1 početak

Uskočimo



Android je osvojio svet na juriš.

Svi žele smartfon ili tablet, a Android uređaji su silno popularni. U ovoj knjizi, naučićemo vas kako da **pravite vlastite aplikacije**, a počećemo tako što ćete da napravite jednu jednostavnu aplikaciju koju ćete izvršiti na jednom virtualnom Android uređaju (Android Virtual Device). Usput, upoznaćete neke od osnovnih komponenti svih Android aplikacija, kao što su **aktivnosti** (engl. *activities*) i **rasporedi** (engl. *layouts*). **Dovoljno je da malo znate Javu...**

Dobro došli u svet Androida

Android je najpopularnija mobilna platforma na svetu. Po poslednjoj evidenciji, bilo je preko *dve milijarde* aktivnih Android uređaja na celom svetu, a njihov broj rapidno raste.

Android je sveobuhvatna platforma sa otvorenim kodom zasnovana na Linuxu, a podržava je Google. To je moćno razvojno okruženje koje sadrži sve što vam treba za pravljenje silnih aplikacija pomoću mešavine Jave i XML-a. Štaviše, omogućava vam da te aplikacije primenite na raznovrsnim uređajima – telefonima, tabletima i tako dalje.

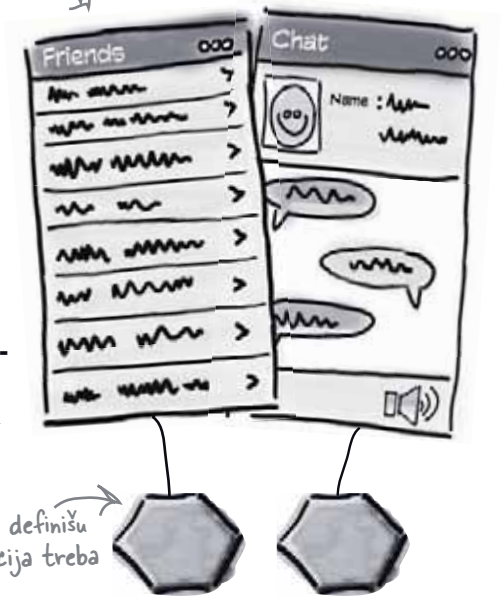
Dakle, od čega se sastoji tipična Android aplikacija?

Naše Android aplikacije ćemo praviti pomoću mešavine Jave i XML-a. Usput ćemo da objašnjavamo, ali da biste mogli da ovu knjigu iskoristite na najbolji načintrebalo bi da dobro razumete Javu.

Rasporedi definišu kako svaki ekran izgleda

Rasporedi kažu Androidu kako će izgledati ekrani u vašoj aplikaciji

Tipična Android aplikacija se sastoji od jednog ekrana ili od više njih. Vi definišete kako svaki ekran izgleda pomoću **rasporeda** (engl. *layout*) koji definiše njegov izgled. Rasporedi se obično definišu pomoću XML-a, a mogu da sadrže GUI komponente (komponente grafičkog korisničkog interfejsa) kao što su dugmad, polja za tekst i natpisi.



Aktivnosti definišu šta aplikacija radi

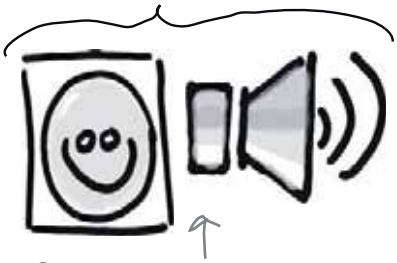
Rasporedi definišu samo *izgled* aplikacije. Vi definišete šta aplikacija *radi* tako što napišete jednu ili više **aktivnosti**. Aktivnost je jedna posebna Java klasa koja se zove **activity** (aktivnost) i koja određuje koji raspored da se upotrebi i kaže aplikaciji kako da reaguje na postupke korisnika. Na primer, ako u rasporedu postoji dugme, vi morate da napišete Java kôd u aktivnosti, da biste definisali šta to dugme treba da uradi kada ga pritisnete.

Aktivnosti definišu šta aplikacija treba da radi

Ponekad su potrebni i dodatni resursi

Pored aktivnosti i rasporeda, Android aplikacijama su često potrebni dodatni resursi kao što su fajlovi slika i podaci aplikacije. Vi možete aplikaciji da dodate sve dodatne fajlove koji su vam potrebni.

Android aplikacije su u stvari samo gomile fajlova u posebnim direktorijima. Kada gradite svoju aplikaciju, svi ti fajlovi se međusobno povezuju, i tako dobijate aplikaciju koju možete da izvršite na svom uređaju.



Resursi mogu da se sastoje od zvučnih i slikovnih fajlova

Seciranje Android platforme

AndroidAndroid platforma se sastoji od niza različitih komponenti. Ona sadrži osnovne aplikacije kao što su kontakti (Contacts), skup API-ja koji vam pomažu da kontrolišete kako vaša aplikacija izgleda i kako se ponaša, i čitavu gomilu pratećih fajlova i biblioteka. Ovde je letimičan pogled na to kako se svi oni uklapaju:



Ne brinite ako vam izgleda da je ovo previše.

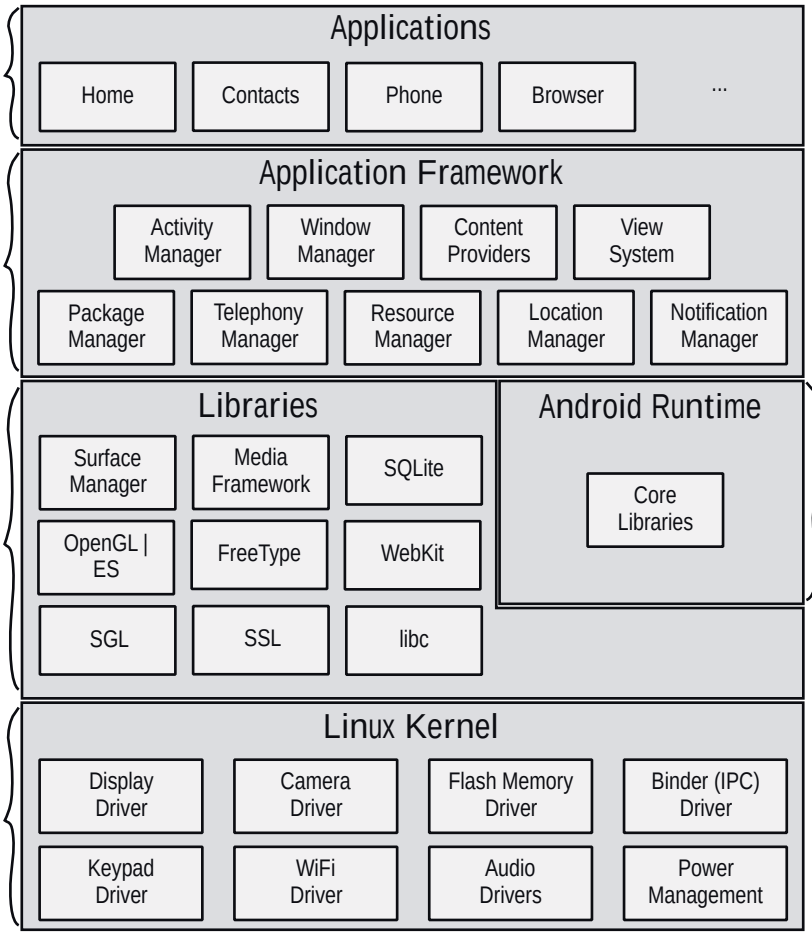
Dajemo vam samo pregled svega što postoji u Android platformi. Objasnićemo različite komponente detaljnije kada bude potrebno.

Android se isporučuje sa skupom osnovnih aplikacija kao što su Contacts, Phone, Calendar i pretraživač.

Kada gradite svoje aplikacije imate pristup istim API-jima (Application Programming Interface – interfejs za programiranje aplikacija) koje koriste osnovne aplikacije. Te API-je koristite da biste kontrolisali kako vaša aplikacija izgleda i kako se ponaša.

Ispod radnog okvira aplikacije nalazi se skup C i C++ biblioteka (engl. *Libraries*). Te biblioteke vam se izlažu putem API-ja radnog okruženja.

Ispod svega ostalog nalazi se jezgro operativnog sistema Linux (engl. *Linux kernel*). Android se oslanja na jezgro za drajvere i za osnovne usluge kao što su bezbednost i upravljanje memorijom.



Odlično je to što se sve moćne Android biblioteke izlažu putem API-ja u radnom okviru aplikacija, a vi koristite te API-je za pravljenje odličnih Android aplikacija. Na početku vam treba samo nešto poznavanja Jave i odlična ideja za aplikaciju.

Android izvršno okruženje (engl. *Android runtime*) sadrži skup biblioteka koje primenjuju većinu programskog jezika Java. Svaka Android aplikacija se izvršava u vlastitom procesu.

Evo šta ćemo da radimo

Dakle, uskočimo i napravimo jednu uprošćenu Android aplikaciju. Treba da uradimo samo nekoliko stvari:

1. **Postavljanje razvojnog okruženja.**
Moramo da instaliramo Android Studio, koji sadrži sve alatke potrebne za razvijanje vaših Android aplikacija.

2. **Izrada proste aplikacije.**
Koristeći Android Studio, napravićemo jednu jednostavnu aplikaciju koja će da prikaže neki uzorak teksta na ekranu.

3. **Izvršavanje aplikacije u emulatoru Androida.**
Upotrebićemo ugrađeni emulator da vidimo kako aplikacija radi.

4. **Menjanje aplikacije.**
Na kraju, malo ćemo da izmenimo aplikaciju koju smo napravili u tački 2, pa ćemo ponovo da je izvršimo.



nema GLUPIH PITANJA

P: Da li se sve Android aplikacije prave u Javi?

O: Android aplikacije mogu da se prave i u drugim jezicima. Većina programera koristi Javu, pa zato to opisujemo u ovoj knjizi.

P: Koliko dobro moram da znam Javu da bih pravio Android aplikacije?

O: Zaista vam je potrebno iskustvo Java SE (Standard Edition). Ako vam treba da osvežite znanje, predlažemo knjigu *Head First Java* pisaca Kathy Sierra i Bert Bates.

P: Da li moram da znam Swing i AWT?

O: Android ne koristi ni Swing ni AWT, pa ne brinite ako nemate iskustva sa grafičkim korisničkim interfejsima Jave.

Vi ste ovde.

- [x] **Postavljanje okruženja**
- [ ] **Izrada aplikacije**
- [ ] **Izvršavanje aplikacije**
- [ ] **Menjanje aplikacije**

Vaše razvojno okruženje

Java je najpopularniji jezik za pravljenje Android aplikacija. Android uređaji ne izvršavaju fajlove tipa *.class* i *.jar*. Umesto toga, da bi se poboljšale brzina i performanse baterije, Android uređaji koriste vlastite optimizovane formate za prevedeni kôd. To znači da ne možete da koristite obično razvojno okruženje za Javu – potrebni su vam i specijalni alati za konvertovanje vašeg prevedenog koda u Android format, za njegovo postavljanje na Android uređaj i za rešavanje problema u radu aplikacije.

Sve se to nalazi u razvojnom kompletu **Android SDK**. Pogledajmo šta tu sve ima.

Android SDK

Razvojni komplet Android SDK (Software Development Kit) sadrži biblioteke i alatke potrebne za pravljenje Android aplikacija. Ovo su neki od glavnih delova:

SDK platforma
Postoji po jedna za svaku verziju Androida.

Dokumentacija
Da biste mogli oflajn da koristite najnoviju API dokumentaciju.

SDK alati
Alati za rešavanje grešaka i za testiranje i drugi korisni uslužni alati. Takođe postoji i skup alatki zavisnih od platforme.



Android podrška
Dodatni API-ji koji ne postoje u standardnoj platformi.

Uzorci aplikacija
Ako su vam potrebni praktični primeri koda da biste lakše shvatili kako da upotrebite neki API, uzorci aplikacija mogu da pomognu.

Google Play Billing
Omogućava vam da u svoju aplikaciju uključite usluge naplaćivanja.

Android Studio je specijalna verzija IntelliJ IDEA

IntelliJ IDEA je jedno od najpopularnijih integrisanih razvojnih IDE okruženja za razvijanje Jave. Android Studio je jedna verzija IDEA okruženja koja sadrži jednu verziju Android SDK i dodatne GUI alatke koje vam pomažu u pravljenju aplikacija.

Osim što vam obezbeđuje editor i pristup alatkama i bibliotekama Android SDK, Android Studio vam daje šablone (engl. *template*) koje možete da koristite kao pomoć u pravljenju novih aplikacija i klasa, i olakšava pakovanje i izvršavanje vaših aplikacija.

Uskočimo

- [ ] **Postavljanje okruženja**
- [ ] **Izrada aplikacije**
- [ ] **Izvršavanje aplikacije**
- [ ] **Menjanje aplikacije**

Instalirajte Android Studio

Pre nego što nastavimo, morate da instalirate Android Studio na svoj računar. Mi u ovu knjigu ne uključujemo instrukcije za instalaciju zato što one mogu brzo da zastare, ali biće u redu ako budete pratili onlajn uputstva.

Mi koristimo Android Studio verziju 2.3. Da biste imali što više koristi od ove knjige, trebalo bi da koristite tu verziju, ili neku noviju.

Najpre ovde proverite sistemske zahteve za Android Studio:

http://developer.android.com/sdk/index.html#Requirements

Google ponekad promeni svoje URL-ove. Ako ovi URL-ovi nisu u redu, izvršite pretraživanje za pojam Android Studio da biste ih pronašli.

Zatim pratite instrukcije za instaliranje Android Studija ovde:

https://developer.android.com/sdk/installing/index.html?pkg=studio

Pošto instalirate Android Studio, otvorite ga i pratite uputstva da biste dodali najnovije SDK alatke i pomoćne biblioteke.

Kada to obavite, trebalo bi da ugledate ekran dobrodošlice u Android Studio. Sada možete da napravite svoju prvu Android aplikaciju.



Ovo je ekran dobrodošlice u Android Studio. On sadrži niz opcija za ono što može da se radi

nema GLUPIH PITANJA

P: Kažete da ćemo koristiti Android Studio kada pravimo Android aplikacije. Da li moram?

O: Strogo govoreći, nije obavezno da se koristi Android Studio kada se prave Android aplikacije. Potrebna vam je samo alatka koja omogućava pisanje i prevođenje Java koda, plus još nekoliko specijalizovanih alatki za konvertovanje prevedenog koda u oblik koji Android uređaji mogu da izvršavaju.

Android Studio je zvanično IDE okruženje za Android, pa Android tim preporučuje da ga koristite, ali mnogi ljudi umesto njega koriste IntelliJ IDEA.

P: Mogu li da pišem Android aplikacije a da ne koristim nijedno IDE okruženje?

O: To je moguće, ali zahteva više posla. Većina Android aplikacija se sada pravi pomoću alatke za gradnju koja se zove Gradle. Gradle projekti mogu da se prave pomoću tekst editora i komandne linije.

P: Alatka za gradnju? Da li je Gradle nešto kao ANT?

O: Slično, ali je Gradle mnogo moćnija alatka od ANT-a. Gradle može da prevede i uvede kôd, isto kao ANT, ali koristi i Maven za preuzimanje svake biblioteke drugih dobavljača potrebne vašem kodu. Gradle takođe koristi Groovy kao jezik za skriptovanje, što znači da pomoću njega možete lako da pravite prilično složene gradnje.

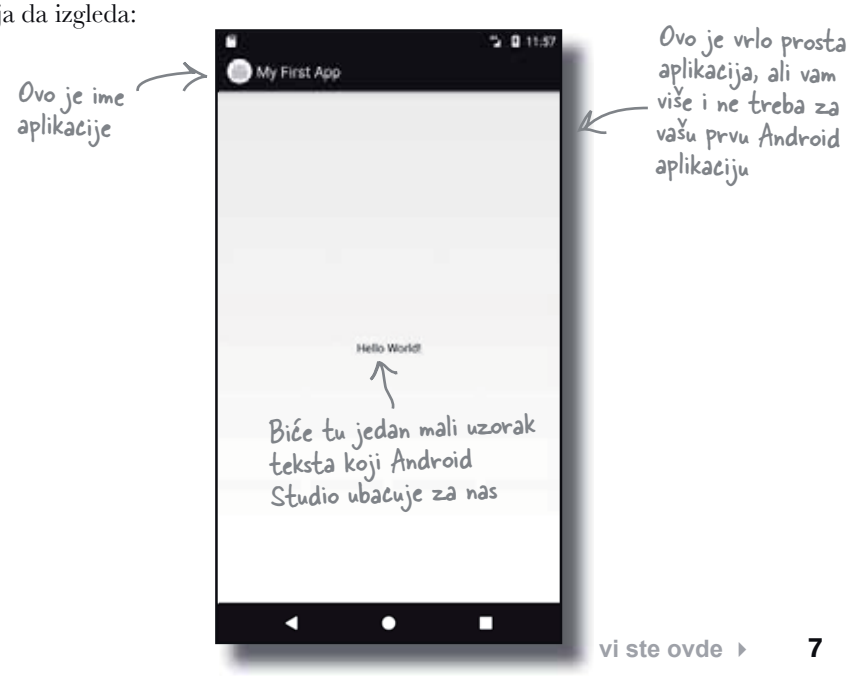
P: Većina aplikacija se pravi pomoću Gradla? Mislio sam da ste rekli da mnogi programeri koriste Android Studio?

O: Android Studio obezbeđuje grafički interfejs za Gradle, ali i za druge alatke za pravljenje rasporeda, za čitanje evidencija i za rešavanje grešaka.

U dodatku II možete saznati više o Gradleu.

Izrada jednostavne aplikacije

Pošto ste sada postavili svoje razvojno okruženje, spremni ste za pravljenje svoje prve Android aplikacije. Evo kako će vaša aplikacija da izgleda:



Uskočimo

Ovaj korak ste završili, pa smo ga potvrdili

- [x] Postavljanje okruženja
- [ ] **Izrada aplikacije**
- [ ] Izvršavanje aplikacije
- [ ] Menjanje aplikacije

Kako da napravimo aplikaciju

Kad god pravite novu aplikaciju, morate za nju da napravite nov projekat. Povedite računa da Android Studio bude otvoren i pratite šta mi radimo.

1. Napravite nov projekat

Ekran dobrodošlice Android Studio nudi vam niz opcija. Mi hoćemo da napravimo nov projekat, zato pritisnite opciju „Start a new Android Studio project" (Započeti nov projekat).



Pritisnite ovu opciju da biste započeli nov projekat Android Studija

Svi projekti koje napravite pojaviće se ovde. Ovo je vaš prvi projekat, pa je to područje trenutno prazno.

- [x] Postavljanje okruženja
- [ ] **Izrada aplikacije**
- [ ] Izvršavanje aplikacije
- [ ] Menjanje aplikacije

Kako da napravimo aplikaciju (nastavak)

2. Konfigurišite projekat

Sada morate da konfigurišete aplikaciju tako što ćete reći Android Studiju kako će aplikacija da se zove, koji kompanijski domen da se koristi i gde biste želeli da čuvate fajlove.

Android Studio koristi kompanijski domen i ime aplikacije kada pravi ime paketa koji će se koristiti za vašu aplikaciju. Na primer, ako svoju aplikaciju nazovete „My First App" i koristite kompanijski domen „hfad.com", Android Studio će iz toga da izvede ime paketa `com.hfad.myfirstapp`. Ime paketa je zaista važno za Android, pošto ga Android uređaji koriste kao jedinstvenu identifikaciju vaše aplikacije.

Upišite ime aplikacije „My First App", ime kompanijskog domena „hfad.com", poništite potvrdu opcije da se uključi C++ podrška i prihvatite podrazumevanu lokaciju projekta. Zatim pritisnite dugme Next.



Ime paketa mora da ostane isto dok god vaša aplikacija živi.

To je jedinstveni identifikator vaše aplikacije i koristi se za upravljanje svim verzijama iste aplikacije.

Neke verzije Android Studija imaju dodatnu opciju sa pitanjem da li želite da uključite Kotlin podršku. Ako ta opcija postoji, poništite njenu potvrdu.

Čarobnjak oblikuje ime paketa kao kombinaciju imena aplikacije i domena kompanije.



Kako da napravimo aplikaciju (nastavak)

3. Odredite minimalni SDK

Sada treba da odredite minimalni SDK Androida koji će vaša aplikacija da koristi. Nivoi API-ja rastu sa svakom novom verzijom Androida. Osim u slučaju da želite da se vaša aplikacija izvršava jedino na najnovijim uređajima, verovatno ćete izabrati neki od starijih API-ja.

Ovde biramo minimalni SDK API nivoa 19, što znači da će moći da se izvršava na većini uređaja. Osim toga, napravićemo verziju aplikacije koja će raditi samo na telefonima i tabletima, pa ostale opcije nećemo potvrditi.

Više o različitim nivoima API-ja videćete na sledećoj strani.

Pošto ovo uradite, pritisnite dugme Next.

Minimum SDK je najniža verzija koju će vaša aplikacija da podržava. Vaša aplikacija će se izvršavati na uređajima koji imaju taj nivo API-ja ili viši. Neće se izvršavati na uređajima koji imaju niže API-je.



ANDROID VERZIJE izbliza



Verovatno ste već čuli za mnoge stvari o Androidu koje zvuče kao nešto ukusno, na primer Jelly Bean (gumena bombona), KitKat, Lollipop (lizalica) ili Nougat. Šta znači sve to poslastičarstvo?

Android verzije imaju broj verzije i kodno ime. Broj verzije znači preciznu verziju Androida (npr. 7.0), dok je kodno ime nešto opštije, „razumljivo" ime koje može da se odnosi na nekoliko verzija Androida (npr. Nougat). API nivo se odnosi na verziju API-ja koju aplikacije koriste. Na primer, za Android verziju 7.1.1 odgovarajući nivo API-ja je 25.

Version	Codename	API level
1.0		1
1.1		2
1.5	Cupcake	3
1.6	Donut	4
2.0–2.1	Eclair	5–7
2.2.x	Froyo	8
2.3–2.3.7	Gingerbread	9–10
3.0–3.2	Honeycomb	11–13
4.0–4.0.4	Ice Cream Sandwich	14–15
4.1–4.3	Jelly Bean	16–18
4.4	KitKat	19–20
5.0–5.1	Lollipop	21–22
6.0	Marshmallow	23
7.0	Nougat	24
7.1–7.1.2	Nougat	25

Retko ko više koristi ove verzije

Većina uređaja koristi neki od ovih API-ja

Kada pravite Android aplikacije, zaista morate da razmotrite sa kojim verzijama Androida želite da vaša aplikacija bude kompatibilna. Ako odredite da će vaša aplikacija biti kompatibilna jedino sa najnovijom verzijom SDK, može se desiti da ona na mnogim uređajima neće moći da se izvršava. Ovde možete da vidite procente uređaja koji izvršavaju svaku konkretnu verziju:

https://developer.android.com/about/dashboards/index.html.

- [x] Postavljanje okruženja
- [ ] **Izrada aplikacije**
- [ ] Izvršavanje aplikacije
- [ ] Menjanje aplikacije

Aktivnosti i rasporedi iz daljine

Sledeće što će se tražiti od vas je da svom projektu dodate neku aktivnost. Svaka Android aplikacija je kolekcija ekrana, a svaki ekran se sastoji od jedne aktivnosti i jednog rasporeda.

Aktivnost (engl, *activity*) je **pojedinačna, definisana stvar koju vaš korisnik može da uradi**. Mogli biste da imate jednu aktivnost da se sastavi e-pošta, da se napravi fotografija, ili da se pronađe neki kontakt. Aktivnosti su obično pridružene jednom ekranu, a pišu se u Javi.

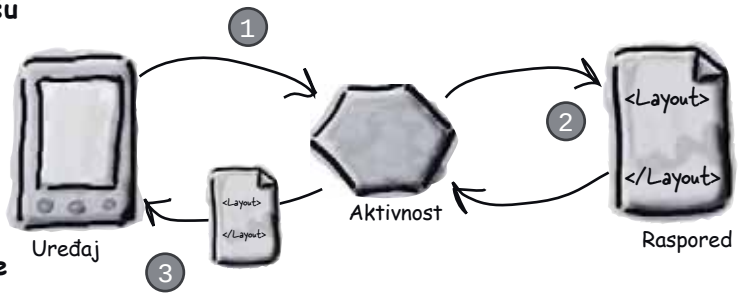
Raspored (engl, *layout*) opisuje **izgled ekrana**. Rasporedi se pišu kao XML fajlovi i oni kažu Androidu kako su raspoređeni različiti elementi ekrana.

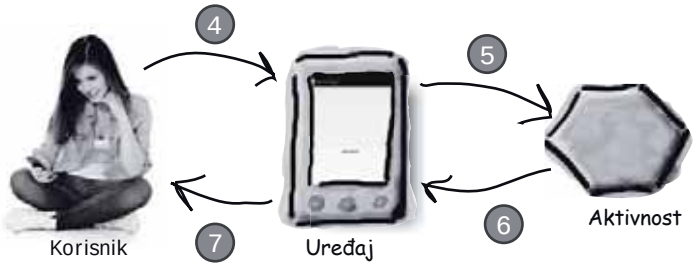
Rasporedi definišu kako se predstavlja korisnički interfejs.

Aktivnosti definišu akcije.

Pogledajmo detaljnije kako aktivnosti i rasporedi sarađuju u korisničkom interfejsu:

1. **Uređaj pokreće vašu aplikaciju i pravi objekat aktivnosti.**
2. **Objekat aktivnosti navede jedan raspored.**
1. **Aktivnost kaže Androidu da prikaže taj raspored na ekranu.**
4. **Korisnik komunicira sa rasporedom koji je prikazan na uređaju.**
5. **Aktivnost reaguje na tu komunikaciju tako što izvršava kôd aplikacije.**
6. **Aktivnost ažurira prikaz...**
7. **...koji korisnik vidi na uređaju.**





Pošto sada malo više znate o tome šta su aktivnosti i rasporedi, pređimo poslednjih par tačaka u čarobnjaku Create New Project, da nam napravi jednu osnovnu aktivnost i jedan raspored.

početak

- [x] Postavljanje okruženja
- [ ] **Izrada aplikacije**
- [ ] Izvršavanje aplikacije
- [ ] Menjanje aplikacije

Kako da napravimo aplikaciju (nastavak)

4. Dodajte jednu aktivnost

Sledeći ekran vam nudi niz šablona koje možete da koristite da biste napravili jednu aktivnost i jedan raspored. Napravićemo aplikaciju sa praznom aktivnošću i rasporedom, zato izaberite opciju Empty Activity i pritisnite dugme Next.



Postoje i druge vrste aktivnosti koje možete da birate, ali sada izaberite opciju Empty Activity

Uskočimo

Kako da napravimo aplikaciju (nastavak)

- [x] Postavljanje okruženja
- [ ] **Izrada aplikacije**
- [ ] Izvršavanje aplikacije
- [ ] Menjanje aplikacije

5. Konfigurišite aktivnost

Sada vam se postavlja pitanje kako želite da se zovu aktivnost i raspored ekrana. Upišite ime aktivnosti „MainActivity”, povedite računa da bude potvrđena opcija da se generiše fajl rasporeda (Generate Layout File), upišite ime rasporeda „activity_main”, a zatim poništite potvrdu opcije Backward Compatibility (AppCompat). Aktivnost je jedna Java klasa, a raspored je XML fajl, pa će imena koja smo ovde odredili napraviti jedan fajl Java klase po imenu *MainActivity.java* i jedan XML fajl po imenu *activity_main.xml*.

Kada pritisnete dugme Finish, Android Studio će napraviti vašu aplikaciju.



Dodelite aktivnosti ime „MainActivity”, a rasporedu ime „activity_main”. Povedite računa da bude potvrđena opcija za generisanje rasporeda.

Poništite potvrdu opcije Backward Compatibility (AppCompat). O ovom parametru ćete više saznati kasnije u knjizi.

Upravo ste napravili svoju prvu Android aplikaciju

- [x] Postavljanje okruženja
- [x] **Izrada aplikacije**
- [ ] Izvršavanje aplikacije
- [ ] Menjanje aplikacije

Dakle, šta se upravo dogodilo?

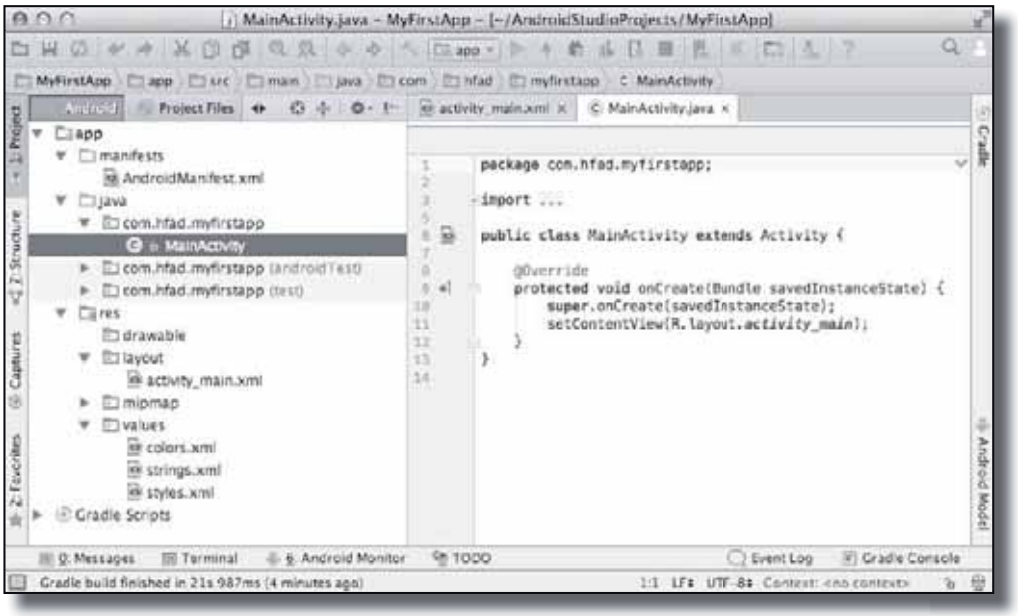
Čarobnjak Create New Project je napravio projekat za vašu aplikaciju, konfigurisan prema vašim specifikacijama.
Vi ste definisali sa kojim verzijama Androida treba aplikacija da bude kompatibilna, pa je čarobnjak napravio sve fajlove i foldere potrebne za propisnu osnovnu aplikaciju.

Čarobnjak je napravio aktivnost i raspored na osnovu šablona za kôd.
Šablon koda sadrži XML rasporeda i Java kôd aktivnosti, sa uzorkom teksta „Hello world!" u rasporedu.

Kada završite pravljenje projekta pomoću čarobnjaka, Android Studio vam automatski prikazuje projekat.

Evo kako vaš projekat izgleda (ne brinite ako vam sada to izgleda komplikovano, raščlanićemo ga na sledećih nekoliko stranica):

Ovo je taj projekat u Android Studiju

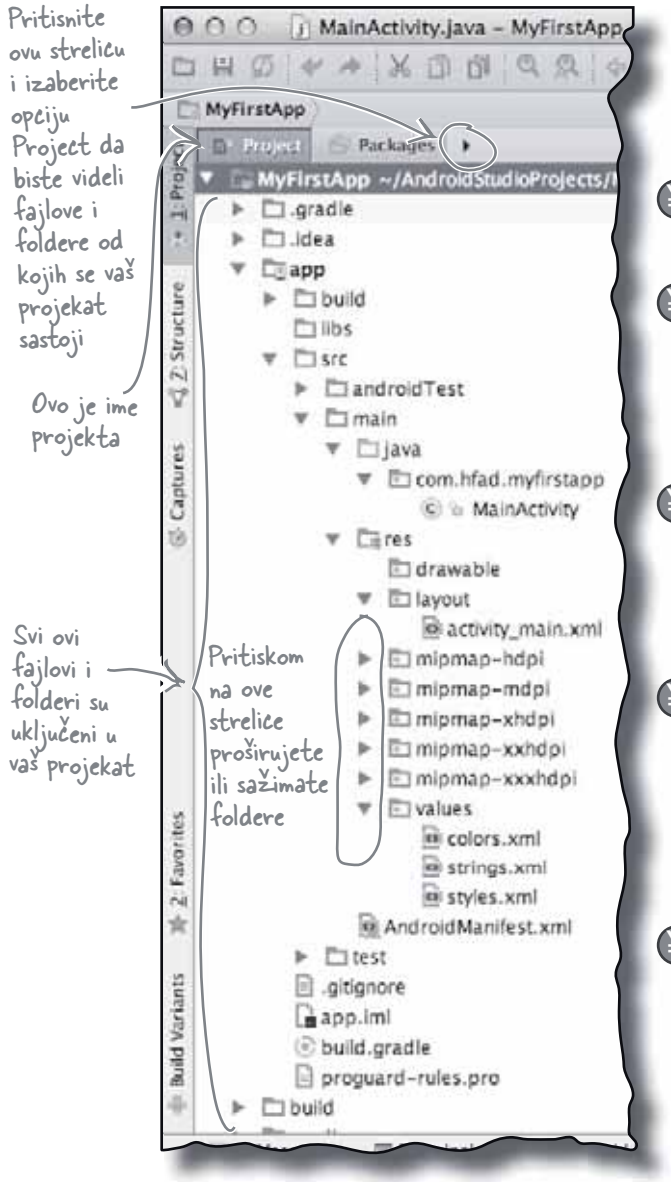


- [x] Postavljanje okruženja
- [x] **Izrada aplikacije**
- [ ] Izvršavanje aplikacije
- [ ] Menjanje aplikacije

Android Studio vam pravi kompletnu strukturu foldera

Android aplikacija je u stvari samo gomila propisnih fajlova u konkretnoj strukturi foldera, a Android Studio vam sve to postavlja kada pravite novu aplikaciju. Najlakši način za pregledanje te strukture foldera je pomoću eksplorera u prvoj koloni Android Studija, sasvim levo.

Eksplorer sadrži sve projekte koji su vam trenutno otvoreni. Da biste proširili ili saželi folder, samo pritisnite na strelicu levo od ikone foldera.



Struktura foldera sadrži različite vrste fajlova

Ako pregledate strukturu foldera, primetićete da vam je čarobnjak napravio različite vrste fajlova i foldera:

Izvorni Java i XML fajlovi
To su fajlovi aktivnosti i rasporeda za vašu aplikaciju.

Java fajlovi koje pravi Android
Postoje i neki dodatni Java fajlovi koje ne morate da dirate, a koje Android Studio automatski pravi za vas.

Fajlovi resursa
To su podrazumevani slikovni fajlovi za ikone, stilovi koje bi aplikacija mogla da koristi i neke zajedničke String vrednosti koje bi aplikacija mogla da traži.

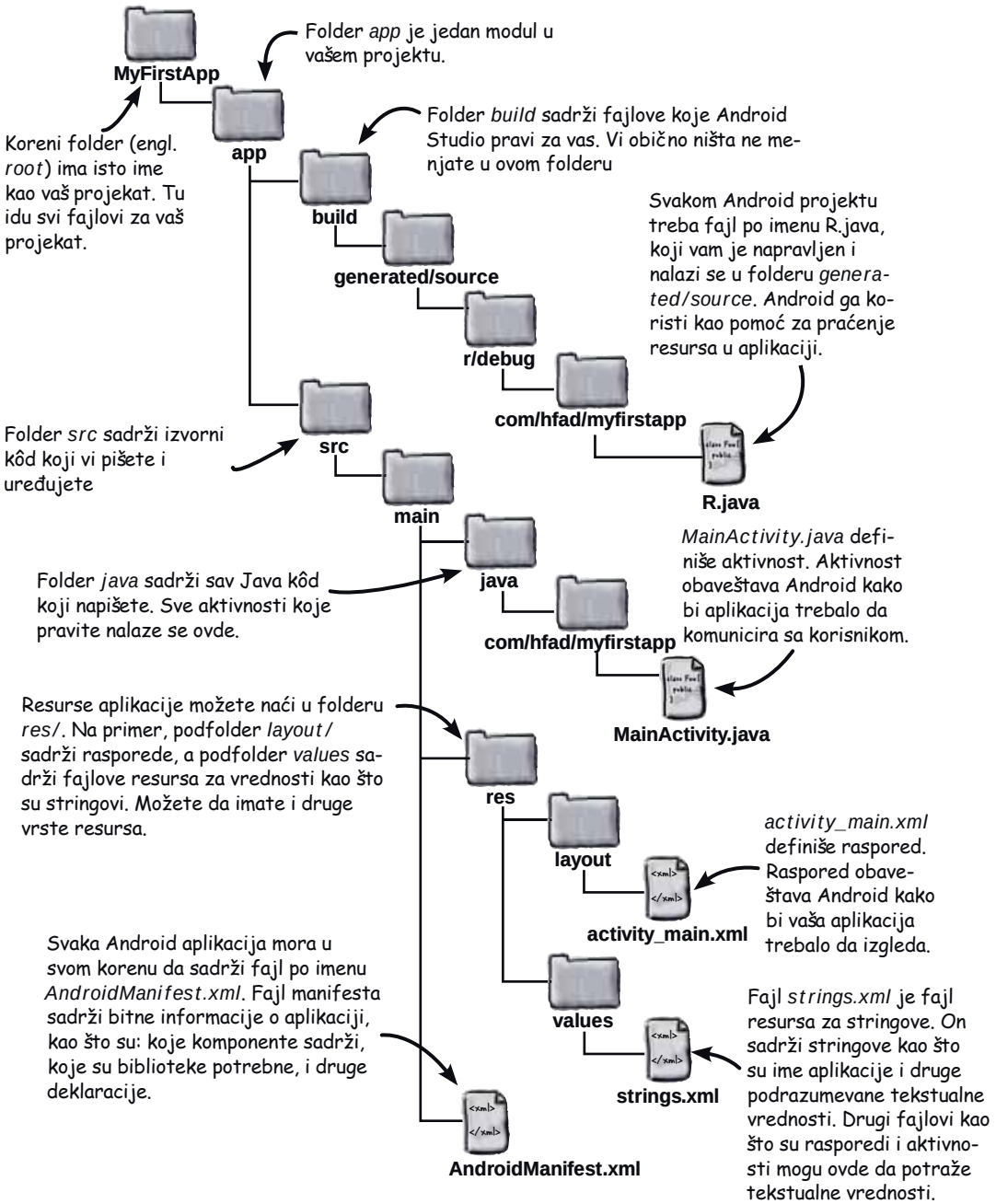
Android biblioteke
U čarobnjaku ste naveli minimalnu verziju SDK sa kojom vaša aplikacija treba da bude kompatibilna. Android Studio obezbeđuje uključivanje relevantnih Android biblioteka za tu verziju.

Konfiguracioni fajlovi
Konfiguracioni fajlovi kazuju Androidu šta se u stvari nalazi u aplikaciji i kako bi ona trebalo da se izvršava.

Pogledajmo izbliza neke od ključnih fajlova i foldera u svetu Androida.

Korisni fajlovi u vašem projektu

Projekti Android Studija koriste sistem za gradnju Gradle za prevođenje i postavljanje vaših aplikacija. Gradle projekti imaju standardnu strukturu. Ovo su neki od ključnih fajlova i foldera sa kojima ćete raditi:



Uskočimo

Menjajte kôd pomoću editora Android Studija

- [x] Postavljanje okruženja
- [x] Izrada aplikacije
- [ ] Izvršavanje aplikacije
- [ ] Menjanje aplikacije

Fajlove pregledate i menjate pomoću editora Android Studija. Pritisnite dva puta na fajl sa kojim hoćete da radite, pa će se sadržaj fajla pojaviti na sredini prozora Android Studio.

Editor koda

Većina fajlova se prikazuje u editoru koda. Editor koda je kao neki tekst editor, ali ima dodatne osobine kao što je kodiranje bojom i provera koda.

Pritisnite dva puta na fajl u eksploreru, a sadržaj fajla se pojavljuje u panelu editora.



Editor dizajna

Ako menjate raspored, imate dodatnu opciju. Umesto da menjate XML (onakav kao što se vidi na sledećoj strani), možete da koristite editor dizajna koji vam omogućava da povučete GUI komponente na svoj raspored i namestite ih kako hoćete. Editor koda i editor dizajna daju različite prikaze istog fajla, pa možete da prelazite iz jednog u drugi kako želite.

ŠTA SAM HTEO?

Ovde je kôd iz jednog primera fajla rasporeda **(to nije fajl koji nam je napravio Android Studio)**. Znamo da niste ranije videli kôd rasporeda, ali samo pogledajte da li možete da uparite opise na dnu stranice sa odgovarajućim redovima koda. Rešili smo vam jedan za početak.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp"
    android:orientation="vertical"
    tools:context="com.hfad.myfirstapp.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!" />
</LinearLayout>
```

Dodavanje popunjavanja (engl. padding) marginama ekrana.

Uključivanje komponente grafičkog korisničkog interfejsa (engl. GUI) <TextView> radi prikazivanja teksta.

Napraviti GUI komponentu dovoljno veliku za njen sadržaj.

Prikazati string "Hello World!"

Obezbediti da raspored bude iste širine i visine kao ekran uređaja.

ŠTA SAM HTEO? REŠENJE

Ovde je kôd iz jednog primera fajla rasporeda **(to nije fajl koji nam je napravio Android Studio)**. Znamo da niste ranije videli kôd rasporeda, ali samo pogledajte da li možete da uparite opise na dnu stranice sa odgovarajućim redovima koda. Rešili smo vam jedan za početak.

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="16dp"
    android:paddingRight="16dp"
    android:paddingTop="16dp"
    android:paddingBottom="16dp"
    android:orientation="vertical"
    tools:context="com.hfad.myfirstapp.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!" />
</LinearLayout>
```

Dodavanje popunjavanja (engl. padding) marginama ekrana.

Prikazati string "Hello World!"

Uključivanje komponente grafičkog korisničkog interfejsa (engl. GUI) <TextView> radi prikazivanja teksta.

Obezbediti da raspored bude iste širine i visine kao ekran uređaja.

Napraviti GUI komponentu dovoljno veliku za njen sadržaj.



Hajde sada da vidimo da li možete da uradite isto sa kodom aktivnosti. **Ovo je primer koda, a ne kôd koji vam je Android Studio napravio.**

Uparite donje opise sa odgovarajućim redovima koda.

MainActivity.java

```
package com.hfad.myfirstapp;

import android.os.Bundle;
import android.app.Activity;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Ovo je ime paketa.

Primena metoda `onCreate()` iz klase `Activity`. Ovaj metod se poziva kada se aktivnost prvi put pravi.

Ovo su Android klase koje se koriste u `MainActivity`.

Određuje koji raspored da se koristi.

`MainActivity` proširuje Android klasu `android.app.Activity`.

ŠTA SAM HTEO? REŠENJE

Hajde sada da vidimo da li možete da uradite isto sa kodom aktivnosti. **Ovo je primer koda, a ne kôd koji vam je Android Studio napravio.**

Uparite donje opise sa odgovarajućim redovima koda.

MainActivity.java

```
package com.hfad.myfirstapp;

import android.os.Bundle;
import android.app.Activity;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Ovo je ime paketa.

Ovo su Android klase koje se koriste u `MainActivity`.

Određuje koji raspored da se koristi.

Primena metoda `onCreate()` iz klase `Activity`. Ovaj metod se poziva kada se aktivnost prvi put pravi.

`MainActivity` proširuje Android klasu `android.app.Activity`.

Izvršavanje aplikacije u emulatoru Androida

Do sada ste videli kako vaša Android aplikacija izgleda u Android Studiju i stekli utisak o tome kako je sastavljena. Ali ono što biste *stvarno* voleli je da vidite kako radi, zar ne?

Za izvršavanje aplikacija imate dve opcije. Prva opcija je da ih izvršite na fizičkom uređaju. Ali šta ako vam uređaj nije pri ruci, ili hoćete da vidite kako izgleda na nekoj vrsti uređaja koju vi nemate?

U tom slučaju možete da upotrebite **emulator Androida** koji je ugrađen u Android SDK. Emulator vam omogućava da postavite jedan ili više **virtuelnih Android uređaja** – AVD-a (Android virtual device) i zatim izvršite svoju aplikaciju u emulatoru *kao da se izvršava na fizičkom uređaju*.

Emulator Androida vam omogućava da izvršite svoju aplikaciju na virtuelnom Android uređaju (AVD). AVD se ponaša isto kao fizički Android uređaj. Možete da postavite više AVD-ova, od kojih svaki emulira drugačiju vrstu uređaja.

Pa kako taj emulator izgleda?

Ovde je jedan AVD koji se izvršava u emulatoru Androida. On izgleda isto kao telefon, a izvršava se na vašem računaru.

Emulator je aplikacija koja kreira precizno hardversko okruženje jednog Android uređaja: od procesora i memorije, sve do čipova zvuka i video prikaza. Emulator je izgrađen na postojećem emulatoru po imenu QEMU (izgovara se „kju em ju"), a on je sličan drugim aplikacijama virtuelnih mašina koje ste možda koristili, kao što su VirtualBox ili VMWare.

Precizan izgled i ponašanje AVD-a zavisi od toga kako ste na početku postavili AVD. Ovaj AVD je postavljen da imitira Nexus 5X, pa će na vašem računaru izgledati i ponašati se tačno kao Nexus 5X.

Hajde da postavimo jedan AVD da biste videli svoju aplikaciju kako radi u emulatoru.



Pošto ste postavili AVD, moći ćete da vidite kako vaša aplikacija radi u njemu. Android Studio pokreće emulator za vas.

Uskočimo

- [x] Postavljanje okruženja
- [x] Izrada aplikacije
- [ ] **Izvršavanje aplikacije**
- [ ] Menjanje aplikacije

Pravimo jedan virtuelni Android uređaj

Postoji nekoliko koraka koje treba da prođete da biste postavili jedan AVD unutar Android Studija. Mi ćemo da postavimo jedan Nexus 5X AVD koji izvršava API nivoa 25, da biste mogli da vidite kako vaša aplikacija izgleda i kako se ponaša kada se izvršava na toj vrsti uređaja. Koraci su uglavnom identični bez obzira na vrstu uređaja koji želite da postavite.

Otvorite Android Virtual Device Manager

AVD Manager dozvoljava da postavite nove AVD-ove, kao i da pregledate i menjate one koje ste ranije napravili. Otvarate ga tako da izaberete Android na meniju Tools, pa izaberete AVD Manager.

Ako nemate nijedan već postavljeni AVD, pojaviće se ekran kojim vam se nudi da ga napravite. Pritisnite dugme „Create a virtual device"(Napraviti virtuelni uređaj).

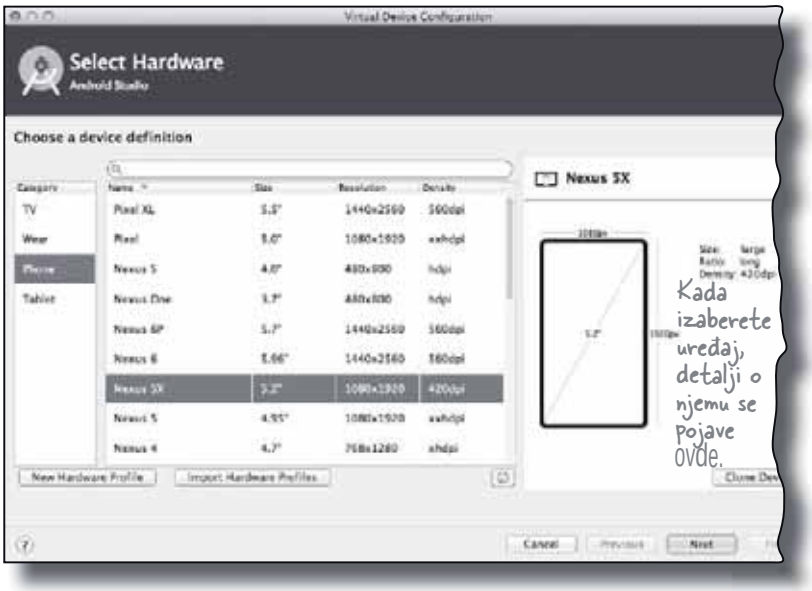


Pritisnite ovo dugme da biste napravili AVD.

Izaberite hardver

Na sledećem ekranu, tražiće se da izaberete definiciju uređaja. To je vrsta uređaja koji će vaš AVD da emulira. Možete da birate raznovrsne telefone, tablete, nosive uređaje, ili TV uređaje.

Mi ćemo da gledamo kako vaša aplikacija izgleda kada se izvršava na telefonu Nexus 5X. Izaberite Phone na meniju Category, a na listi izaberite Nexus 5X. Zatim pritisnite dugme Next.



Pravimo jedan AVD (nastavak)

Izaberite sliku sistema

Posle toga, treba da izaberete sliku sistema. Slika sistema vam daje instaliranu verziju operativnog sistema Android. Možete da birate koju verziju Androida želite za svoj AVD.

Morate da izaberete sliku sistema za API nivo koji je kompatibilan sa aplikacijom koju pravite. Na primer, ako hoćete da vaša aplikacija radi minimalno na API nivou 19, izaberite sliku sistema za *najmanje* API nivo 19. Mi hoćemo da naš AVD koristi API nivoa 25, zato izaberite sliku sistema sa imenom izdanja Nougat i sa ciljem (engl. *target*) Android 7.1.1 (API nivoa 25). Zatim pritisnite dugme Next.

Ako vam ova slika sistema nije instalirana, ponudiće vam se opcija da je preuzmete.



Na sledećoj strani nastavljamo sa postavljanjem AVD-a.

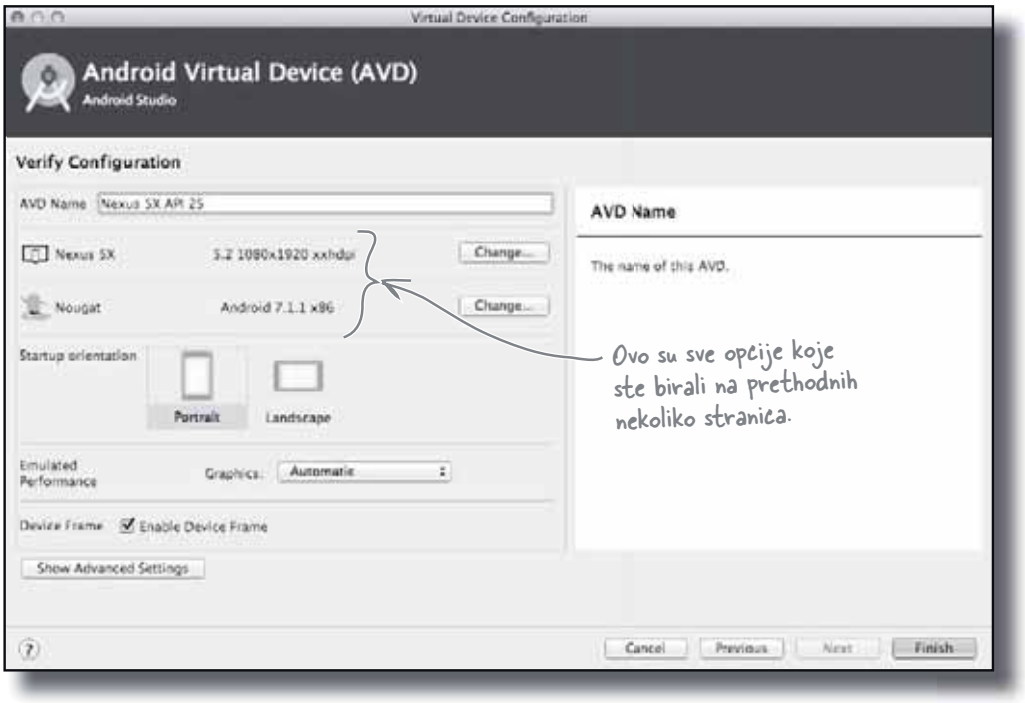
Uskočimo

- [x] Postavljanje okruženja
- [x] Izrada aplikacije
- [ ] **Izvršavanje aplikacije**
- [ ] Menjanje aplikacije

Pravimo jedan AVD (nastavak)

Proverite AVD konfiguraciju

Na sledećem ekranu, zatražiće se da proverite AVD konfiguraciju. Na ovom ekranu je rekapitulacija opcija koje ste odabrali na nekoliko prethodnih ekrana i nudi vam se opcija da nešto promenite (engl. *Change*). Prihvatite opcije, pa pritisnite dugme Finish.



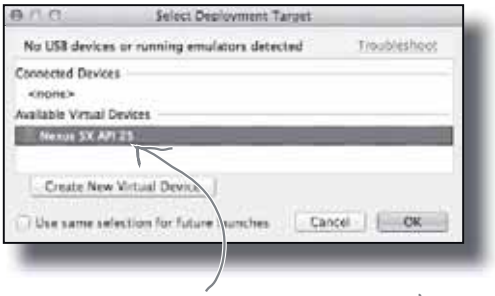
AVD Manager će vam napraviti AVD, a kada završi, prikazaće ga u listi uređaja AVD Managera. Sada možete da zatvorite AVD Manager



Izvršavanje aplikacije u emulatoru

Pošto ste sada postavili svoj AVD, hajde da na njemu izvršimo vašu aplikaciju. Da biste to uradili, izaberite komandu „Run 'app'" na meniju Run. Kada se pojavi zahtev da izaberete uređaj (engl. *Select Deployment Target*), izaberite AVD Nexus 5X koji ste upravo napravili. Zatim pritisnite dugme OK.

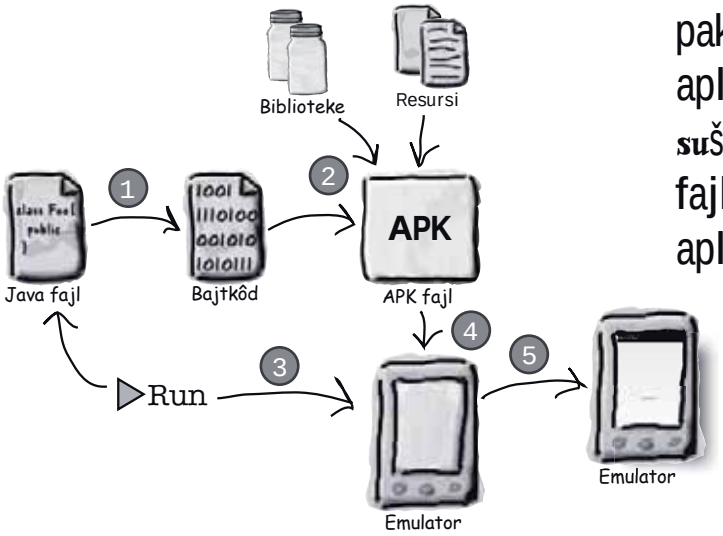
Dok strpljivo čekamo da se AVD pojavi, što može da potraje nekoliko minuta, pogledajmo šta se dešava kada izaberete Run.



Ovo je AVD koji smo upravo napravili.

Prevođenje, pakovanje, stavljanje u upotrebu i izvršavanje

Kada izaberete opciju Run, ona neće samo da izvrši vašu aplikaciju. Ona se takođe bavi svim pripremnim poslovima potrebnim da bi se aplikacija izvršavala:



APK fajl je pakovanje Android aplikacije. To je u suštini JAR ili ZIP fajl za Android aplikacije.

1. **Izvorni Java fajlovi se prevode u bajtkôd.**
2. **Pravi se pakovanje Android aplikacije, ili APK fajl.**
   APK fajl sadrži prevedene Java fajlove, pored biblioteka i resursa potrebnih vašoj aplikaciji.
3. **Pod pretpostavkom da se trenutno ne izvršava nijedan, pokreće se emulator, a zatim AVD.**
4. **Pošto je emulator pokrenut, a AVD aktivan, APK fajl se učitava u AVD i instalira.**
5. **AVD pokreće glavnu aktivnost pridruženu aplikaciji.**
   Vaša aplikacija se prikazuje na AVD ekranu spremna da je testirate.

Uskočimo

Možete na konzoli da pratite napredovanje

- [x] Postavljanje okruženja
- [x] Izrada aplikacije
- [ ] **Izvršavanje aplikacije**
- [ ] Menjanje aplikacije

Ponekad može dosta da potraje dok se ne pokrene emulator sa vašim AVD-om – često *nekoliko minuta*. Ako želite, možete na konzoli Android Studija da posmatrate šta se dešava. Konzola vas detaljno izveštava šta radi Gradle sistem za gradnju, a ako dođe do neke greške, ona će biti istaknuta u tekstu.

Predlažemo da radite nešto drugo dok čekate da se emulator pokrene. Na primer, ručni rad, ili kuvanje malog obroka.

Konzolu možete naći na dnu ekrana Android Studija (ako se nije automatski pojavila, pritisnite opciju Run na dnu ekrana):



Ovde je izlaz u našem prozoru konzole kada smo izvršili vašu aplikaciju:

```
03/13 10:45:41: Launching app
$ adb install-multiple -r /Users/dawng/AndroidStudioProjects/MyFirstApp/
app/build/intermediates/split-apk/debug/dep/dependencies.apk /Users/
dawng/AndroidStudioProjects/MyFirstApp/app/build/intermediates/split-apk/
debug/slices/slice_1.apk /Users/dawng/AndroidStudioProjects/MyFirstApp/
app/build/intermediates/split-apk/debug/slices/slice_2.apk /Users/dawng/
AndroidStudioProjects/MyFirstApp/app/build/intermediates/split-apk/debug/
slices/slice_0.apk /Users/dawng/AndroidStudioProjects/MyFirstApp/app/
build/intermediates/split-apk/debug/slices/slice_3.apk /Users/dawng/
AndroidStudioProjects/MyFirstApp/app/build/intermediates/split-apk/debug/
slices/slice_6.apk /Users/dawng/AndroidStudioProjects/MyFirstApp/app/
build/intermediates/split-apk/debug/slices/slice_4.apk /Users/dawng/
AndroidStudioProjects/MyFirstApp/app/build/intermediates/split-apk/debug/
slices/slice_5.apk /Users/dawng/AndroidStudioProjects/MyFirstApp/app/
build/intermediates/split-apk/debug/slices/slice_7.apk /Users/dawng/
AndroidStudioProjects/MyFirstApp/app/build/intermediates/split-apk/debug/
slices/slice_8.apk /Users/dawng/AndroidStudioProjects/MyFirstApp/app/
build/intermediates/split-apk/debug/slices/slice_9.apk /Users/dawng/
AndroidStudioProjects/MyFirstApp/app/build/outputs/apk/app-debug.apk

Split APKs installed

$ adb shell am startservice com.hfad.myfirstapp/com.android.tools.fd.runtime.
InstantRunService

$ adb shell am start -n "com.hfad.myfirstapp/com.hfad.myfirstapp.MainActivity" -a
android.intent.action.MAIN -c android.intent.category.LAUNCHER

Connected to process 2685 on device Nexus_5X_API_25 [emulator-5554]
```

Instaliranje aplikacije.

Android Studio je završio pokretanje AVD-a koji smo upravo napravili.

Emulator pokreće našu aplikaciju tako što pokreće njenu glavnu aktivnost. To je aktivnost koju je čarobnjak napravio za nas.



Probna vožnja

Dakle, da vidimo šta se u stvari dešava na ekranu kada vi izvršite svoju aplikaciju.

Najpre, pali se emulator u zasebnom prozoru. Emulatoru je potrebno neko vreme da učita AVD, ali onda vidite kako on izgleda na pravom Android uređaju.



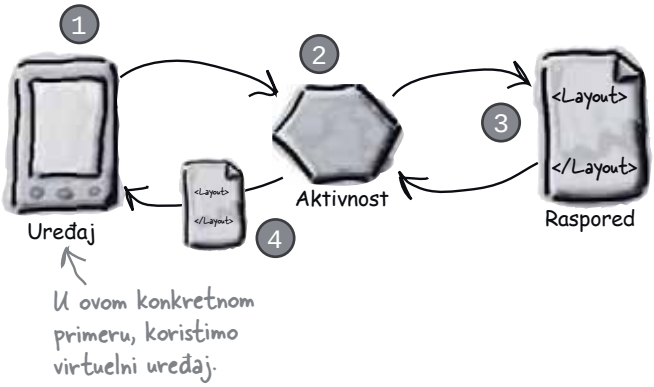
Sačekajte još malo i videćete aplikaciju koju ste upravo napravili. Ime aplikacije se vidi na vrhu ekrana, a podrazumevani uzorak teksta „Hello world!" prikazan je na sredini ekrana.

- [x] Postavljanje okruženja
- [x] Izrada aplikacije
- [x] **Izvršavanje aplikacije**
- [ ] Menjanje aplikacije

Šta se to desilo?

Hajde da raščlanimo to što se dešava kada izvršite aplikaciju:

1. **Android Studio pokreće emulator, učitava AVD i instalira aplikaciju.**
2. **Kada se aplikacija pokrene, pravi se objekat aktivnosti od MainActivity.java.**
3. **Aktivnost kaže da koristi raspored activity_main.xml.**
4. **Aktivnost naređuje Androidu da prikaže raspored na ekranu.** Tekst „Hello World!" se prikazuje.



nema GLUPIH PITANJA

P: Pomenuli ste da kada pravite APK fajl, izvorni Java kôd se prevodi u bajtkod i dodaje se u APK. Pretpostavljam da ste mislili da se prevodi u Java bajtkôd, zar ne?

O: Tako je, ali to nije kraj priče. Stvari su malo drugačije u Androidu.

Glavna razlika u Androidu je da se vaš kôd u stvari ne izvršava u običnoj Java VM (virtuelnoj mašini). On se izvršava u Android izvršnom okruženju (ART Android runtime), a na starijim uređajima se izvršava u prethodniku ART-a koji se zove Dalvik. To znači da vi napišete svoj izvorni Java kôd, prevedete ga u .class fajlove pomoću Java kompajlera, a zatim se .class fajlovi povežu u jedan fajl u DEX formatu, koji je manji, efikasniji bajtkôd. ART zatim izvršava DEX kôd. Više detalja o tome možete naći u Dodatku III

P: To izgleda komplikovano. Zašto se ne koristi normalna Java VM?

O: ART ume da konvertuje DEX bajtkôd u svojstveni kôd koji može da se izvršava direktno u procesoru Android uređaja. Tako se aplikacija izvršava mnogo brže i mnogo manje koristi snagu baterije.

P: Zar bi Java virtuelna mašina zaista bila toliko opterećenje?

O: Da. Zato što se u Androidu svaka aplikacija izvršava u vlastitom procesu. Kad bi koristila obične JVM-ove, bilo bi potrebno daleko više memorije.

P: Da li moram da pravim nov AVD kad god pravim novu aplikaciju?

O: Ne, kad jednom napravite AVD možete da ga koristite za bilo koju aplikaciju. Možda će vam koristiti da napravite više AVD-ova da biste testirali svoje aplikacije u različitim situacijama. Na primer, mogli biste, pored AVD-a za telefon, da napravite tablet AVD, da biste videli kako vaša aplikacija izgleda i kako se ponaša na većim uređajima.

početak

- [x] Postavljanje okruženja
- [x] Izrada aplikacije
- [x] Izvršavanje aplikacije
- [ ] **Menjanje aplikacije**

Doterivanje aplikacije

Na prethodnih nekoliko stranica, napravili ste jednostavnu Android aplikaciju i videli ste je kako se izvršava u emulatoru. Sada ćemo da doteramo aplikaciju koju ste napravili.

Trenutno, aplikacija prikazuje uzorak teksta „Hello world!" koji nam je čarobnjak ubacio kao čuvar mesta. Vi ćete da promenite taj tekst u nešto drugo. Dakle, šta moramo da promenimo da bismo to postigli? Da bismo odgovorili na to pitanje, vratimo se malo i pogledajmo kako je aplikacija trenutno izgrađena.



Aplikacija ima jednu aktivnost i jedan raspored

Kada smo pravili aplikaciju, rekli smo Android Studiju kako da je konfiguriše, pa je čarobnjak obavio ostalo. Čarobnjak nam je napravio jednostavnu aktivnost i takođe jedan podrazumevani raspored.

Aktivnost kontroliše šta će aplikacija da radi

Android Studio nam je napravio aktivnost po imenu *MainActivity.java*. Aktivnost određuje šta aplikacija **radi** i kako bi trebalo da odgovara korisniku.

Raspored kontroliše izgled aplikacije

MainActivity.java navodi da koristi raspored koji nam je Android Studio napravio i koji se zove *activity_main.xml*. Raspored određuje kako aplikacija **izgleda**.



Mi hoćemo da promenimo izgled aplikacije tako što ćemo promeniti tekst koji se prikazuje. To znači da moramo da se pozabavimo Android komponentom koja kontroliše kako aplikacija izgleda, zato moramo izbliza da pogledamo *raspored*.

Šta je u rasporedu?

Mi hoćemo da promenimo uzorak teksta „Hello world!" koji nam je Android Studio napravio, pa ćemo da počnemo od fajla rasporeda *activity_main.xml*. Ako nije već otvoren u editoru, otvorite ga sada tako da u eksploreru pronađete fajl u folderu *app/src/main/res/layout* i pritisnete ga dva puta.

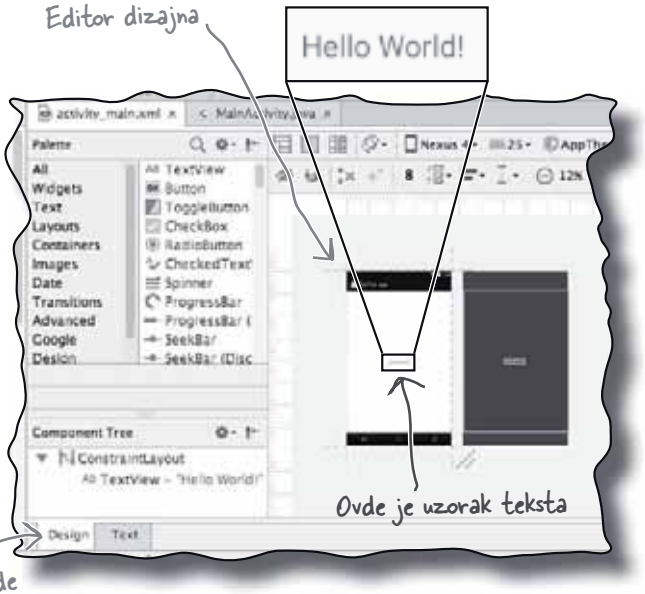


Editor dizajna

Kao što ste ranije naučili, u Android Studiju postoje dva načina za pregledanje i menjanje fajlova rasporeda: kroz **editor dizajna** i kroz **editor koda**.

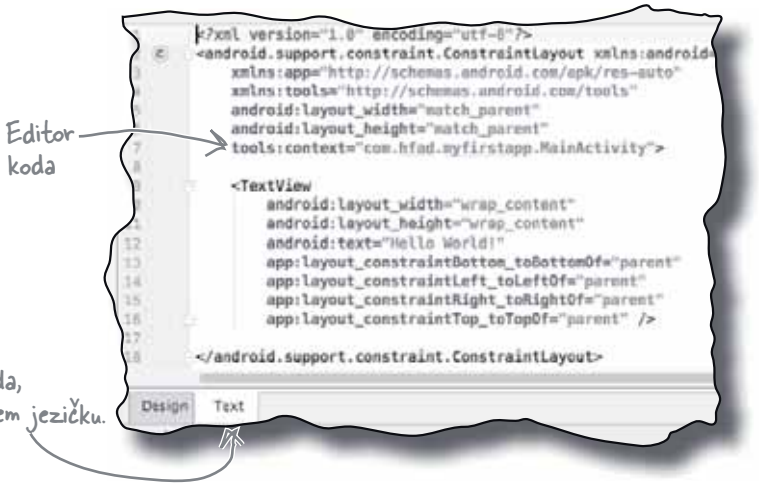
Kada izaberete opciju dizajna, možete da vidite da se uzorak teksta „Hello world!" pojavljuje u rasporedu kao što biste očekivali. Ali šta se nalazi u XML-u u njegovoj osnovi?

To ćemo da vidimo ako se prebacmo u editor koda.



Editor koda

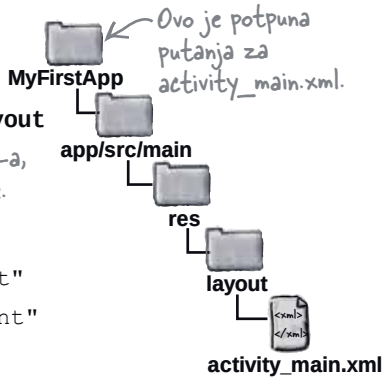
Kada izaberete opciju editora koda, prikazuje se sadržaj fajla *activity_main.xml*. Hajde da ga bolje pogledamo.



activity_main.xml se sastoji od dva elementa

Ovde je kôd fajla *activity_main.xml* koji nam je Android Studio napravio. Izostavili smo neke detalje o kojima trenutno ne morate da razmišljate, obradićemo ih detaljnije u ostatku knjige.

Ovde je naš kôd:




```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    ... >
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        ... />
</android.support.constraint.ConstraintLayout>
```

Ovaj element određuje kako treba prikazati komponente, u ovom rimeru to je tekst „Hello World!".

Android Studio je ovde ubacio još XML-a, ali trenutno ne morate da mislite o tome.

Ovo je <TextView> element.

Izostavili smo i deo XML-a u <TextView>.

Kao što vidite, kôd sadrži dva elementa.

Prvi element je element `<android.support.constraint.ConstraintLayout>`. Ovaj element rasporeda kaže Androidu kako da prikaže komponente na ekranu uređaja. Imate na raspolaganju različite vrste komponenti rasporeda, saznaćete više o njima u ostatku knjige.

Za sada je najvažniji drugi element, `<TextView>`. Ovaj element se koristi da bi se korisniku prikazao tekst, a u našem slučaju to je uzorak teksta „Hello world!".

Ključni deo koda u elementu `<TextView>` je red koji počinje sa `android:text`. To je svojstvo teksta koje opisuje tekst koji treba prikazati:

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello World!"
    ... />
```

Element <TextView> opisuje tekst u rasporedu.

Ovo je tekst koji se prikazuje.

Hajde da promenimo tekst u nešto drugo.



Ne brinite ako kôd vašeg rasporeda nije isti kao naš.

Android Studio može da vam napravi neznatno drugačiji XML, zavisno od verzije koju koristite. To ne treba da vas brine, jer ćete počevši od sledećeg poglavlja učiti kako da pravite vlastiti kôd za raspored i zamenite veliki deo onoga što dobijete od Android Studija.

Ažuriramo tekst prikazan u rasporedu

- [x] Postavljanje okruženja
- [x] Izrada aplikacije
- [x] Izvršavanje aplikacije
- [ ] Menjanje aplikacije

Ključni deo elementa `<TextView>` je ovaj red:

```
android:text="Hello World!" />
```

`android:text` znači da je ovo svojstvo `text` elementa `<TextView>` i da navodi koji tekst treba da se prikaže u rasporedu. U ovom slučaju, tekst koji se prikazuje je „Hello World!"

Prikazati tekst... ... „Hello World!"

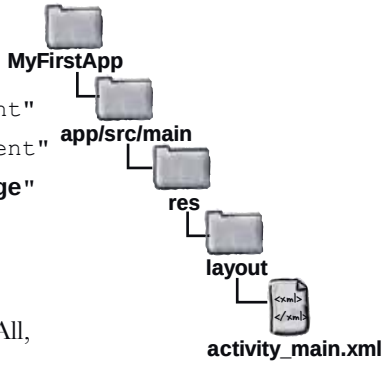
```
android:text="Hello World!" />
```

Da biste ažurirali tekst koji se prikazuje u rasporedu, jednostavno promenite vrednost svojstva `text` iz `"Hello World!"` u `"Sup doge"`. Novi kôd za `<TextView>` trebalo bi da izgleda ovako:

Izostavili smo delove koda, pošto sada samo menjamo tekst koji se prikazuje.

```
...
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello World!Sup doge"
    ... />
...
```

Promenite ovde tekst iz „Hello World!" u „Sup doge".



Pošto ažurirate fajl, idite na meni File i izaberite opciju Save All, da biste sačuvali svoju izmenu.

nema GLUPIH PITANJA

P: Moj kôd rasporeda se razlikuje od vašeg. Da li je to u redu?

O: Da, U redu je. Android Studio može da generiše neznatno drugačiji kôd ako ne koristite istu verziju kao mi, ali to zaista nije važno. Od sada ćete učiti kako da pravite vlastiti kôd rasporeda, pa ćete da zamenite veliki deo onoga što vam napravi Android Studio.

P: Jesam li u pravu što mislim da tekst koji treba da se prikaže kodiramo u program?

O: Da, samo da biste videli kako da ažurirate tekst u rasporedu. Postoji bolji način da se prikažu tekstualne vrednosti nego da se kodiraju u raspored, ali morate da sačekate sledeće poglavlje da biste naučili šta je to.

P: Folderi u mom oknu project eksplorera se razlikuju od vaših. Zbog čega?

O: Android Studio dozvoljava da birate kako će se prikazati hijerarhija foldera, a podrazumevani prikaz je „Android". Mi više volimo prikaz „Project", pošto on odražava strukturu foldera u osnovi. Vi možete da prebacite svoj eksplorer u prikaz „Project" ako pritisnete strelicu na vrhu eksplorerovog okna, pa izaberete opciju „Project".

Mi koristimo prikaz Project.



Pritisnite ovu strelicu za promenu prikaza u eksploreru.



Povedimo aplikaciju na probnu vožnju

Pošto ste promenili fajl, pokušajte ponovo da izvršite svoju aplikaciju u emulatoru tako da izaberete komandu „Run 'app'" na meniju Run. Trebalo bi da vidite da vaša aplikacija sada kaže „Sup doge" umesto „Hello world!".



Sada ste napravili i ažurirali svoju prvu Android aplikaciju.

Uskočimo



Vaša kutija Android alatki

Savladali ste Poglavlje 1 i sada ste dodali osnovne koncepte Androida u svoju kutiju alatki.

Sav kôd za ovo poglavlje možete da preuzmete sa https://tinyurl.com/HeadFirstAndroid.

UKRATKO

- Verzije Androida imaju broj verzije, API nivo i kodno ime.
- Android Studio je specijalna verzija IntelliJ IDEA sa interfejsom prema Android SDK-u (Software Development Kit) i sistemu gradnje Gradle.
- Tipična Android aplikacija se sastoji od fajlova aktivnosti, rasporeda i resursa.
- Rasporedi opisuju kako vaša aplikacija izgleda. Oni se čuvaju u folderu *app/src/main/res/layout*.
- Aktivnosti opisuju šta vaša aplikacija radi i kako komunicira sa korisnikom. Aktivnosti koje napišete čuvaju se u folderu *app/src/main/java*.
- *AndroidManifest.xml* sadrži informacije o samoj aplikaciji. Nalazi se u folderu *app/src/main*.
- AVD (Android Virtual Device) je virtuelni Android uređaj. Izvršava se u Android emulatoru i imitira fizički Android uređaj.
- APK (Android application package) je paket Android aplikacije. To je kao JAR fajl za Android aplikacije i sadrži bajtkôd vaše aplikacije, biblioteke i resurse. Aplikaciju instalirate na uređaj tako što instalirate APK.
- Android aplikacije se izvršavaju u zasebnim procesima i koriste Android izvršno okruženje (ART Android runtime).
- Element `<TextView>` se koristi za prikazivanje teksta.