman tabanlı olayları tetiklemek için alarm yöneticisi gibi Android hizmetlerine erişim imkanı sağlar. Etkinlikler ve hizmetler Bağlam sınıfından türetilmiştir (extend). Bu nedenle, Bağlama doğrudan erişmek için kullanılabilir.

9. Temel Kullanıcı Arabirim Bileşenleri

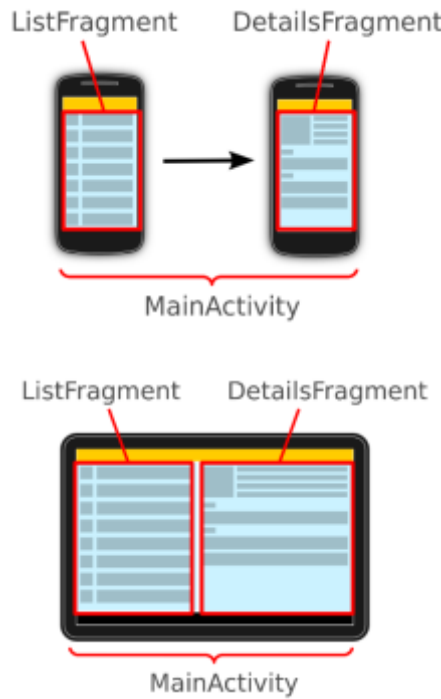
Aşağıdaki açıklama en önemli kullanıcı arabirimi ile ilgili bileşen ve bir Android uygulama parçalarının bir bakış verir.

9.1. Etkinlik (Activity)

Android kullanıcı arayüzü için temel yapıtaşı Etkinliklerdir. Zaten Bölüm 8.4'de "Etkinlik (Activity)" tanıtılmıştır.

9.2. Parçalar (Fragments)

Parçalar (Fragments) bir etkinliğin bağlamı (context) kapsamında çalışacak bileşenleridir. Bir parça (fragment) uygulama kodunu kapsüller (encapsulate) dolayısıyla tekrar kullanımı kolaylaşır ve farklı boyutta cihazlarda kullanımını destekler. Aşağıdaki resim MainActivity adında bir etkinliği göstermektedir. Küçük bir ekranda sadece bir parçası gösterir ve kullanıcının diğer parçalara gezinmesine izin verir. Geniş ekranda ise her iki parçayı da göstermektedir.



9.3. Görünüm (View) ve Düzen (Layout) Yöneticisi

Görünümler, örneğin düğme veya metin alanları gibi kullanıcı arayüzü widgetleridir. Görünümler kendi görünüm ve davranışlarını yapılandırmak için kullanılabilecek özelliklere (attribute) sahiptir.

Bir ViewGroup diğer görünümün yerleşiminden sorumludur. Aynı zamanda düzen yöneticisi olarak da bilinir. Bu düzen yöneticileri için temel sınıf android.view.ViewGroup sınıfıdır ve görünümü için temel sınıftan android.view.View sınıfından türetilmiştir (extends). Düzen yöneticileri karmaşık düzenleri oluşturmak için iç içe kullanılabilirler.