packt

PREVOD DRUGOG IZDANJA

KOTLIN

ZA ANDROID APLIKACIJE

Praktičan vodič za razvoj, testiranje i publikovanje prve Android aplikacije

ALEX FORRESTER
ERAN BOUDJNAH
ALEXANDRU DUMBRAVAN
JOMAR TIGCAL

kompjuter biblioteka

PREVOD DRUGOG IZDANJA

KOTLIN

ZA ANDROID APLIKACIJE

Skenirajte QR kod, registrujte knjigu i osvojite nagradu

Želite da se bavite razvojem aplikacija za Android 13, ali ne znate kako da počnete? Izgradnja Android aplikacija uz Kotlin je sveobuhvatan vodič uz koji možete da započnete karijeru App Developer-a.

Knjiga počinje od osnova razvoja aplikacija, predstavlja Android Studio i Kotlin i početak gradnje Android projekta. Uz vođene vežbe, naučićete da kreirate aplikacije i da ih pokrećete na virtuelnim uređajima. Dok napredujete kroz poglavlja, upoznajete Android biblioteku RecyclerView da biste na najbolji način koristili liste, slike i mape i da biste naučili da preuzimate podatke sa veb servisa.

Učićete o testiranju, o održavanju čiste arhitekture, čuvanju podataka i steći ćete osnovno znanje o obrascu injektiranja zavisnosti. Na kraju ćete videti kako se publikuje aplikacija u Google Play prodavnici.

Radićete na realističnim projektima, podeljenim na male vežbe i aktivnosti, koje će svakom početniku biti prijatan i dostižan izazov. Izgradićete aplikaciju za kreiranje kvizova, za čitanje novinskih članaka, proveru izveštaja o vremenskoj prognozi, skladištenje recepata, preuzimanje informacija o filmovima i za pamćenje parking mesta.

Do kraja ove knjige steći ćete veštinu i samopouzdanje da izgradite sopstvene kreativne Android aplikacije pomoću jezika Kotlin.

Naučićete da

- pomoću jezika Kotlin, kreirate skalabilne aplikacije, jednostavne za održavanje
- sagledavate životni ciklus razvoja Android aplikacije
- pojednostavite razvoj aplikacija pomoću Google arhitekturnih komponenti
- koristite standardne biblioteke za injektiranje zavisnosti i raščlanjivanje podataka
- primenjujete obrazac skladišta za preuzimanje podataka iz spoljnih izvora
- izgradite korisničke interfejse pomoću biblioteke Jetpack Compose
- istražujete Android asinhrono programiranje pomoću biblioteke Coroutines i Flow API-ja
- objavljujete svoju aplikaciju u Google Play prodavnici

ISBN: 978-86-7310-591-8

Kotlin za Android aplikacije

Prevod II izdanja

Praktičan vodič za razvoj, testiranje i publikovanje prve Android aplikacije

Alex Forrester
Eran Boudjnah
Alexandru Dumbravan
Jomar Tigcal

kompjuter biblioteka

Packt›

Izdavač:

kompjuter biblioteka

Obalskih radnika 4a
Beograd, Srbija

Tel: 011/2520272

e-pošta: kombib@gmail.com

veb-sajt: www.kombib.rs

Za izdavača:
Mihailo J. Šolajić, direktor

Autori:
Alex Forrester
Eran Boudjnah
Alexandru Dumbravan
Jomar Tigcal

Prevod: Slavica Prudkov

Lektura: Nemanja Lukić

Slog: Zvonko Aleksić

Znak Kompjuter biblioteke:
Miloš Milosavljević

Štampa: „Pekograf“, Zemun

Tiraž: 500

Godina izdanja: 2023.

Broj knjige: 568

Izdanje: Prvo

ISBN: 978-86-7310-5xx-x

Naslov originala:

How to Build Android Apps with Kotlin
Second Edition
ISBN 978-1-83763-493-4

Packt Publishing Ltd.
Birmingham, UK, packt.com

Kotlin za Android aplikacije

Autorizovani prevod sa engleskog jezika.

SARADNICI

O AUTORIMA

Alex Forrester je iskusan programer softvera sa više od 20 godina iskustva u programiranju za mobilne uređaje, za veb i za sisteme za upravljanje sadržajem. Programira za Android platformu od 2010. godine i kreira vodeće aplikacije za blue-chip kompanije iz širokog spektra industrija, Sky, The Automobile Association, HSBC, Discovery Channel i O2. Aleks živi u Hertfordširu sa suprugom i ćerkom. Pored programiranja, voli ragbi i trčanje po brdima Chiltern.

Eran Boudjnah je programer sa preko 20 godina iskustva u razvoju desktop aplikacija, veb sajtova, interaktivnih atrakcija i mobilnih aplikacija. Koristi Android platformu od 2011. godine, razvija aplikacije i vodi mobilne timove za širok spektar klijenata, od novoosnovanih (JustEat i Plume Design) do velikih kompanija (Sky i HSBC) i konglomerata. Voli društvene igre (poseduje skromnu kolekciju od nekoliko stotina igara) i ima kolekciju Transformersa na koju je prilično ponosan. Eran živi u Brentvudu, u Engleskoj, sa Leom, svojom suprugom.

Alexandru Dumbravan je Android programer sa više od 10 godina iskustva u izradi Android aplikacija, a fokusiran je na fintech aplikacije od 2016. godine, kada se preselio u London. U slobodno vreme Aleks uživa u video igricama, filmovima i povremenim posetama teretani.

Jomar Tigcal je Android programer sa preko 14 godina iskustva u razvoju softvera i aplikacija za mobilne uređaje Radio je na različitim fazama razvoja Android aplikacija za mala novoosnovana preduzeća i velike kompanije od 2012. godine. Jomar je takođe držao predavanja i vodio obuku i radionice o Androidu. U slobodno vreme voli da trči i da čita. Živi u Vankuveru, BC, Kanada, sa suprugom Selin.

O RECENZENTIMA

Ed Holloway-George je Android programer i Google Developer Expert poreklom iz Oksforda u Engleskoj, a trenutno živi u Notingemu. Android programer je nešto više od 10 godina. Ed sada radi za ASOS kao vodeći programer, koji je ranije radio na dobro poznatim aplikacijama, kao što su National Trust, My Oxfam, Snoop, Carling Tap i mnoge druge.

U slobodno vreme Ed govori na konferencijama, piše postove na blogu i deli slike svog psa.

Guruprasad Bagade je stariji programer i timlider sa više od decenije iskustva u razvoju mobilnih aplikacija i softvera. Bio je svedok promena u razvoju Android-a od Java-e do Kotlin-a sa najnovijim bibliotekama radnog okvira. Prvenstveno je radio u domenu bankarstva za klijente Barclays i JP Morgan. Angažovao je i početnike i iskusne programere za organizacije i pomogao u formiranju timova, a istovremeno je objavljivao članke o Androidu na internim portalima različitih organizacija u kojima je radio.

Objavio je tehničke istraživačke radove na **Institute of Electrical and Electronic Engineers** (**IEEE**) i na međunarodnim i nacionalnim konferencijama. Takođe, doprinosi projektima otvorenog koda. U slobodno vreme prati najnovije tehnologije.

Predgovor

Android vlada tržištem aplikacija u poslednjoj deceniji, a programeri sve više grade sopstvene Android aplikacije. Knjigu *Kotlin za Android aplikacije* počinjemo opisom gradivnih blokova Android razvoja i učimo vas da koristite Android Studio, **integrisano razvojno okruženje** (**IDE**) za Android, sa programskim jezikom Kotlin, za razvoj aplikacija.

Zatim ćete naučiti da kreirate aplikacije i da ih pokrećete na virtuelnim uređajima uz vođene vežbe. Obuhvatićemo osnove Android razvoja, od strukturiranja aplikacije do izgradnje korisničkog interfejsa, sa aktivnostima, fragmentima i različitim navigacionim obrascima. Napredujući kroz poglavlja, učićete o Android biblioteci RecyclerView, da biste na najbolji način iskoristili prikazivanje lista podataka, upoznali preuzimanje podataka sa veb servisa i rukovanje slikama.

Zatim, učićete o mapiranju, servisima lokacije i modelu dozvola, pre nego što počnete da koristite obaveštenja i da skladištite podatke. Zatim ćete izgraditi korisničke interfejse pomoću biblioteke Jetpack Compose. Naučićete više o testiranju, obuhvatajući ceo spektar testne piramide. Takođe, naučićete da koristite **Android Architecture Components** (**AAC**) za čisto strukturiranje koda i istraživanje različitih arhitekturnih obrazaca i prednosti injektiranja zavisnosti.

Coroutines i Flow API su opisani za asinhrono programiranje. Fokus, zatim, vraćamo na korisnički interfejs i pokazaćemo vam kako da dodate kretanje i prelaze kada korisnici stupe u interakciju sa vašim aplikacijama. Pri kraju knjige ćete izgraditi zanimljivu aplikaciju za preuzimanje i prikazivanje popularnih filmova iz baze podataka, a zatim ćete videti kako da objavite svoje aplikacije na Google Play prodavnici.

Do kraja ove knjige imaćete veštine i samopouzdanje potrebne za izgradnju potpuno razvijenih Android aplikacija pomoću jezika Kotlin.

Kome je ova knjiga namenjena

Ako želite da gradite sopstvene Android aplikacije pomoću jezika Kotlin, ali niste sigurni kako da počnete, onda je ovo knjiga za vas. Osnovno razumevanje programskog jezika Kotlin će vam pomoći da brže shvatite teme obrađene u ovoj knjizi.

Šta obuhvata ova knjiga

Poglavlje 1, *Kreiranje prve aplikacije* - pokazaćemo vam kako da koristite Android Studio da biste izgradili svoju prvu Android aplikaciju. Kreiraćete Android Studio projekat, shvatićete od čega se sastoji i istražićete alatke neophodne za izgradnju i raspoređivanje aplikacije na virtuelnom uređaju. Takođe, učićete o strukturi Android aplikacije.

Poglavlje 2, *Izgradnja tokova korisničkog ekrana* - govorićemo o Android ekosistemu i gradivnim blokovima Android aplikacije. Predstavićemo koncepte, kao što su aktivnosti i njihov životni ciklus, namere i zadaci, kao i vraćanje stanja i prosleđivanje podataka između ekrana ili aktivnosti.

Poglavlje 3, *Kreiranje korisničkog interfejsa pomoću fragmenata* - učićete o osnovama korišćenja fragmenata za korisnički interfejs Android aplikacije. Naučićete da koristite fragmente na više načina, za izgradnju rasporeda aplikacija za telefone i tablete i naučićete da koristite komponente Jetpack Navigation.

Poglavlje 4, *Izgradnja navigacije za aplikaciju* - opisaćemo različite tipove navigacije u aplikaciji. Učićete o fiokama za navigaciju sa kliznim rasporedom, navigaciji na dnu i navigaciji sa karticama.

Poglavlje 5, *Osnovne biblioteke: Retrofit, Moshi i Glide* - naučićete da izgradite aplikacije koje preuzimaju podatke iz udaljenog izvora podataka, uz korišćenje biblioteka Retrofit i Moshi za konvertovanje podataka u Kotlin objekte. Takođe, učićete o Glide biblioteci, koja učitava udaljene slike u vašu aplikaciju.

Poglavlje 6, *Dodavanje i interakcija pomoću biblioteke RecyclerView* - predstavljamo koncept izgradnje lista i njihovog prikazivanja uz pomoć vidžeta RecyclerView.

Poglavlje 7, *Android dozvole i Google mape* - predstavljamo koncept dozvola i kako da ih zatražite od korisnika da bi vaša aplikacija izvršavala određene zadatke a, takođe, predstavljamo i Maps API.

Poglavlje 8, *Servisi, WorkManager i obaveštenja* - detaljno opisujemo koncept pozadinskog rada u Android aplikaciji i kako aplikacija može da izvrši određene zadatke na način koji je nevidljiv korisniku a, takođe, naučićete i da prikažete obaveštenje o ovom radu.

Poglavlje 9, *Izgradnja korisničkih interfejsa pomoću biblioteke Jetpack Compose* - pokazujemo kako funkcioniše biblioteka Jetpack Compose, kako da primenite stilove i teme i kako da koristite Jetpack Compose u projektima koji su započeti fajlovima rasporeda.

Poglavlje 10, *Jedinični testovi i integracijski testovi pomoću radnih okvira JUnit, Mockito i Espresso* - učićete o različitim tipovima testova za Android aplikaciju, radnim okvirima koji služe za svaki tip testa i o konceptu razvoja zasnovanog na testovima.

Poglavlje 11, *Komponente Android arhitekture* - predstavljamo komponente iz Android Jetpack biblioteka, kao što je ViewModel, što će vam pomoći da odvojite poslovnu logiku od koda korisničkog interfejsa. Zatim, pogledaćemo kako možemo da koristimo tokove podataka koje možemo da pratimo, kao što je LiveData, za isporuku podataka korisničkom interfejsu. Na kraju, pogledaćemo biblioteku Room za analizu zadržavanja podataka.

Poglavlje 12, *Trajni podaci* - pokazujemo različite načine čuvanja podataka na uređaju, od SharedPreferences do fajlova. Takođe, predstavićemo koncept Repository, dajući vam ideju kako da strukturirate svoju aplikaciju u različitim slojevima.

Poglavlje 13, *Injektiranje zavisnosti pomoću radnih okvira Dagger, Hilt i Koin* - objašnjavamo koncept injektiranja zavisnosti i prednosti koje ono pruža aplikaciji. Predstavljeni su radni okviri, kao što su Dagger, Hilt i Koin, kao pomoć za upravljanje zavisnostima.

Poglavlje 14, *Korutine i tok* - upoznajemo vas sa obavljanjem pozadinskih operacija i manipulacijom podataka pomoću korutina i klase Flow. Takođe, učićete o manipulisanju i prikazivanju podataka pomoću Flow operatora i LiveData transformacije.

Poglavlje 15, *Arhitekturni obrasci* - objašnjavamo obrasce arhitekture, koje možete da koristite da strukturirate svoje Android projekte, da biste ih razdvojili u različite komponente sa različitim funkcionalnostima. To vam olakšava razvoj, testiranje i održavanje koda.

Poglavlje 16, *Animacije i prelazi pomoću funkcija CoordinatorLayout i MotionLayout* - govorimo o tome kako da poboljšate svoje aplikacije animacijama i prelazima, pomoću funkcija `CoordinatorLayout` i `MotionLayout`.

Poglavlje 17, *Pokretanje aplikacije u Google Play prodavnici* - završavamo ovu knjigu tako što ćemo vam pokazati kako da objavite svoje aplikacije u Google Play prodavnici: od pripreme izdanja do kreiranja Google Play Developer naloga i, konačno, pokretanja aplikacije.

Da biste izvukli maksimum iz ove knjige

Svako veliko putovanje počinje skromnim korakom. Pre nego što možemo da izgradimo sjajne elemente u Android-u, potrebno je da pripremimo produktivno okruženje. U ovom odeljku ćete videti kako se to radi.

Minimalni hardverski zahtevi

Za optimalno iskustvo učenja preporučujemo sledeću konfiguraciju hardvera:

- **Procesor**: Intel Core i5 ili ekvivalentan, ili noviji
- **Memorija**: 8 GB RAM-a, ili više
- **Skladište**: Najmanje 8 GB dostupnog prostora

Softverski zahtevi

Takođe, potreban vam je unapred instaliran sledeći softver:

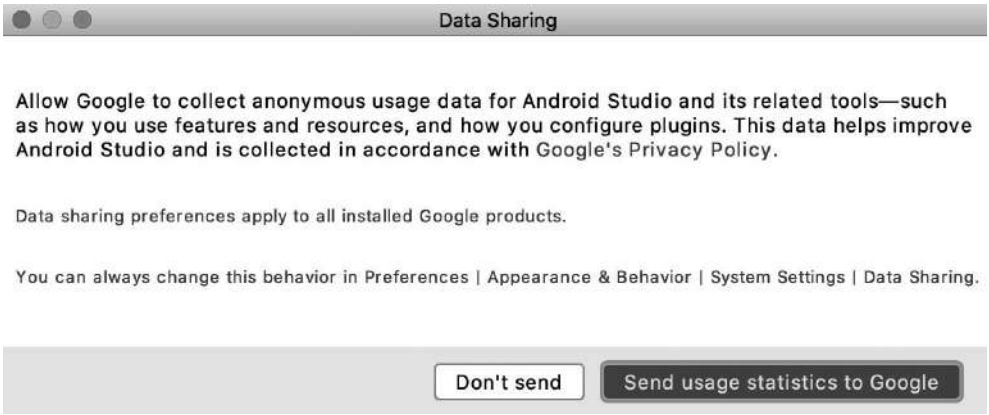
- **OS**: 64-bitni Windows 8/10/11, macOS ili 64-bitni Linux
- Android Studio Electric Eel, ili noviji

Instalacija i podešavanje

Pre nego što počnete da čitate ovu knjigu, potrebno je da instalirate Android Studio Electric Eel (ili noviji), softver koji ćete koristiti. Možete da preuzmete Android Studio na adresi `https://developer.android.com/studio`.

Na macOS sistemu pokrenite DMG fajl i prevucite i otpustite Android Studio u direktorijum `Applications`. Kada to uradite, otvorite Android Studio. U operativnom sistemu Windows pokrenite EXE fajl. Ako koristite Linux, raspakujte ZIP fajl na željenu lokaciju. Otvorite Terminal i otvorite direktorijum `androidstudio/bin/` i izvršite komandu `studio.sh`.

Otvoriće se okvir za dijalog **Data Sharing**; kliknite na dugme **Send usage statistics to Google** ili na dugme **Don't send**, da biste onemogućili slanje anonimnih podataka o korišćenju Google-u:



Okvir za dijalog Data Sharing

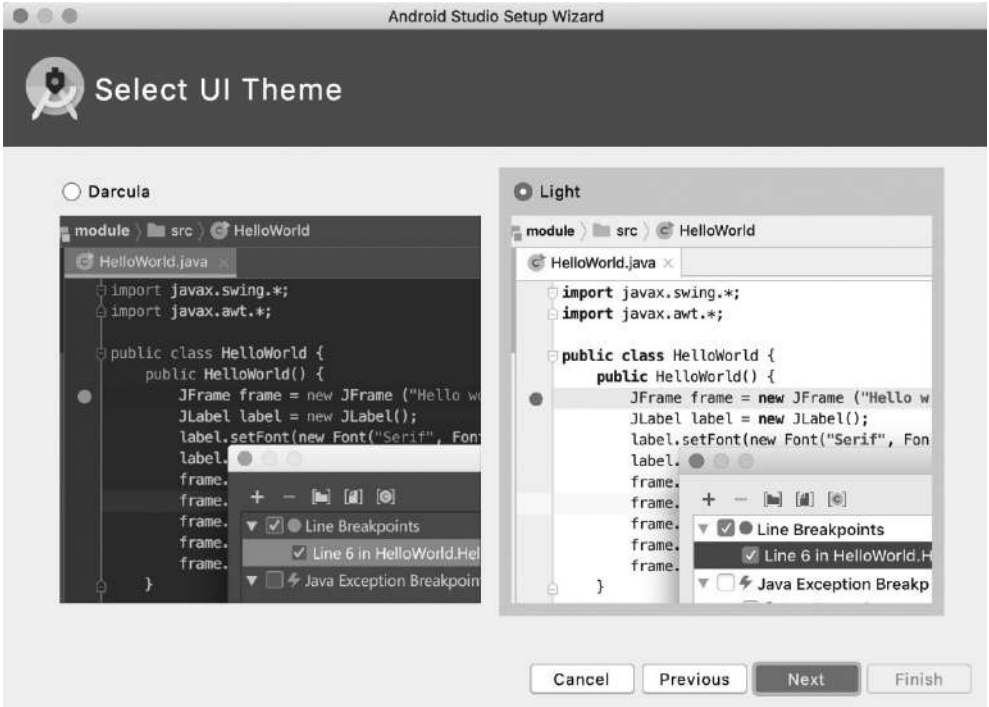
U okviru za dijalog **Welcome** kliknite na dugme **Next** da biste počeli podešavanje:

Okvir za dijalog Welcome

U okviru za dijalog **Install Type** izaberite **Standard,** da biste instalirali preporučena podešavanja. Zatim, kliknite na dugme **Next**:

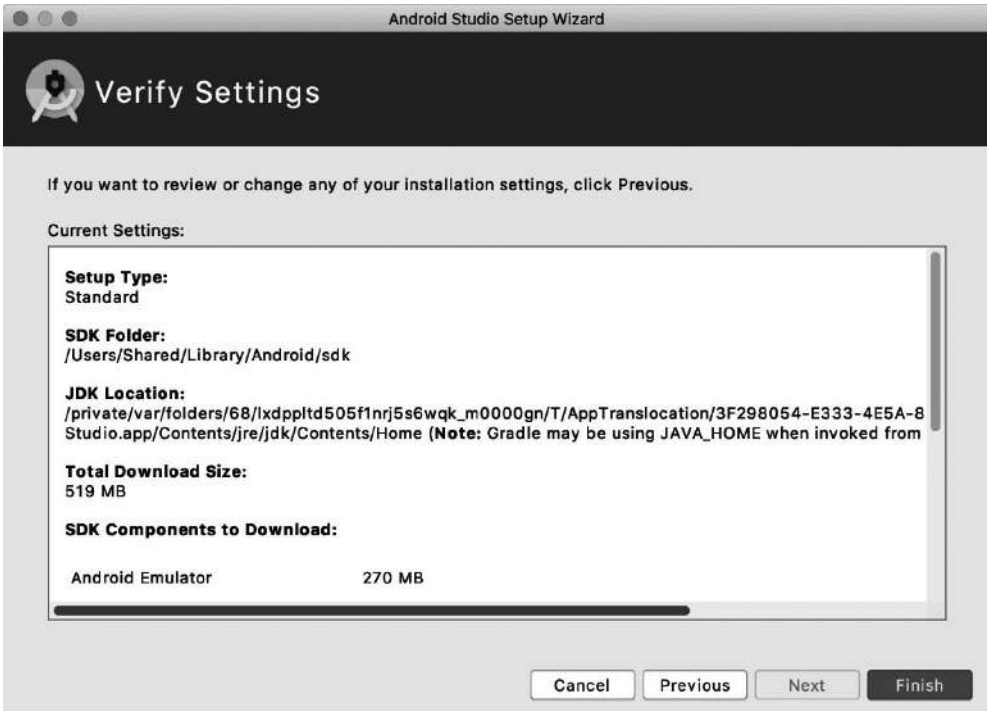
Okvir za dijalog Install Type

U okviru za dijalog **Select UI Theme** izaberite željenu IDE temu — **Light** ili **Darcula** (tamna tema) — a zatim kliknite na dugme **Next**:



Okvir za dijalog Select UI Theme

U okviru za dijalog **Verify Settings**, pregledajte podešavanja, a zatim kliknite na dugme **Finish**. Čarobnjak za podešavanje instaliraće dodatne komponente, uključujući i Android SDK:



Okvir za dijalog Verify Settings

Kada je preuzimanje završeno, kliknite na dugme **Finish**. Sada ste spremni da kreirate svoj Android projekat.

Ako koristite digitalnu verziju ove knjige, savetujemo vam da sami unosite kod ili da pristupite kodu iz GitHub skladišta za ovu knjigu (link je dostupan u sledećem odeljku). To će vam pomoći da izbegnete sve potencijalne greške povezane sa kopiranjem i pejstovanjem koda.

Preuzmite fajlove sa primerima koda

Preuzmite fajlove sa primerima koda za ovu knjigu sa GitHub-a, na adresi `https://github.com/PacktPublishing/How-to-Build-Android-Apps-with-Kotlin-Second-Edition`. Ako postoji ažuriranje koda, biće ažurirano u GitHub skladištu.

Imamo i druge pakete koda iz našeg bogatog kataloga knjiga i video zapisa, koji su dostupni na adresi `https://github.com/PacktPublishing/`. Pogledajte ih!

Preuzmite slike u boji

Takođe, obezbedili smo i PDF fajl koji sadrži kolorne snimke ekrana i dijagrame, koji su korišćeni u ovoj knjizi. Možete ga preuzeti na adresi: `https://packt.link/vnOCn`.

Korišćene konvencije

Postoji veliki broj tekstualnih konvencija koje su korišćene u ovoj knjizi.

`Kod u tekstu`: označava kodne reči u tekstu, nazive tabela baze podataka, nazive direktorijuma, nazive fajlova, ekstenzije fajlova, nazive putanja, lažne URL adrese, korisnički unos i Twitter objave. Na primer: „Možete ga naći u glavnom prozoru projekta, pod `MyApplication|app|src|main`."

Blok koda je postavljen na sledeći način:

```
<resources>
    <string name="app_name">My Application</string>
</resources>
```

Kada želimo da vam skrenemo pažnju na određeni deo bloka koda, relevantne linije ili stavke su podebljane:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">My Application</string>
    <string name="first_name_text">First name:</string>
    <string name="last_name_text">Last name:</string>
</resources>
```

Podebljan tekst: označava novi termin, važnu reč ili reči koje vidite na ekranu. Na primer, reči u menijima ili okvirima za dijalog prikazane su **podebljane**. Na primer: „Kliknite na **Finish** i biće kreiran virtuelni uređaj."

> **Saveti ili** važne napomene
> Prikazane su ovako.

Postanite član Kompjuter biblioteke

Kupovinom jedne naše knjige stekli ste pravo da postanete član Kompjuter biblioteke. Kao član možete da kupujete knjige u pretplati sa 40% popusta i učestvujete u akcijama kada ostvarujete popuste na sva naša izdanja. Potrebno je samo da se prijavite preko formulara na našem sajtu.
Link za prijavu: kombib.rs/kblista.php

Skenirajte QR kod
registrujte knjigu
i osvojite nagradu

DEO 1

Osnova Android-a

U ovom prvom delu korisnicima predstavljamo Android Studio, **integrisano razvojno okruženje** (**IDE**) koje služi za razvoj Android aplikacija, a zatim predstavljamo i gradivne blokove razvoja Android aplikacija. To je sveobuhvatan pregled Android radnog okvira, uz vođene vežbe koje potenciraju ciljeve učenja, da bi se stečeno znanje moglo zadržati.

Ovim delom obuhvaćena su sledeća poglavlja:

- *Poglavlje 1, Kreiranje prve aplikacije*
- *Poglavlje 2, Izgradnja tokova korisničkog ekrana*
- *Poglavlje 3, Kreiranje korisničkog interfejsa pomoću fragmenata*
- *Poglavlje 4, Izgradnja navigacije za aplikaciju*

1

Kreiranje prve aplikacije

Ovo poglavlje je uvod u Android, uz koje ćete podesiti svoje okruženje i fokusirati se na osnove razvoja Android aplikacija. Do kraja ovog poglavlja steći ćete znanje potrebno da kreirate Android aplikaciju od nule i da je instalirate na virtuelni ili fizički Android uređaj.

Moći ćete da analizirate i razumete važnost `AndroidManifest.xml` fajla i da koristite alatku Gradle, da biste konfigurisali svoju aplikaciju i da biste implementirali elemente **korisničkog interfejsa** (**UI**) iz Material Design specifikacije.

Android je najrasprostranjeniji operativni sistem za mobilne telefone, sa preko tri milijarde aktivnih uređaja. To predstavlja mogućnost da, poznavanjem Androida, doprinesete i izvršite uticaj izgradnjom aplikacija koje imaju globalni domet. Međutim, za programera koji je tek počeo da programira za Android postoji mnogo problema sa kojima mora da se suoči da bi započeo učenje i postao produktivan.

U ovoj knjizi ćemo se baviti tim pitanjima. Nakon što naučite da koristite alate i upoznate razvojno okruženje, istražićete osnovne prakse izgradnje Android aplikacija. Obuhvatićemo širok spektar razvojnih izazova sa kojima se suočavaju programeri i istražićemo različite tehnike za njihovo prevazilaženje.

U ovom poglavlju ćete naučiti da kreirate osnovni Android projekat i da mu dodate funkcije. Upoznaćete sveobuhvatno razvojno okruženje, Android Studio, i učićete o osnovnim oblastima softvera koji će vam omogućiti da budete produktivni.

Android Studio obezbeđuje sve alate za razvoj aplikacija, ali ne i znanje. U ovom prvom poglavlju ćemo vas voditi kroz efikasno korišćenje softvera za izgradnju aplikacije i konfigurisanje najčešćih oblasti Android projekta.

Ovim poglavljem su obuhvaćene sledeće teme:

- Kreiranje Android projekta pomoću integrisanog razvojnog okruženja Android Studio

- Podešavanje virtuelnog uređaja i pokretanje aplikacije
- Android Manifest
- Korišćenje alatke Gradle za izgradnju, konfigurisanje i upravljanje zavisnostima aplikacija
- Struktura Android aplikacije

Tehnički zahtevi

Kompletan kod za sve vežbe i aktivnosti u ovom poglavlju dostupan je u GitHub skladištu, na adresi `https://packt.link/96l1D`

Kreiranje Android projekta pomoću integrisanog razvojnog okruženja Android Studio

Da biste bili produktivni u izgradnji Android aplikacija, neophodno je da znate da koristite **Android Studio**. To je zvanično **integrisano razvojno okruženje** (**integrated development environment - IDE**) za razvoj Android aplikacija, izgrađeno na JetBrains-ovom **IntelliJ IDEA IDE-u** i razvijeno od strane Android Studio tima u Google-u. Koristićete ga tokom ovog kursa za izgradnju aplikacija i postepeno dodavanje naprednijih funkcija.

Razvoj integrisanog razvojnog okruženja Android Studio pratio je razvoj IntelliJ IDEA IDE-a. Naravno, prisutne su osnovne funkcije IDE-a, što vam omogućava da optimizujete kod pomoću predloga, prečica i standardnog refaktorisanja. Programski jezik koji ćete koristiti tokom ovog kursa za izgradnju Android aplikacija je Kotlin. Ranije je standardan jezik za kreiranje Android aplikacija bio Java.

Od Google I/O 2017 (godišnja Google konferencija za programere), to je Google-ov preferirani jezik za razvoj Android aplikacija. Ono što Android Studio zaista izdvaja od drugih razvojnih okruženja za Android je to što je jezik **Kotlin** kreirala JetBrains, kompanija koja je kreirala i IntelliJ IDEA, softver na kom je izgrađen Android Studio. Stoga, možete imati koristi od uspostavljene i napredne, prvoklasne podrške za Kotlin.

Kotlin je kreiran da reši neke od nedostataka jezika Java, u smislu opširnosti, rukovanja null tipovima i dodavanja funkcionalnijih tehnika programiranja, pored mnogih drugih problema. Pošto je Kotlin preferirani jezik za razvoj Android aplikacija od 2017. godine, kada je zamenio jezik Java, koristićete ga u ovoj knjizi.

Upoznavanje razvojnog okruženja Android Studio omogućiće vam da se osećate samouvereno u radu i izgradnji Android aplikacija. Počnimo kreiranje prvog projekta.

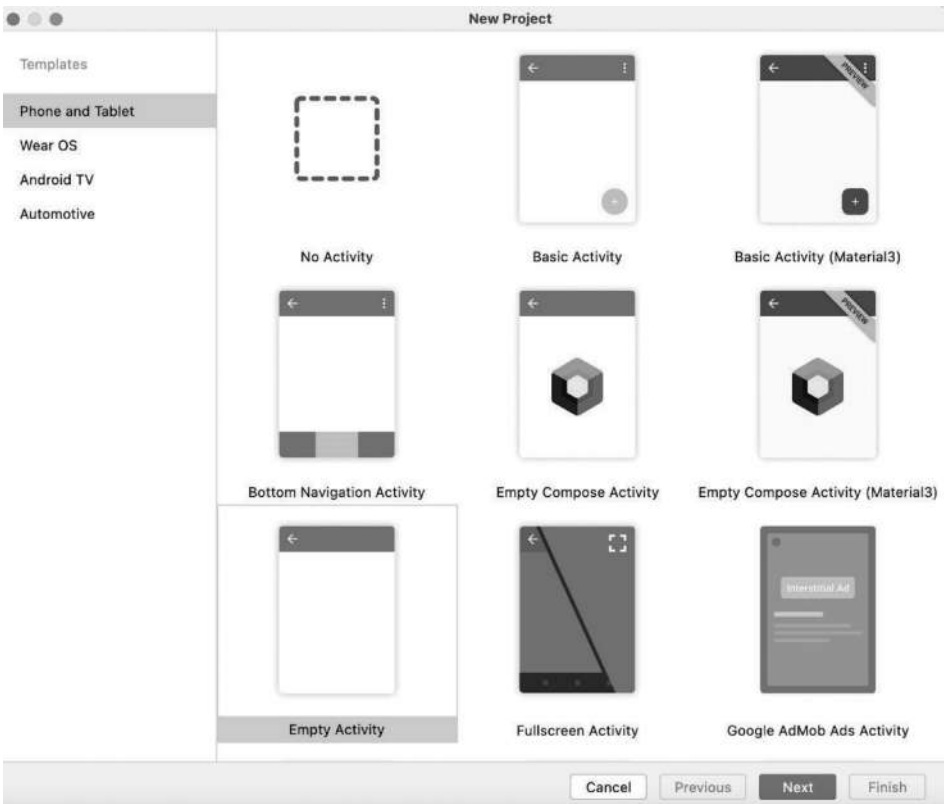
> **Napomena**
>
> Instalacija i podešavanje za Android Studio su opisani u *Predgovoru*. Završite te korake pre nego što nastavite dalje.

Vežba 1.01 – kreiranje Android Studio projekta za vašu aplikaciju

Ovo je početna tačka kreiranja strukture projekta na kojoj će biti izgrađena aplikacija. Pristup zasnovan na šablonima će vam omogućiti da kreirate osnovni projekat u kratkom vremenskom roku, postavljajući gradivne blokove koje možete da koristite za razvoj aplikacije.

Da biste završili ovu vežbu, izvršite sledeće korake:

1. Kada otvorite Android Studio, videćete prozor sa pitanjem da li želite da kreirate novi projekat ili da otvorite postojeći. Izaberite opciju **Create New Project**.
2. Ulazite u jednostavan tok vođen čarobnjakom, koji u velikoj meri pojednostavljuje kreiranje vašeg prvog Android projekta. Sledeći ekran koji vidite sadrži veliki broj opcija za početna podešavanja koja želite da vaša aplikacija ima:

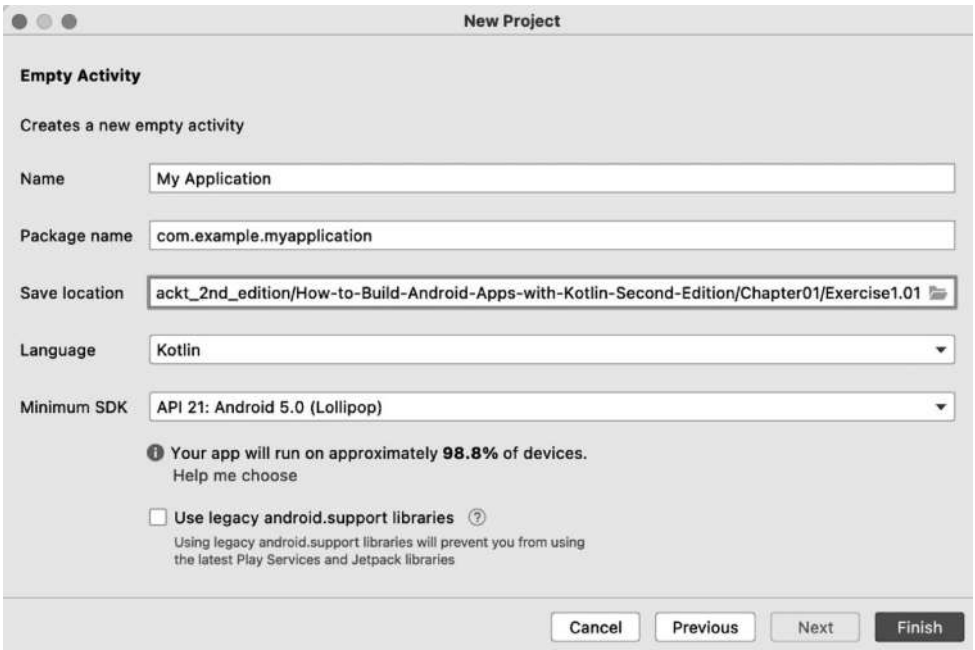


Slika 1.1 – Pokretanje šablona projekta za vašu aplikaciju

3. Dobrodošli u svoj prvi ekosistem za razvoj Android aplikacija. Reč prikazana u većini tipova projekata je *Activity*. U Androidu, Activity je stranica ili ekran. Sve opcije koje možete da izaberete kreiraju ovaj početni ekran drugačije.

   Opisi opisuju kako će izgledati prvi ekran aplikacije. To su šabloni za izgradnju aplikacije. Izaberite **Empty Activity** iz šablona i kliknite na **Next**.

Ekran za konfiguraciju projekta je sledeći:

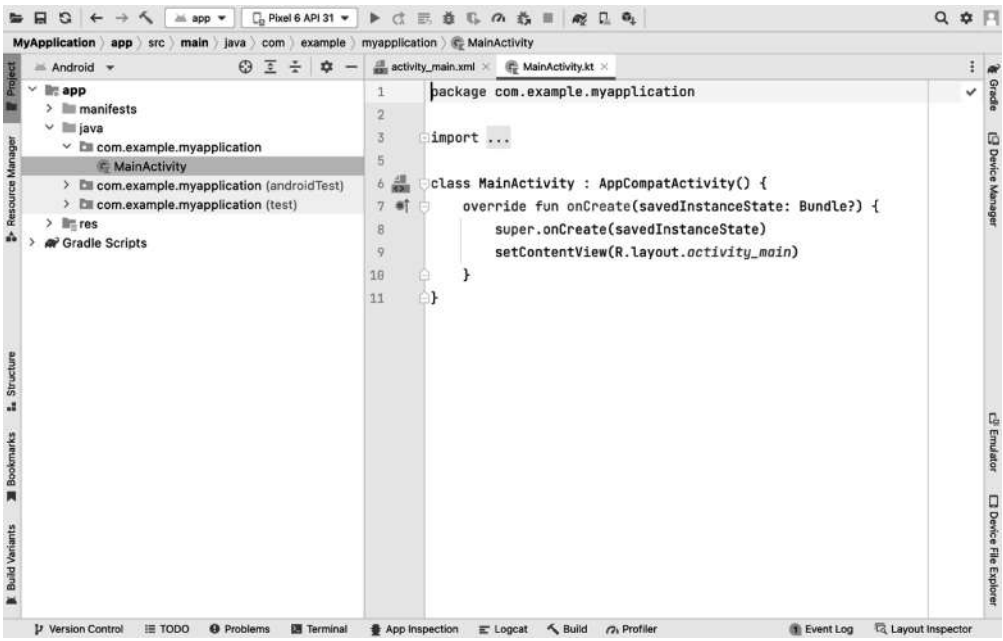


Slika 1.2 – Konfiguracija projekta

4. Na prethodnom ekranu konfigurišete aplikaciju. Pogledajmo sve opcije:
   - **Name**: slično nazivu vašeg Android projekta, ovaj naziv će biti prikazan kao podrazumevani naziv vaše aplikacije kada je instalirana na telefonu i vidljiva u Google Play prodavnici.
   - **Package name**: koristi standardan obrnuti obrazac imena domena za kreiranje naziva. Služi kao identifikator adrese za izvorni kod i elemente u aplikaciji. Najbolje je da ovaj naziv učinite što jasnijim i opisnijim i što je moguće bliži svrsi vaše aplikacije. Zbog toga, verovatno je najbolje da promenite ovaj naziv da biste koristili jedan ili više poddomena (kao što su `com.sample.shop.myshop`). Kao što je prikazano na *slici 1.2*, vrednost **Name** za aplikaciju (malim slovima sa uklonjenim razmacima) dodat je domenu.
   - **Save location**: to je lokalni direktorijum na vašoj mašini u koji će aplikacija inicijalno biti uskladištena. To možete da promenite u budućnosti, tako da verovatno možete da zadržite podrazumevanu vrednost ili da je uredite nešto drugačije (na primer, `Users/MyUser/android/projects`). Podrazumevana lokacija će se razlikovati u zavisnosti od operativnog sistema koji koristite. Podrazumevano, projekat će biti sačuvan u novom direktorijumu sa nazivom aplikacije i uklonjenim razmacima. To dovodi do kreiranja direktorijuma projekta `MyApplication`. Promenite ovaj naziv na `Exercise` (vežbu) ili `Activity` (ekran) na kom radite, pa za ovaj projekat direktorijum nazovite `Exercise1.01`.

- **Language**: **Kotlin** je Google-ov preferirani jezik za razvoj Android aplikacija.
- **Minimum SDK**: u zavisnosti od toga koju verziju razvojnog okruženja Android Studio preuzmete, podrazumevana vrednost može biti ista kao što je prikazano na *slici 1.2* ili drugačija verzija. Neka to ostane isto. Većina novih funkcija Android platforme je kompatibilna sa starijim verzijama, tako da će vaša aplikacija dobro raditi na većini starijih uređaja. Međutim, ako želite da ciljate novije uređaje, trebalo bi da razmislite o podizanju minimalnog API nivoa. Link **Help Me Choose** vodi vas do okvira za dijalog koji objašnjava skup funkcija kom imate pristup, u cilju razvoja na različitim verzijama Android-a, i trenutni procenat uređaja širom sveta koji pokreću svaku verziju Android-a.
- **Use legacy android.support libraries**: ostavite ovu opciju neoznačenu. Koristićete AndroidX biblioteke, koje su zamena za biblioteke podrške dizajnirane da učine funkcije na novijim verzijama Android-a kompatibilnim sa starijim verzijama, ali one pružaju mnogo više od toga. Takođe sadrže nove Android komponente, pod nazivom **Jetpack**, koje, kao što im naziv govori, *poboljšavaju* razvoj Android aplikacija i obezbeđuju mnoštvo bogatih funkcija koje ćete želeti da koristite u svojoj aplikaciji, čime su pojednostavljene uobičajene operacije.

Kada popunite sve ove detalje kliknite na **Finish**. Vaš projekat će biti izgrađen, a zatim će vam biti prikazan sledeći ekran, ili sličan. Na jednoj kartici odmah ćete videti aktivnost koja je kreirana (`MainActivity`), a na drugoj kartici ćete videti raspored koji koristite za ekran (`activity_main.xml`). Direktorijumi strukture aplikacije nalaze se na levom panelu:



Slika 1.3 – Android Studio podrazumevani projekat

U ovoj vežbi prošli ste korake za kreiranje svoje prve Android aplikacije pomoću razvojnog okruženja Android Studio. Ovaj pristup zasnovan na šablonima pokazao vam je osnovne opcije, koje je potrebno da konfigurišete za svoju aplikaciju.

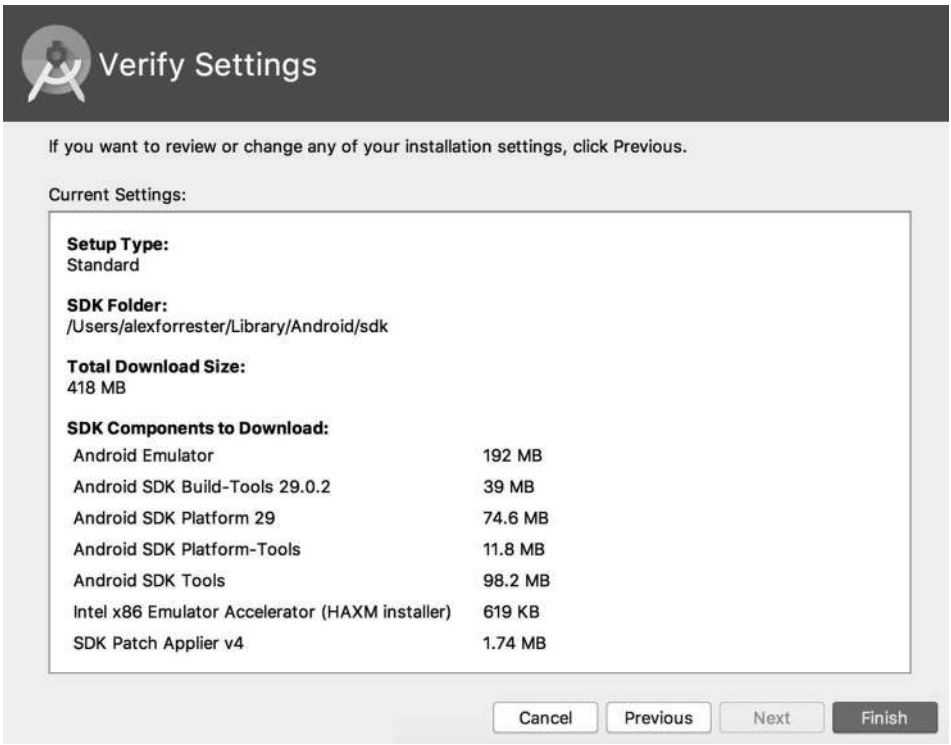
U sledećem odeljku ćete podesiti virtuelni uređaj i videti aplikaciju pokrenutu prvi put.

Podešavanje virtuelnog uređaja i pokretanje aplikacije

Kao deo instaliranja razvojnog okruženja Android Studio, preuzeli ste i instalirali komponente najnovijeg Android **skupa alata za razvoj softvera** (**software development kit - SDK**). SDK obuhvata osnovni emulator koji ćete konfigurisati da biste kreirali virtuelni uređaj za pokretanje Android aplikacija. Emulator oponaša hardverske i softverske funkcije i konfiguraciju pravog uređaja. Prednost je to što možete da unosite promene i brzo da ih pregledate na desktopu, dok razvijate aplikaciju. Iako virtuelni uređaji nemaju sve funkcije stvarnog uređaja, ciklus povratnih informacija je često brži od povezivanja pravog uređaja.

Takođe, iako bi trebalo da obezbedite da vaša aplikacija radi kako se očekuje na različitim uređajima, možete da je standardizujete ciljanjem na specifičan uređaj preuzimanjem profila uređaja, čak i ako nemate pravi uređaj, ako je to zahtev vašeg projekta.

Ekran koji ćete videti (ili nešto slično) kada instalirate Android Studio je sledeći:



Slika 1.4 – SDK komponente

Pogledajmo SDK komponente koje su instalirane i kako se uklapa virtuelni uređaj:

- **Android Emulator**: ovo je osnovni emulator koji ćemo konfigurisati za kreiranje virtuelnih uređaja različitih Android marki i modela.
- **Android SDK Build-Tools**: Android Studio koristi alate za izgradnju za kreiranje aplikacije. Ovaj proces obuhvata kompajliranje, povezivanje i pakovanje aplikacije, da biste je pripremili za instalaciju na uređaj.
- **Android SDK Platform**: ovo je verzija Android platforme koju ćete koristiti za razvoj aplikacije. Platforma se odnosi na API nivo.
- **Android SDK Platform-Tools**: ovo su alati koje možete da koristite, obično iz komandne linije, za interakciju i otklanjanje grešaka u aplikaciji.
- **Android SDK Tools**: za razliku od platformskih alata, ovo su alati koje koristite pretežno iz razvojnog okruženja Android Studio da biste izvršili određene zadatke, kao što su virtuelni uređaj za pokretanje aplikacija i SDK upravljač za preuzimanje i instaliranje platformi i drugih komponenti SDK-a.
- **Intel x86 Emulator Accelerator (HAXM installer)**: ako je vaš OS ima, to je funkcija na nivou hardvera vašeg računara, koju je potrebno da omogućite, da bi emulator radio brže.
- **SDK Patch Applier v4**: kako postaju dostupne novije verzije razvojnog okruženja Android Studio, ovo omogućava primenu zakrpa za ažuriranje verzije koju koristite.

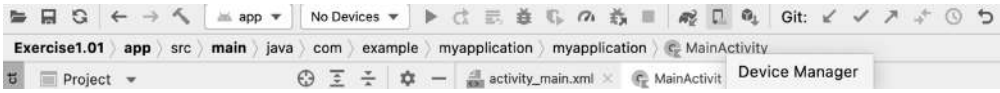
Počnimo sledeću vežbu ovog poglavlja.

Vežba 1.02 – podešavanje virtuelnog uređaja i pokretanje aplikacije na njemu

U *Vežbi 1.01, Kreiranje Android Studio projekta za aplikaciju*, postavili smo Android Studio projekat za kreiranje aplikacije, a sada ćemo je pokrenuti na virtuelnom uređaju. Takođe, možete da pokrenete svoju aplikaciju na pravom uređaju, ali ćete u ovoj vežbi koristiti virtuelni uređaj. Ovaj proces je kontinuirani ciklus tokom rada na aplikaciji. Kada implementirate funkciju možete da potvrdite da izgleda i da se ponaša onako kako je potrebno.

Za ovu vežbu kreiraćete jedan virtuelni uređaj, ali bi trebalo da budete sigurni da pokrećete aplikaciju na više uređaja, da biste potvrdili da li su njen izgled i ponašanje dosledni. Izvršite sledeće korake:

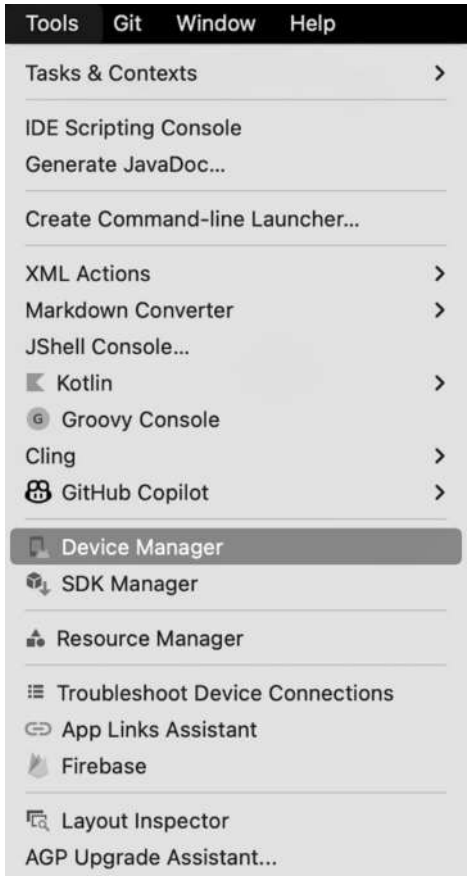
1. Na traci sa alatkama u razvojnom okruženju Android Studio videćete dva padajuća menija, jedan pored drugog, sa unapred izabranim opcijama **app** i **No devices**:



Slika 1.5 – Traka sa alatkama razvojnog okruženja Android Studio

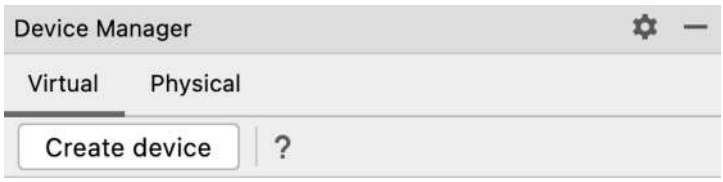
app je konfiguracija aplikacije koju ćemo pokrenuti. Pošto još nismo postavili virtuelni uređaj, izabrana je opcija **No devices**.

2. Da biste kreirali virtuelni uređaj, kliknite na **Device Manager**, kao što je prikazano na *slici 1.5*, da biste otvorili prozor/ekran virtuelnih uređaja. Opciji, kojom to možete da uradite, takođe možete da pristupite iz menija **Tools**:



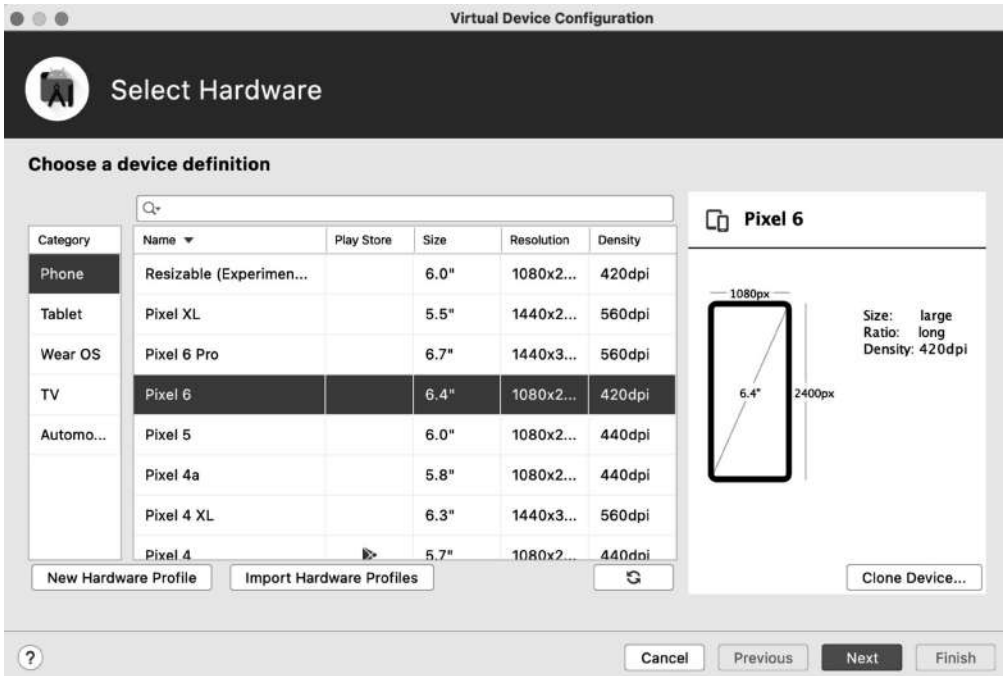
Slika 1.6 – Device Manager u meniju Tools

3. Kliknite na dugme ili opciju na traci sa alatkama da biste otvorili **Device Manager** prozor i kliknite na dugme **Create device**, kao što je prikazano na *slici 1.7*:



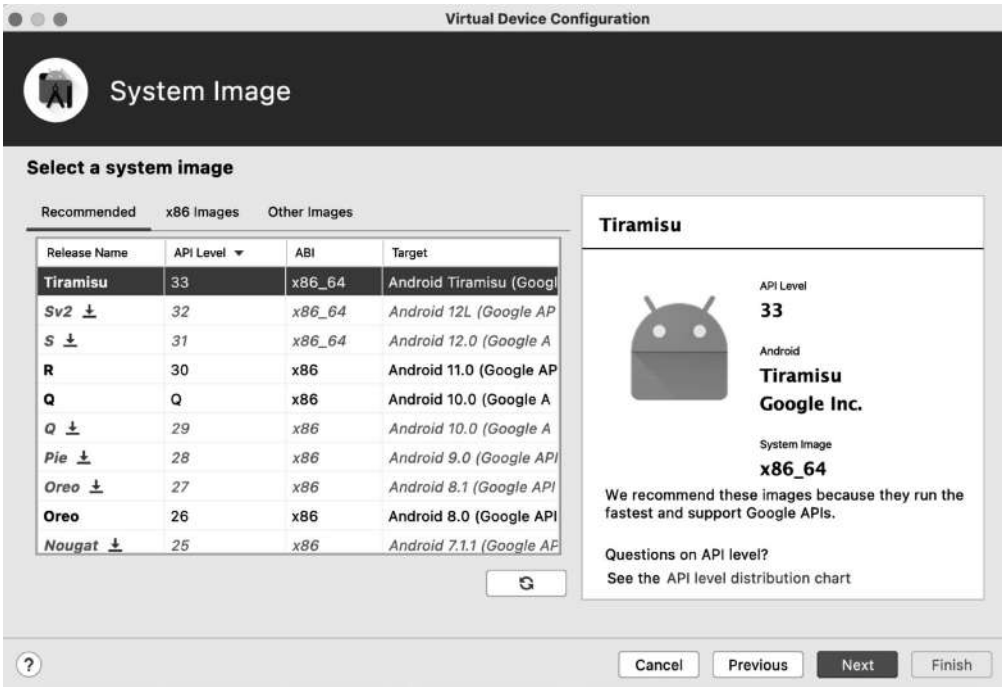
Slika 1.7 – Prozor Device Manager

Videćete ekran kao na *slici 1.8*:



Slika 1.8 – Kreiranje definicije uređaja

4. Izabraćemo uređaj **Pixel 6**. Pravi (nevirtuelni uređaj) Pixel asortiman uređaja je razvio Google i ima pristup najsavremenijim verzijama Android platforme. Kada izaberete uređaj, kliknite na dugme **Next**:



Slika 1.9 – Imidž sistema

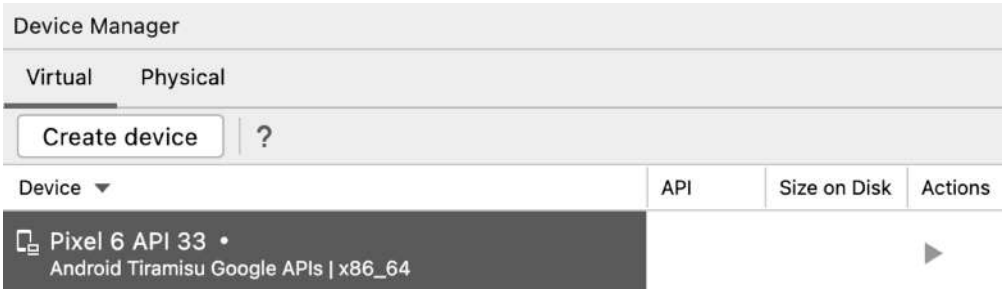
Naziv **Tirimasu**, prikazan ovde, je inicijalni kod/naziv izdanja za Android 13. Izaberite najnoviji dostupan imidž sistema. U koloni **Target**, takođe, može da bude prikazan **(Google Play)** ili **(Google APIs)** u nazivu. Google API-ji znače da imidž sistema dolazi sa unapred instaliranim Google Play servisima.

To je bogat skup funkcija Google API-ja i Google aplikacija, koje vaša aplikacija može da koristi i sa kojima može da komunicira. Kada prvi put pokrenete aplikaciju, videćete aplikacije kao što su Maps i Chrome, umesto obične slike emulatora. Imidž sistema Google Play znači da će, pored Google API-ja, biti instalirana i aplikacija Google Play.

5. Trebalo bi da razvijete svoju aplikaciju na najnovijoj verziji Android platforme, da biste iskoristili najnovije funkcije. Kada prvi put kreirate virtuelni uređaj, potrebno je da preuzmete imidž sistema. Ako je link **Download** prikazan pored **Release Name**, kliknite na njega i sačekajte da se preuzimanje završi. Izaberite dugme **Next** da biste videli virtuelni uređaj koji ste podesili:

Slika 1.10 – Konfiguracija virtuelnog uređaja

6. Kliknite na **Finish** i vaš virtuelni uređaj će biti kreiran. Videćete istaknut svoj uređaj:



Slika 1.11 – Navedeni virtuelni uređaji

7. Pritisnite dugme sa strelicom za reprodukciju ispod kolone **Actions** da biste pokrenuli virtuelni uređaj:

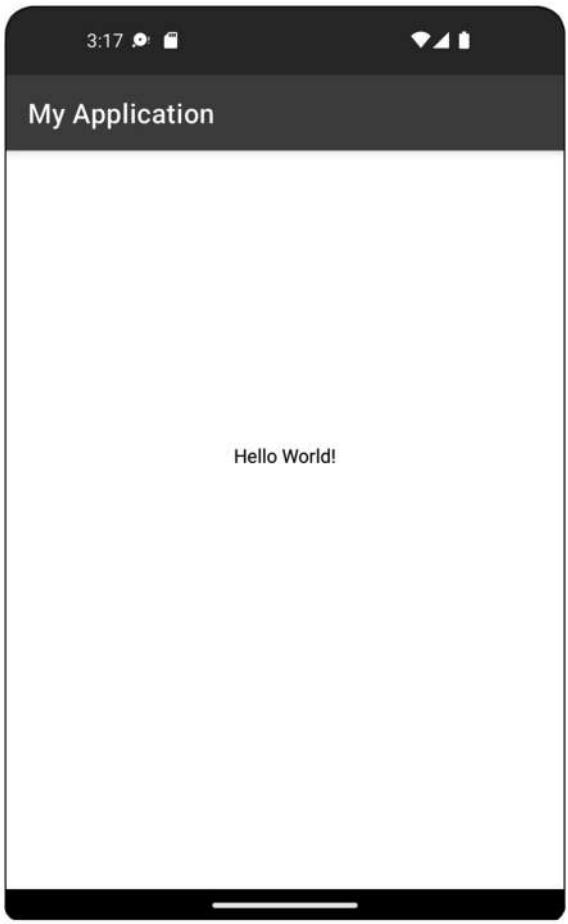
Slika 1.12 – Pokrenut virtuelni uređaj

Tada ćete videti virtuelni uređaj koji je pokrenut u okviru razvojnog okružanja Android Studio, u prozoru alatke **Emulator**. Sada, kada ste kreirali virtuelni uređaj i on radi, možete da se vratite u Android Studio da biste pokrenuli svoju aplikaciju.

8. Izabran je virtuelni uređaj koji ste podesili i pokrenuli. Pritisnite zeleni trougao/dugme za reprodukciju da biste pokrenuli aplikaciju:

Slika 1.13 – Konfiguracija pokretanja aplikacije

Aplikacija je učitana u emulator, kao što je prikazano na *slici 1.14*.



Slika 1.14 – Aplikacija pokrenuta na virtuelnom uređaju

U ovoj vežbi prošli ste korake za kreiranje virtuelnog uređaja i za pokretanje aplikacije, koju ste kreirali, na virtuelnom uređaju. Android Virtual Device Manager, koji ste koristili za ovu vežbu, omogućava vam da kreirate uređaj (ili opseg uređaja) za koji želite da namenite svoju aplikaciju. Pokretanje aplikacije na virtuelnom uređaju omogućava brzi ciklus povratnih informacija, da biste proverili kako se razvoj nove funkcije ponaša i da li je prikazana onako kako očekujete.

Sada ćete istražiti `AndroidManifest.xml` fajl vašeg projekta, koji sadrži informacije i konfiguraciju vaše aplikacije.

Android manifest

Aplikacija koju ste upravo kreirali, iako jednostavna, obuhvata osnovne gradivne blokove, koje ćete koristiti u svim projektima koje kreirate. Aplikacija je pokrenuta iz `AndroidManifest.xml` fajla, koji sadrži detalje o sadržaju vaše aplikacije. Nalazi se na putanji `app|manifests|AndroidManifest.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android=
    "http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_
              rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplication"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.
                  MAIN" />
                <category android:name="android.intent.
                  category.LAUNCHER" />
            </intent-filter>
```

```
        </activity>
    </application>
</manifest>
```

Tipičan manifest fajl, uopšteno govoreći, je fajl najvišeg nivoa, koji opisuje obuhvaćene fajlove ili druge podatke i povezane metapodatke koji čine grupu ili jedinicu. Android manifest primenjuje ovaj koncept na vašu Android aplikaciju kao XML fajl.

Svaka Android aplikacija ima klasu aplikacije koja vam omogućava da konfigurišete aplikaciju. Nakon otvaranja elementa `<application>`, definišite komponente aplikacije. Pošto smo upravo kreirali aplikaciju, ona sadrži samo prvi ekran prikazan u sledećem kodu:

```
<activity android:name=".MainActivity">
```

XML čvor potomak je sledeći:

```
<intent-filter>
```

Android koristi intente kao mehanizam za interakciju sa aplikacijama i komponentama sistema. Intenti su poslati, a filter intenta registruje sposobnost aplikacije da reaguje na te intente. `<android.intent.action.MAIN>` je glavna ulazna tačka u vašu aplikaciju, koja, kako je prikazana u priloženom XML fajlu `.MainActivity`, određuje da će ovaj ekran biti pokrenut kada je aplikacija pokrenuta. `Android.intent.category.LAUNCHER` navodi da će vaša aplikacija biti prikazana u pokretačkom programu uređaja korisnika.

Pošto ste kreirali svoju aplikaciju iz šablona, ona ima osnovni manifest koji će pokrenuti aplikaciju i prikazati početni ekran pri pokretanju pomoću komponente `Activity`. U zavisnosti od toga koje druge funkcije želite da dodate svojoj aplikaciji, možda će biti potrebno da dodate dozvole u fajl Android manifest.

Dozvole su grupisane u tri različite kategorije: normalne, potpisane i opasne:

- **Normalne**: ove dozvole obuhvataju pristup stanju mreže, Wi-Fi mreži, internetu i Bluetooth-u. One su, obično, dozvoljene bez traženja saglasnosti korisnika u toku rada.
- **Potpisane**: ove dozvole deli ista grupa aplikacija koje moraju da budu potpisane istim sertifikatom. To znači da ove aplikacije mogu slobodno da dele podatke, ali druge aplikacije ne mogu da dobiju pristup.
- **Opasne**: ove dozvole su usredsređene na korisnika i njegovu privatnost, kao što je slanje SMS poruka, pristup nalozima i lokaciji i čitanje i pisanje u sistem fajlova i kontakte.

Ove dozvole moraju da budu navedene u manifestu, a u slučaju opasnih dozvola, od Android Marshmallow API 23 (Android 6 Marshmallow) pa nadalje, takođe, potrebno je da tražite od korisnika da odobri dozvole tokom izvršavanja.

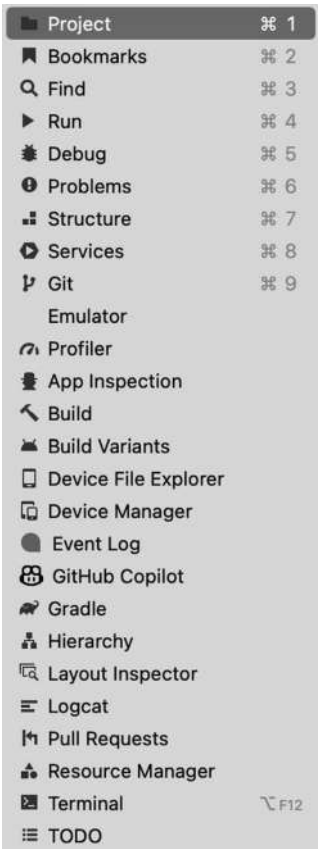
U sledećoj vežbi konfigurisaćemo Android manifest. Detaljnu dokumentaciju o ovom fajlu možete pronaći na adresi `https://developer.android.com/guide/topics/manifest/manifest-intro`.

Vežba 1.03 – konfigurisanje internet dozvole Android manifesta

Ključna dozvola, koju većina aplikacija zahteva, je pristup internetu. To, podrazumevano, nije dodato. U ovoj vežbi ćemo to ispraviti i, u procesu, učitati `WebView`, koji omogućava aplikaciji da prikazuje veb stranice. Ovaj slučaj upotrebe je veoma čest u razvoju Android aplikacija, jer će većina komercijalnih aplikacija prikazati politiku privatnosti, uslove i odredbe i tako dalje. Pošto su ovi dokumenti verovatno zajednički za sve platforme, uobičajeni način za njihovo prikazivanje je učitavanje veb stranice. Da biste to uradili, izvršite sledeće korake:

1. Kreirajte novi Android Studio projekat, kao što ste to uradili u *Vežbi 1.01*, *Kreiranje Android Studio projekta za vašu aplikaciju*.
2. Prebacite kartice na klasu `MainActivity`. Iz glavnog prozora projekta, klasa se nalazi na putanji `app|java|com|example|myapplication`.

Možete da promenite šta će biti prikazano u prozoru projekta tako što ćete otvoriti prozor **Tool** izborom opcije **View|Tool Windows|Project** – biće selektovan prikaz **Project**. Padajuće opcije na vrhu prozora **Project** vam omogućavaju da promenite način prikaza projekta, pri čemu su najčešće korišćeni ekrani **Project** i **Android**:



Slika 1.15 – Padajući meni Tools Windows

Prilikom otvaranja klase `MainActivity`, videćete da ima sledeći sadržaj, ili sličan:

```
package com.example.myapplication

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?)
    {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }
}
```

Detaljnije ćete ispitati sadržaj ovog fajla u sledećem odeljku ovog poglavlja, ali za sada je potrebno samo da budete svesni da iskaz `setContentView (R.layout.activity_main)` postavlja raspored korisničkog interfejsa, koji ste videli kada ste prvi put pokrenuli aplikaciju na virtuelnom uređaju.

3. Koristite sledeći kod da biste izvršili sledeće promene:

```
package com.example.myapplication

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.webkit.WebView

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?){
        super.onCreate(savedInstanceState)
        val webView = WebView(this)
        webView.settings.javaScriptEnabled = true
        setContentView(webView)
        webView.loadUrl("https://www.google.com")
    }
}
```

Dakle, zamenjujete fajl rasporeda prikazom `WebView`. Ključna reč `val` je referenca read-only svojstva, koja ne može da bude promenjena kada je postavljena. Potrebno je da bude omogućen JavaScript u komponenti `WebView` za izvršenje JavaScript-a.

Napomena

Ne postavljamo tip, ali Kotlin sadrži zaključivanje tipa, tako da će zaključiti tip ako je moguće. Dakle, eksplicitno navođenje tipa pomoću iskaza `val webView: WebView = WebView(this)` nije neophodno. U zavisnosti od toga koje ste programske jezike koristili u prošlosti, redosled definisanja naziva i tipa parametra može vam biti poznat ili ne. Kotlin prati Pascal notaciju, odnosno, naziv praćen tipom.

4. Sada pokrenite aplikaciju i biće prikazan tekst, kao na sledećem snimku ekrana:

Slika 1.16 – Poruka o grešci bez dozvole za internet

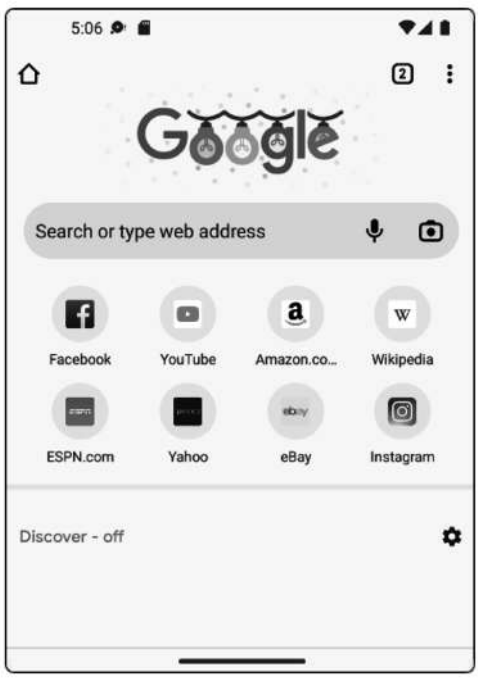
5. Ova greška se javlja jer `INTERNET` dozvola nije dodata fajlu `AndroidManifest.xml`. (Ako dobijete grešku `net::ERR_CLEARTEXT_NOT_PERMITTED`, to je zato što URL koji učitavate u `WebView` nije HTTPS, a saobraćaj koji nije HTTPS je onemogućen sa API nivoa 28, za Android 9.0 Pie i novije verzije).
6. Ispravimo to dodavanjem `INTERNET` dozvole u manifest. Otvorite Android manifest i iznad oznake `<application>` dodajte sledeće:

```
<uses-permission android:name="android.permission.
INTERNET" />
```

   Kompletan Android manifest fajl, zajedno sa dozvolom, možete naći na adresi :`https://packt.link/smzpl`

   Deinstalirajte aplikaciju sa virtuelnog uređaja pre nego što je ponovo pokrenete. Potrebno je to da uradite, jer dozvole aplikacije ponekad mogu da budu keširane.

   Uradite to dugim pritiskom na ikonicu aplikacije i odabirom opcije **App Info**, koja će biti prikazana, a zatim pritisnite ikonicu Bin ispod koje se nalazi tekst **Uninstall**. Alternativno, dugo pritisnite ikonicu aplikacije, a zatim je prevucite na ikonicu Bin pored koje se nalazi tekst **Uninstall** u gornjem desnom uglu ekrana.
7. Ponovo instalirajte aplikaciju i videćete da je veb stranica prikazana u prikazu `WebView`:



Slika 1.17 – Aplikacija koja prikazuje WebView

U ovom primeru ste naučili da dodate dozvolu u manifest. Android Manifest možete posmatrati kao sadržaj aplikacije. Navodi sve komponente i dozvole koje aplikacija koristi. Kao što ste videli pokretanjem aplikacije iz pokretačkog programa, takođe pruža ulazne tačke u aplikaciju.

U sledećem odeljku ćete istražiti Android sistem za izgradnju, koji koristi alatku za izgradnju Gradle, za pokretanje aplikacije.

Korišćenje alatke Gradle za izgradnju, konfigurisanje i upravljanje zavisnostima aplikacija

Tokom kreiranja ovog projekta, uglavnom ste koristili SDK za Android platformu. Potrebne Android biblioteke su preuzete kada ste instalirali Android Studio. Međutim, to nisu jedine biblioteke za izgradnju vaše aplikacije. Da biste konfigurisali i izgradili Android projekat ili aplikaciju, koristićete alatku za izgradnju, pod nazivom **Gradle**.

Gradle je višenamenski alat za izgradnju, koji Android Studio koristi za izgradnju aplikacije. Android Studio, podrazumevano, koristi Groovy, dinamički tipiziran **Java virtual machine** (**JVM**) jezik za konfigurisanje procesa izgradnje i omogućavanje jednostavnog upravljanja zavisnostima, tako da možete da dodajete biblioteke svom projektu i da specifikujete verzije.

Android Studio, takođe, može da bude konfigurisan da koristi Kotlin za konfigurisanje izgradnje, ali pošto je podrazumevani jezik Groovy, vi ćete ga koristiti. Fajlovi u kojima se čuvaju ove informacije o izgradnji i konfiguraciji nazvani su `build.gradle`.

Kada prvi put kreirate aplikaciju postoje dva fajla `build.gradle`, jedan na osnovnom/najvišem nivou projekta i jedan specifičan za vašu aplikaciju, u direktorijumu aplikacije `modul`.

Fajl build.gradle na nivou projekta

Pogledajmo sada fajl `build.gradle` na nivou projekta. Ovde postavljate sva osnovna podešavanja projekta, koja mogu da budu primenjena na podmodule/projekte:

```
plugins {
    id 'com.android.application' version '7.4.2' apply
    false
    id 'com.android.library' version '7.4.2' apply false
    id 'org.jetbrains.kotlin.android' version '1.8.0'
    apply false
}
```

Gradle radi na sistemu dodatnih modula (eng. plugin), tako da možete da napišete sopstveni dodatni modul, koji obavlja zadatak ili niz zadataka i da ga priključite u pipeline izgradnje. Tri prethodno navedena dodatna modula izvršavaju sledeće:

- `com.android.application`: dodaje podršku za kreiranje Android aplikacije
- `com.android.library`: omogućava da potprojekti/moduli budu Android biblioteke
- `org.jetbrains.kotlin.android`: obezbeđuje integraciju i jezičku podršku za Kotlin u projektu

Iskaz `apply false` omogućava ove dodatne module samo za potprojekte/module, a ne i za osnovni nivo projekta. Iskaz `version ,7.3.1‘` specifikuje verziju dodatnog modula, koja je primenjena na sve potprojekte/module.

Fajl build.gradle na nivou aplikacije

Aplikacija `build.gradle` je specifična za konfiguraciju projekta:

```
plugins {
    id 'com.android.application'
    id 'org.jetbrains.kotlin.android'
}
android {
    namespace 'com.example.myapplication'
    compileSdk 33
    defaultConfig {
        applicationId "com.example.myapplication"
        minSdk 24
        targetSdk 33
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner
            "androidx.test.runner.AndroidJUnitRunner"}
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-
               android-optimize.txt'), 'proguard-rules.pro'
        }
    }
```

```
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
    kotlinOptions {
        jvmTarget = '1.8'
    }
}
dependencies {…}
```

Dodatni moduli za Android i Kotlin, detaljno opisani u osnovnom fajlu `build.gradle`, ovde su primenjeni na vaš projekat pomoću ID-ja u linijama `plugins`.

U bloku `android`, koji je obezbeđen dodatnim modulom `com.android.application`, konfigurišete podešavanja specifična za Android:

- `namespace`: postavljen je iz naziva paketa koji ste specifikovali prilikom kreiranja projekta. Služi za generisanje identifikatora izgradnje i resursa.
- `compileSdk`: služi za definisanje API nivoa kojim je aplikacija kompajlirana, a aplikacija može da koristi funkcije ovog API-ja i nižeg.
- `defaultConfig`: osnovna konfiguracija vaše aplikacije.
- `applicationId`: postavljen na paket aplikacije i to je identifikator aplikacije, koji koristite u Google Play prodavnici za jedinstvenu identifikaciju aplikacije. Ukoliko je potrebno, može da bude promenjen da bi se razlikovao od naziva paketa.
- `minSdk`: minimalni API nivo koji aplikacija podržava. Filtrira aplikaciju tako da ne bude prikazana u Google Play prodavnici za uređaje koji su niži od ovog nivoa.
- `targetSdk`: API nivo na koji ciljate. To je API nivo za koji je namenjena izgrađena aplikacija i kojim je testirana.
- `versionCode`: Specifikuje kod verzije aplikacije. Svaki put kada je potrebno izvršiti ažuriranje aplikacije, kod verzije je potrebno povećati za jedan ili više.
- `versionName`: naziv verzije prilagođen korisniku, koji obično prati semantičko kreiranje verzija *X.Y.Z*, gde je *X* glavna verzija, *Y* je manja verzija a *Z* je verzija zakrpe, na primer, *1.0.3*.
- `testInstrumentationRunner`: pokretač testiranja koji ćete koristiti za UI testove.

- `buildTypes`: pod opcijom `buildTypes` dodato je izdanje koje konfiguriše aplikaciju za kreiranje verzije `release`. Ako je vrednost `minifyEnabled` podešena na `true`, veličina aplikacije će biti smanjena uklanjanjem neiskorišćenog koda i maskiranjem aplikacije. Ovaj korak maskiranja menja naziv referenci izvornog koda na vrednosti kao što su `a.b.c()`. To kod čini manje sklonim obrnutom inženjeringu i dodatno smanjuje veličinu izgrađene aplikacije.
- `compileOptions`: nivo jezika Java izvornog koda (`sourceCompatibility`) i byte koda (`targetCompatibility`).
- `kotlinOptions`: `jvm` biblioteka, koju bi dodatni modul `kotlin gradle` trebalo da koristi.

Blok `dependencies` specifikuje biblioteke koje aplikacija koristi povrh SDK-a Android platforme, kao što je prikazano u sledećem kodu (sa dodatnim komentarima):

```
dependencies {
// Kotlin extensions, jetpack component with Android
   Kotlin language features
implementation 'androidx.core:core-ktx:1.7.0'
// Provides backwards compatible support libraries and
   jetpack components
implementation 'androidx.appcompat:appcompat:1.6.1'
// Material design components to theme and style your
   app
implementation
    'com.google.android.material:material:1.8.0'
// The ConstraintLayout ViewGroup updated separately
   from main Android sources
implementation
    'androidx.constraintlayout:constraintlayout:2.1.4'
// Standard Test library for unit tests
testImplementation 'junit:junit:4.13.2'
// UI Test runner
androidTestImplementation
    'androidx.test.ext:junit:1.1.5'
// Library for creating Android UI tests
androidTestImplementation
    'androidx.test.espresso:espresso-core:3.5.1'
}
```

Zavisnosti prate Maven **Project Object Model** (**POM**) konvenciju za `groupId`, `artifactId` i `versionId`, odvojene dvotačkom (`:`). Na primer, ranije navedena kompatibilna biblioteka podrške je prikazana kao:

```
'androidx.appcompat:appcompat:1.6.1'
```

`groupId` je `android.appcompat`, `artifactId` je `appcompat`, a `versionId` je `1.5.1`. Sistem izgradnje locira i preuzima ove zavisnosti za izgradnju aplikacije iz bloka `repositories`, koji je detaljno opisan u fajlu `settings.gradle`, koji je objašnjen u sledećem odeljku.

> **Napomena**
>
> Verzije zavisnosti navedene u prethodnom odeljku koda i u sledećim odeljcima ovog i drugih poglavlja, podložne su promenama i biće ažurirane, tako da će verovatno biti veće kada budete kreirali ove projekte.

Notacija `implementation` za dodavanje ovih biblioteka znači da njihove interne zavisnosti neće biti izložene vašoj aplikaciji, što ubrzava kompajliranje.

Komponente `androidx` su ovde dodate kao zavisnosti, a ne u izvoru Android platforme, da bi mogle da budu ažurirane nezavisno od Android verzija. `androidx` sadrži paket Android Jetpack biblioteka i ponovo upakovanu biblioteku podrške.

Sledeći Gradle fajl koji je potrebno da ispitate je `settings.gradle`, koji inicijalno izgleda ovako:

```
pluginManagement {
    repositories {
        google()
        mavenCentral()
        gradlePluginPortal()
    }
}
dependencyResolutionManagement {
repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories {
        google()
        mavenCentral()
    }
}
rootProject.name = "My Application"
include ':app'
```

Prilikom prvog kreiranja projekta pomoću razvojnog okruženja Android Studio, postoji samo jedan modul, `app`, ali kada dodate još funkcija možete da dodate nove module, koji su namenjeni da sadrže izvor neke funkcije, a ne da bude upakovana u glavni modul `app`.

Te module nazivamo **moduli funkcija**, a možete da ih dopunite drugim tipovima modula, kao što su deljeni moduli koje koriste svi ostali moduli, kao što je mrežni modul. Ovaj fajl, takođe, sadrži skladišta dodatnih modula i zavisnosti, koje možete da preuzmete u posebne blokove za dodatne module i zavisnosti.

Podešavanje vrednosti `RepositoriesMode.FAIL_ON_PROJECT_REPOS` osigurava da su ovde definisana sva skladišta zavisnosti; u suprotnom, generisana je greška u izgradnji.

Vežba 1.04 – istraživanje kako je Material Design upotrebljen za temu aplikacije

U ovoj vežbi ćete učiti o Google-ovom novom jeziku za projektovanje, **Material Design** i upotrebićete ga za učitavanje aplikacije sa temom Material Design. Material Design je jezik za projektovanje koji je kreirao Google, a dodaje obogaćene elemente korisničkog interfejsa zasnovane na efektima iz stvarnog sveta, kao što su osvetljenje, dubina, senke i animacije. Izvršite sledeće korake da biste završili vežbu:

1. Kreirajte novi Android Studio projekat, kao što ste to uradili u *Vežbi 1.01*, *Kreiranje Android Studio projekta za aplikaciju*.
2. Prvo, pogledajte blok `dependencies` i pronađite Material Design zavisnost:

```
implementation
'com.google.android.material:material:1.8.0'
```

3. Zatim, otvorite fajl `themes.xml`, koji se nalazi na putanji `app|src|main|res|values|themes.xml`: Fajl `themes.xml` takođe postoji i u direktorijumu `values-night`, koji koristite za tamni režim, koji ćemo kasnije istražiti:

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.MyApplication" parent="Theme.
        MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/purple_500
            </item>
        <item name="colorPrimaryVariant">@color/
            purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
```

```
        <item name="colorSecondary">@color/teal_200
            </item>
        <item name="colorSecondaryVariant">@color/
            teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/
            colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>
</resources>
```

Roditeljska klasa klase `Theme.MaterialComponents.DayNight.DarkActionBar` je `Theme.MyApplication`.

Material Design zavisnost, koja je dodata u blok `dependencies`, ovde služi za primenu teme aplikacije. Jedna od ključnih razlika koje nudi **Material Design Components** (**MDC**) u odnosu na `AppCompat` teme koje su im prethodile, je mogućnost obezbeđivanja varijacija primarnih i sekundarnih boja vaše aplikacije.

Na primer, `colorPrimaryVariant` omogućava vam da dodate nijansu primarnoj boji, koja može da bude svetlija ili tamnija od boje `colorPrimary`. Pored toga, možete da stilizujete boje elemenata prikaza u prvom planu aplikacije pomoću opcije `colorOnPrimary`.

Ovo zajedno donosi kohezivno brendiranje za temu vaše aplikacije. Da biste videli kako to funkcioniše, izvršite sledeće promene da biste invertovali primarnu i sekundarnu boju:

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.MyApplication" parent="Theme.
        MaterialComponents.DayNight.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/teal_200</item>
        <item name="colorPrimaryVariant">@color/
            teal_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/purple_200
            </item>
        <item name="colorSecondaryVariant">@color/
```

```
            purple_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor">?attr/
            colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>
</resources>
```

4. Pokrenite aplikaciju i videćete da je tema aplikacije drugačija. Traka akcije i statusna traka imaju promenjenu boju pozadine u odnosu na podrazumevanu aplikaciju sa temom Material, kao što je prikazano na *slici 1.18*:

Slika 1.18 – Aplikacija sa invertovanim primarnim i sekundarnim bojama

U ovoj vežbi ste naučili kako Material Design može da bude upotrebljen za dodavanje teme aplikaciji. Pošto trenutno prikazujete samo `TextView` na ekranu, nije jasno koje prednosti pruža Material Design, ali to će se promeniti kada počnete više da koristite Material UI vidžete za dizajn.

Sada, kada ste naučili kako je projekat izgrađen i konfigurisan, u sledećem odeljku ćete detaljno istražiti strukturu projekta, naučiti kako je kreiran i upoznati se sa osnovnim oblastima razvojnog okruženja.

Struktura Android aplikacije

Sada kada smo opisali kako funkcioniše alat za izgradnju Gradle, istražićemo ostatak projekta. Najjednostavniji način da to uradite je da ispitate strukturu direktorijuma aplikacije. U gornjem levom uglu okruženja Android Studio nalazi se prozor sa alatkama, pod nazivom **Project**, koji vam omogućava da pregledate sadržaj aplikacije.

Podrazumevano, prozor je podešen na **open**/**selected** kada je Android projekat prvi put kreiran. Kada selektujete projekat, videćete prikaz sličan snimku ekrana na *slici 1.19*. Ako ne vidite nijednu traku prozora na levoj strani ekrana, u gornjoj traci sa alatkama selektujte opciju **View | Appearance | Tool Window Bars** i uverite se da je označena.

Postoji mnogo različitih opcija za pregled projekta, ali će unapred biti selektovan **Android**. Ovaj prikaz uredno grupiše strukturu direktorijuma `app`, pa ćemo je sada pregledati.

Sledi pregled tih fajlova, sa više detalja o najvažnijim. Kada otvorite prikaz videćete da se sastoji od sledeće strukture direktorijuma: