

Predavanje br. 4

Razvoj mobilnih aplikacija i servisa

dr. sci. Alma Šećerbegović, van. prof.

Sadržaj predavanja

01

Android platforme - osnove

Upoznavanje sa Android operativnim sistemom, njegovom arhitekturom i osnovnim karakteristikama

02

Hello World

Kreiranje prve Android aplikacije i razumijevanje osnovne strukture projekta

03

02

Android Studio

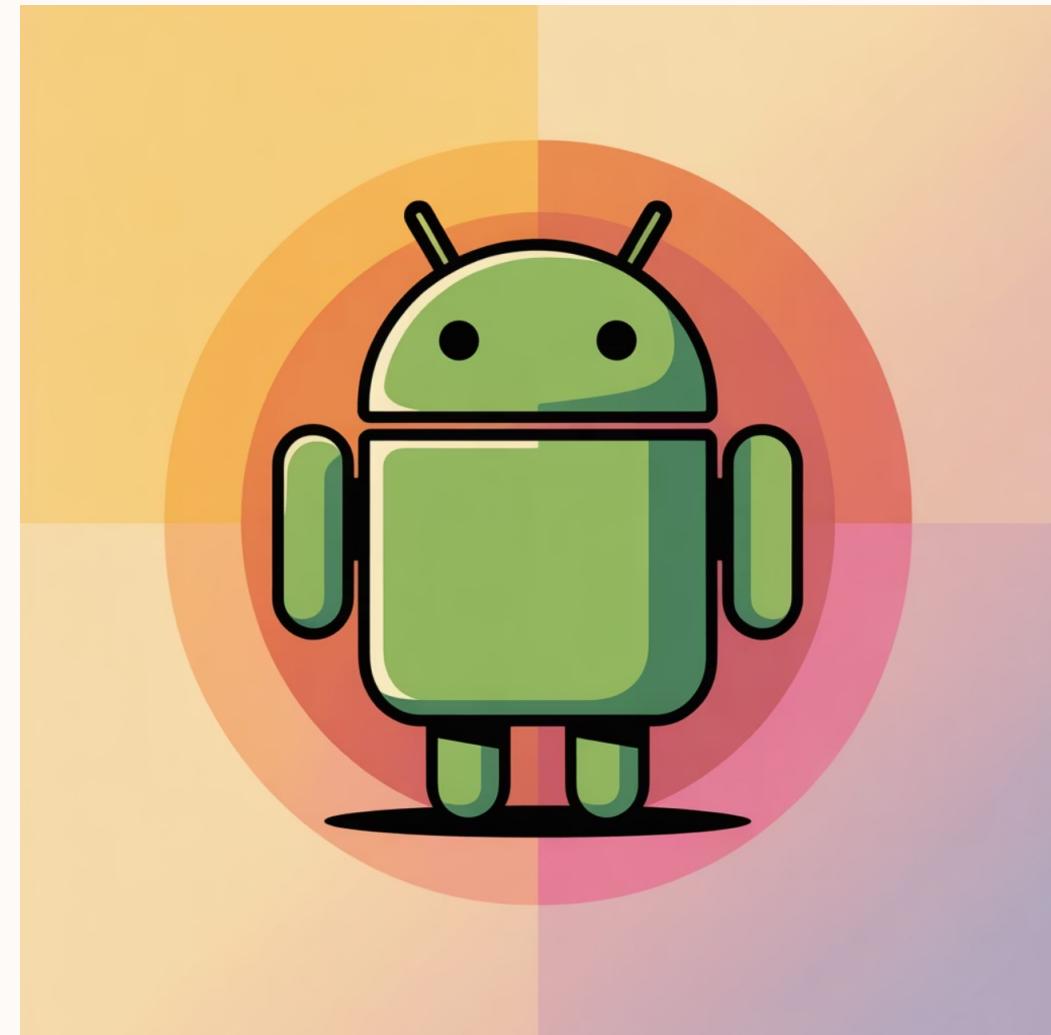
Alat za razvoj Android aplikacija i postavljanje razvojnog okruženja

Šta je Android?

Android je **operativni sistem zasnovan na Linux-u**, koji predstavlja jednu od najrasprostranjenijih mobilnih platformi u svijetu. Za razliku od tradicionalnih Linux distribucija, Android donosi određena ograničenja u pristupu:

- Nema shell pristupa po defaultu
- Nema pristupa root korisniku
- GNU C biblioteka (glibc) je zamijenjena sa Bionic bibliotekom, optimizovanom za mobilne uređaje

Android je **platforma otvorenog koda**, što znači da je izvorni kod dostupan na <https://source.android.com>. Međutim, važno je napomenuti da je samo operativni sistem otvoren - Google Play Store i Google Play Services su vlasnički softver Google LLC.



Android - historija: Rane verzije (2008-2010)

Android je prošao kroz značajan razvojni put od svojih prvih dana. Evo pregleda prvih verzija koje su postavile temelje moderne Android platforme:

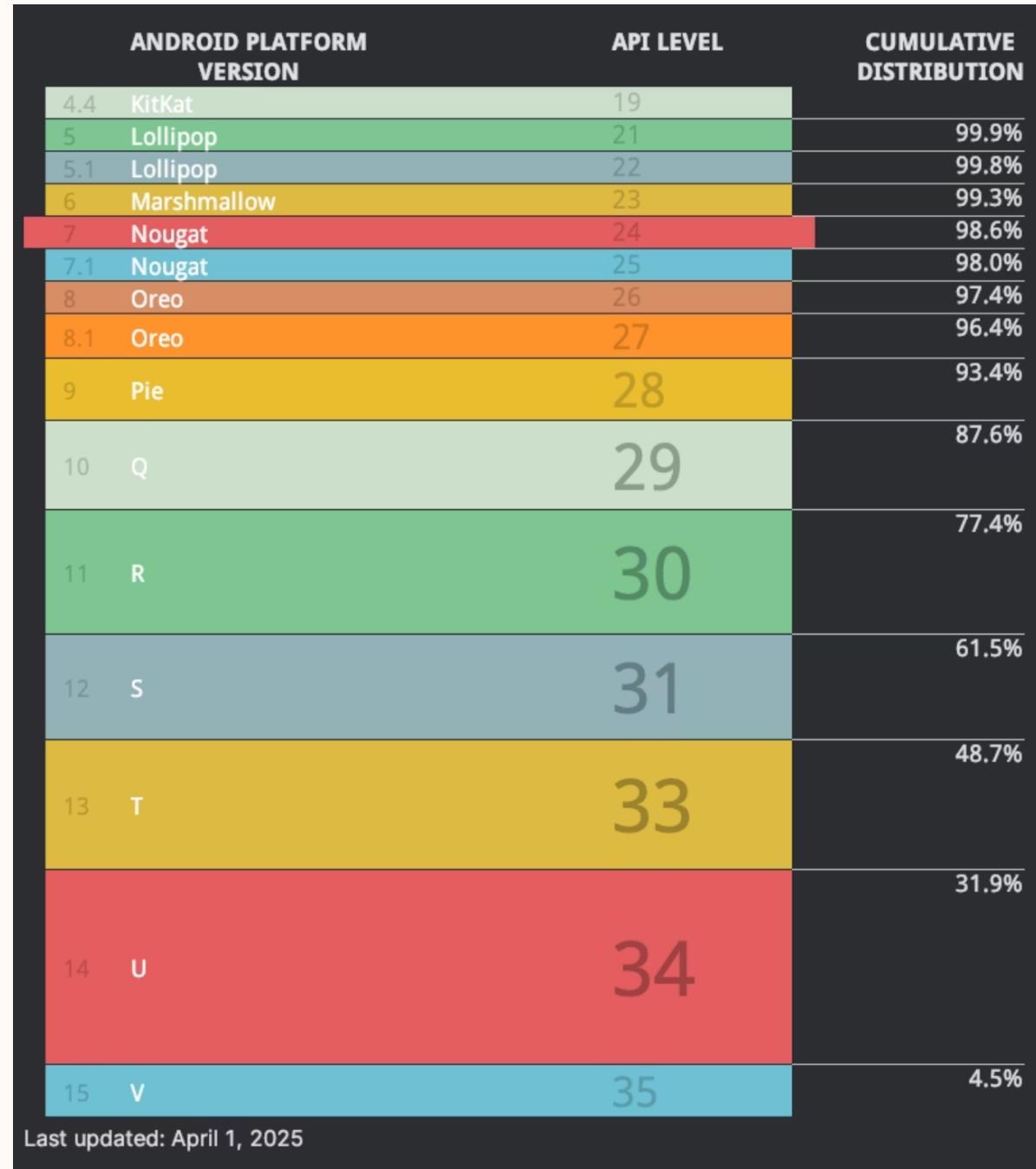


Android - historija: Era sazrijevanja (2011-2015)

Ovaj period donio je revolucionarne promjene u dizajnu i funkcionalnosti Android platforme, uključujući prelazak na moderne standarde korisničkog interfejsa.

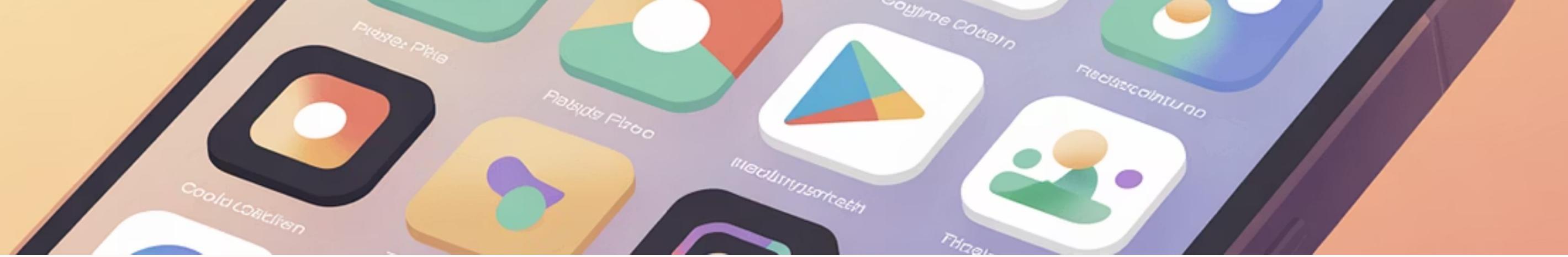


Android verzije: Moderna era (2016-2025)



Posljednjih godina Android je postao stabilnija i sigurnija platforma sa fokusom na privatnost korisnika:

- **Android 7.0-7.1.2 Nougat** (22. august 2016) - Multi-window podrška
- **Android 8.0-8.1 Oreo** (21. august 2017) - Notification channels
- **Android 9.0 Pie** (6. august 2018) - Gesture navigacija
- **Android 10.0** (3. septembar 2019) - Dark tema, pojačana privatnost
- **Android 11.0** (8. septembar 2020) - Kontrola uređaja, chat bubbles
- **Android 12.0** (19. oktobar 2021) - Material You dizajn
- **Android 13.0** (15. august 2022) - Granularne dozvole za medije
- **Android 14.0** (4. oktobar 2023) - Unaprijeđena personalizacija
- **Android 15.0** (3. septembar 2024) - Unaprijeđena privatnost
- **Android 16.0** (10. juni 2025) - live updates, grupirane notifikacije



Android ekosistem i modeli monetizacije

Android ekosistem nudi raznovrsne mogućnosti za distribuciju i monetizaciju aplikacija kroz Google Play prodavnicu, koja je dostupna u većini zemalja svijeta.

Besplatne aplikacije

Najpopularniji model, aplikacije se preuzimaju bez naknade. Prihod se ostvaruje kroz oglase ili druge metode.

Plaćene aplikacije

Korisnici plaćaju jednokratnu naknadu za preuzimanje. Manje popularan model kod Android korisnika.

In-app kupovine

Pretplate i jednokratna plaćanja unutar aplikacije za dodatne funkcionalnosti ili sadržaj.

Oglašavanje

Dominantan model monetizacije jer Android korisnici pokazuju manju sklonost plaćanju za aplikacije.

Izbor odgovarajućeg modela monetizacije zavisi od tipa aplikacije, ciljne publike i tržišnih trendova. Kombinacija više modela često daje najbolje rezultate.

Android prednosti: Zašto izabrati ovu platformu?

Android nudi brojne prednosti koje ga čine atraktivnom platformom za programere i korisnike širom svijeta.

Tehnološke prednosti

- **Otvoren kod** - Potpuna transparentnost i mogućnost prilagođavanja
- **Velika korisnička baza** - Milijarde aktivnih korisnika globalno
- **Prilagodljivost** - Mogućnost instaliranja custom tastatura, launcher-a i tema
- **Multi-platformska podrška** - Alati dostupni na Linux-u, Windows-u i MacOS-u
- **Pixel referentni uređaji** - Google Pixel telefoni kao standard za razvoj

Prednosti za programere

- **Niski troškovi** - Jednokratna naknada od \$25 (nasuprot Apple-ovim \$99 godišnje)
- **Fleksibilna distribucija** - Alpha, beta kanali i fazna distribucija aplikacija
- **Izvještavanje o greškama** - Automatski crash reports i Android Vitals
- **Pre-launch izvještaji** - Testiranje prije objavljivanja
- **Ogromna open source zajednica** - Biblioteke kao što su OkHttp, Retrofit, Ktor, Dagger, Hilt, Flipper, RxJava

Android nedostaci i izazovi

Uprkos svojoj popularnosti, Android platforma suočava se sa nizom izazova koji mogu otežati razvoj i održavanje aplikacija.

Fragmentacija

Ogromna raznolikost uređaja sa različitim karakteristikama predstavlja veliki izazov za programere koji moraju testirati aplikacije na brojnim konfiguracijama.

Sporo usvajanje novih verzija

Mnogi korisnici ostaju na starijim verzijama OS-a godinama, što otežava korištenje novih API-ja i funkcionalnosti.

Različite veličine ekrana

Od malih telefona do velikih tableta - dizajniranje responsivnog interfejsa zahtijeva dodatni napor.

Razlike u performansama

Različiti procesori i količine RAM memorije zahtijevaju pažljivo optimizovanje koda.

Uređaji nižeg kvaliteta

Pristup jeftinim Android uređajima znači da aplikacije moraju raditi i na slabijim hardverskim konfiguracijama.

Aplikacije lošeg kvaliteta

Otvorenost platforme omogućava objavljivanje aplikacija upitnog kvaliteta, poput simulatora bežičnog punjenja i sličnih prevara.

Android sigurnost: Višeslojni pristup zaštiti

Android implementira sofisticiran sigurnosni model koji štiti podatke korisnika i integriteta sistema kroz različite mehanizme.



Zaštita root pristupa

Root korisnik nije dostupan po defaultu. Root-ovanje uređaja radi dobijanja dodatnih funkcionalnosti često poništava garanciju.



Sistem dozvola (< API 23)

Stariji pristup gdje se sve dozvole odobravaju tokom instalacije aplikacije. Korisnik je morao prihvati sve dozvole odjednom.



Runtime dozvole (\geq API 23)

Moderan pristup gdje korisnik odobrava "opasne" dozvole tokom korištenja aplikacije, što povećava transparentnost i kontrolu.



TargetSDK uticaj

Verzija TargetSDK-a direktno utiče na ponašanje dozvola i drugih sigurnosnih aspekata aplikacije.



Google Play Bouncer

Automatski servis koji kontinuirano skenira Google Play prodavnicu za zlonamjerne aplikacije i štetni kod.

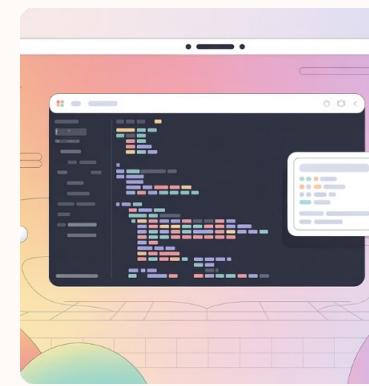
Alati za razvoj Android aplikacija

Razvoj Android aplikacija zahtijeva niz specijalizovanih alata koji omogućavaju efikasan rad, testiranje i optimizaciju koda.



Android Studio

Zvanično integrisano razvojno okruženje (IDE) za Android razvoj, zasnovano na IntelliJ IDEA platformi.



Android SDK

Skup alata uključujući ADB (Android Debug Bridge), Lint, Emulator, aapt, aidl, dx, D8 kompajler i druge esencijalne komponente.



ProGuard/R8

Alati za optimizaciju koda, obfuskaciju i smanjivanje veličine aplikacije kroz uklanjanje nekorištenog koda.



Android NDK

Native Development Kit omogućava razvoj na C/C++ jeziku, cross compile i integraciju native biblioteka za maksimalne performanse.

Gradle: Build sistem za Android

Gradle je moćan alat za automatizaciju build procesa koji predstavlja osnovu modernog Android razvoja. Ovaj fleksibilan i skalabilan sistem dizajniran je da podržava razvoj na više jezika, uključujući Java, Kotlin i C++.

Upravljanje zavisnostima

Automatski upravlja spoljnim bibliotekama i zavisnostima kroz jednostavnu deklarativnu sintaksu, eliminajući potrebu za ručnim preuzimanjem i konfiguracijom.

Automatizacija build procesa

Kompletna automatizacija kompilacije, pakovanja, testiranja i distribucije Android aplikacija, što značajno ubrzava razvojni ciklus.

Prilagodljivost

Visoko prilagodljiv kroz build skripte napisane u Groovy ili Kotlin jeziku, omogućava kreiranje custom taskova i konfiguracija.

Inkrementalni build-ovi

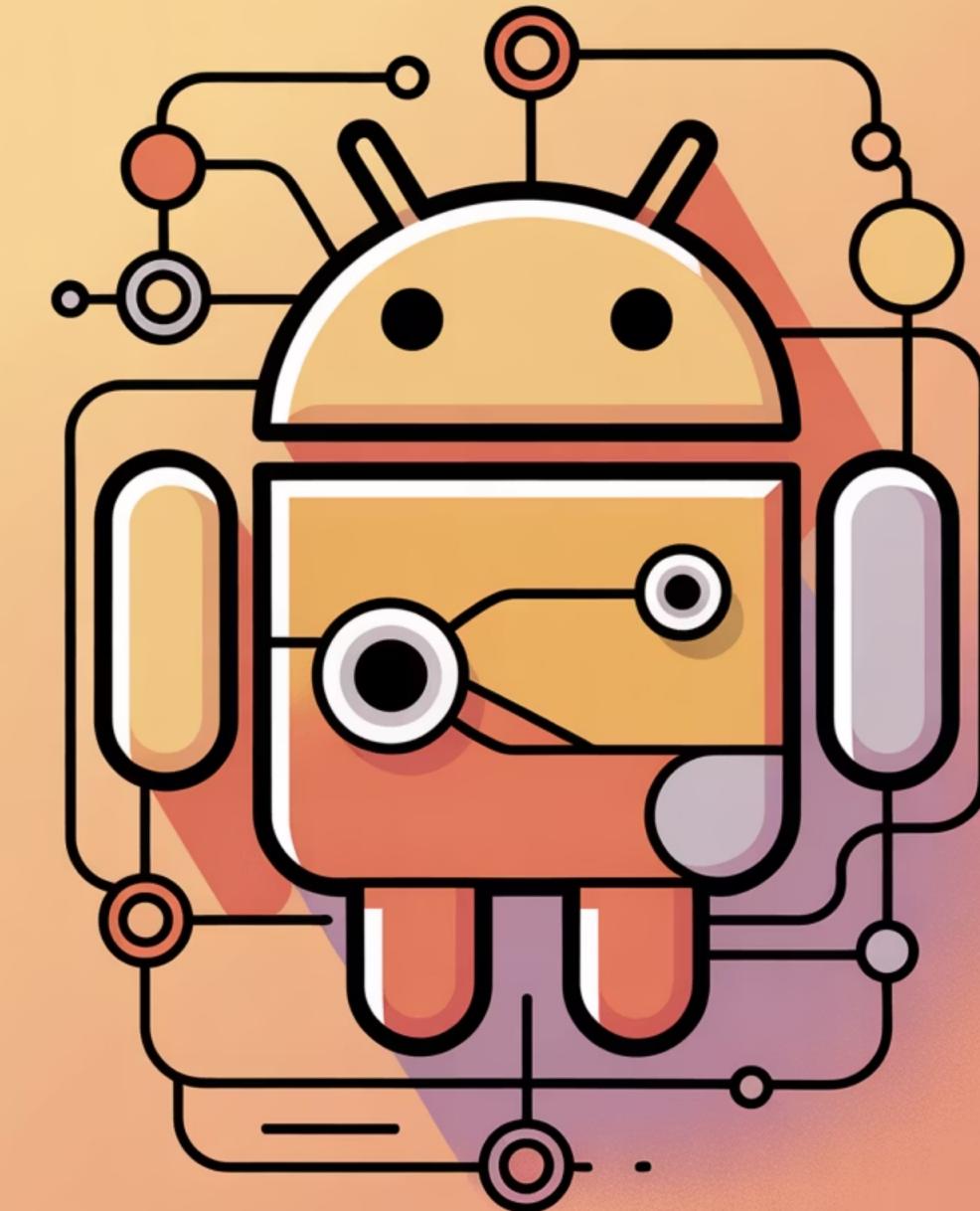
Optimizuje vrijeme build-a tako što rekompajlira samo ono što je promijenjeno, što drastično skraćuje vrijeme razvoja.

Ključne karakteristike

- Plugin sistem** - Lako dodavanje funkcionalnosti kroz Android plugin i druge ekstenzije
- Task-based arhitektura** - Definisanje i konfiguracija taskova za specifične korake build procesa
- Odlična integracija** - Bezobzirna integracija sa Android Studio-m za efikasan razvojni tok

Android Arhitektura

Razumijevanje osnovnih komponenti i strukture Android operativnog sistema.



Android kao multi-user sistem

Android je **više-korisnički Linux operativni sistem** kreiran od strane Open Handset Alliance-a i predvođen od strane Google-a. Ovaj pristup obezbeđuje visok nivo sigurnosti i izolacije između aplikacija.

Jedinstveni korisnički ID

Svaka aplikacija dobija svoj vlastiti Linux korisnički ID (UID) prilikom instalacije, što je osnova za izolaciju.

Sandbox okruženje

Svakoj aplikaciji je dodijeljen svoj "sandbox" gdje su fajlovi podešeni tako da samo ta aplikacija može da ih čita i piše.

Virtuelna mašina

Svaka aplikacija radi u svojoj vlastitoj virtuelnoj mašini, tj. kod se izvršava potpuno izolovano od drugih aplikacija.

Zaseban proces

Svaka aplikacija je zaseban Linux proces sa svojim resursima, što sprečava da problemi u jednoj aplikaciji utiču na druge.

Ovaj multi-layer pristup sigurnosti čini Android jednom od najsigurnijih mobilnih platformi, gdje kompromitovanje jedne aplikacije ne znači automatski kompromitovanje cijelog sistema.

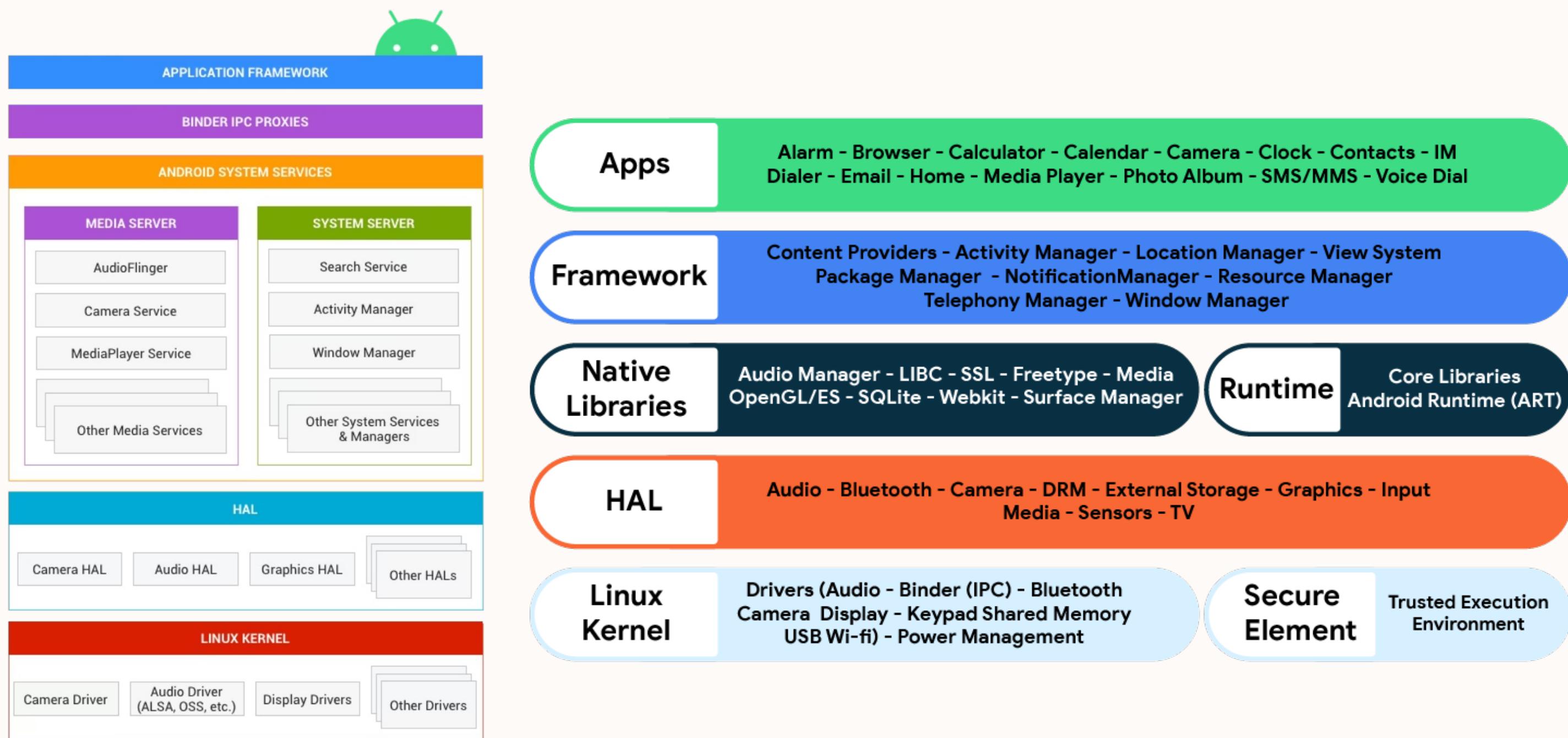
Android arhitektura: Slojeviti pristup

Android arhitektura predstavlja sofisticiran slojeviti sistem gdje svaki sloj pruža specifične funkcionalnosti i servise višim slojevima. Ovaj pristup omogućava fleksibilnost, sigurnost i optimizaciju performansi na širokom spektru uređaja.

Arhitektura se može podijeliti na nekoliko ključnih nivoa:



Detaljna Android arhitektura



Osnove Android platforme

Android aplikacije generalno su napisane u **Kotlin-u i/ili Java-i**, mada postoje opcije za razvoj na C, C++ i drugim jezicima kroz Android NDK.

Ključne karakteristike

- **APK format** - Kompajlirana Android aplikacija dolazi u .apk formatu (Android Package)
- **Digitalni potpis** - Sve aplikacije moraju biti digitalno potpisane kako bi se mogle izvršavati na uređaju
- **Debug certifikat** - Za vrijeme razvoja koristi se automatski debug certifikat koji dolazi sa instalacijom Android Studio-a
- **Release certifikat** - Za objavljivanje aplikacija potreban je privatni keystore certifikat



 **Važno:** Izgubljeni release keystore certifikat znači da više nećete moći ažurirati vašu aplikaciju na Google Play prodavnici, pa je kritično čuvati backup!

Glavne komponente Android aplikacija

Android aplikacije se sastoje od posebnih građevnih blokova koji omogućavaju različite funkcionalnosti. Razumijevanje ovih komponenti je ključno za razvoj robusnih aplikacija.



Activities

Predstavljaju jedan ekran sa korisničkim interfejsom (UI). Svaka aktivnost je nezavisna komponenta koja može interagovati sa korisnikom.



Services

Predstavljaju proces koji se izvršava u pozadini bez korisničkog interfejsa, kao što je reprodukcija muzike ili preuzimanje podataka.



Content Providers

Omogućavaju dijeljenje podataka između aplikacija kroz standardizovan interfejs, čuvajući podatke i omogućavajući siguran pristup.

Broadcast Receivers

Slušaju i reaguju na sistemske poruke (broadcasts) na nivou cijelog sistema, kao što je punjenje baterije ili promjena mrežne konekcije.



Application

Bazna klasa koja sadrži globalno stanje aplikacije i koordinira aktivnosti koje čine koherentnu cjelinu.

Intent

Mehanizam za prenos poruka između komponenti, omogućava pokretanje aktivnosti, servisa i slanje broadcast poruka.

Activity: Temelj korisničke interakcije

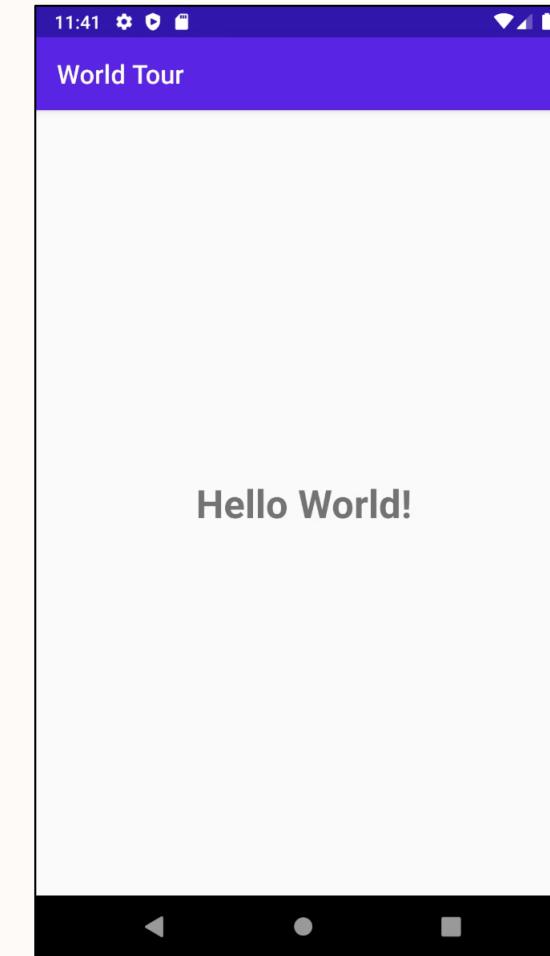
Aktivnost (Activity) je fundamentalna komponenta svake Android aplikacije koja predstavlja jedan ekran sa kojim korisnik interaguje.

Osnovne karakteristike

- Svaka aktivnost je **jedan ekran** vaše aplikacije sa svojim korisničkim interfejsom
- Aplikacija je zapravo **zbir povezanih aktivnosti** koje zajedno formiraju korisničko iskustvo
- Možete smatrati svaku aktivnost i **ekranom i funkcijom** - vizuelnom i logičkom jedinicom
- Aktivnosti mogu biti **nezavisne** - jedna aplikacija može pokrenuti aktivnost druge aplikacije

Primjeri aktivnosti

- Login ekran
- Lista proizvoda
- Detalji proizvoda
- Forma za plaćanje
- Postavke aplikacije



❑ **Važno:** Aktivnosti imaju složen životni ciklus (`onCreate`, `onStart`, `onResume`, `onPause`, `onStop`, `onDestroy`) koji programeri moraju razumjeti za pravilno upravljanje resursima.

Services: Rad u pozadini

Servis (Service) je komponenta koja omogućava izvršavanje dugotrajnih operacija u pozadini, bez potrebe za korisničkim interfejsom. Ovo je ključna komponenta za funkcionalnosti koje trebaju raditi čak i kada korisnik ne interaguje aktivno sa aplikacijom.



Reprodukcia muzike

Najčešći primjer - muzika nastavlja da svira čak i kada korisnik pređe na drugu aplikaciju ili zaključa telefon.



Praćenje lokacije

Kontinuirano prikupljanje GPS podataka za navigaciju, fitness aplikacije ili praćenje dostave.



Preuzimanje podataka

Download velikih fajlova ili sinhronizacija podataka sa serverom u pozadini bez blokiranja korisničkog interfejsa.



Push notifikacije

Slušanje i obrada dolazećih poruka i notifikacija sa remote servera.

Postoje dva tipa servisa: **Started Services** (pokreću se i rade dok ne završe zadatak) i **Bound Services** (druge komponente se vezuju za njih i komuniciraju).

Content Provider: Dijeljenje podataka

Content Provider je komponenta koja upravlja zajedničkim skupom podataka aplikacije i omogućava sigurno dijeljenje tih podataka sa drugim aplikacijama kroz standardizovan interfejs.

Karakteristike

- **Centralizovano upravljanje** - Sve operacije sa podacima prolaze kroz kontrolisani interfejs
- **Fleksibilno skladištenje** - Podaci mogu biti u SQLite bazi, fajlovima, remote servisu ili bilo kom drugom izvoru
- **Kontrolisan pristup** - Aplikacije moraju imati odgovarajuće dozvole za pristup podacima
- **Privatnost** - Content provideri mogu biti privatni (dostupni samo vlastitoj aplikaciji) ili javni



Primjeri korištenja

- **Kontakti** - Pristup telefonskom imeniku
- **Kalendar** - Čitanje i pisanje događaja
- **Medija** - Pristup slikama, audio i video fajlovima
- **Custom podaci** - Dijeljenje podataka vaše aplikacije sa drugim aplikacijama

Content provideri koriste **URI (Uniform Resource Identifier)** šemu za identifikaciju podataka, što omogućava elegantan i standardizovan način pristupa podacima iz različitih izvora.

Broadcast Receiver

Broadcast Receiver je ključna komponenta Android sistema koja omogućava aplikacijama da reaguju na sistemske najave. Ove najave se manifestuju kao Intents i predstavljaju važan komunikacijski mehanizam unutar Android ekosistema.



Sistemske najave

Informacije o statusu sistema se dostavljaju putem broadcast poruka - npr. uređaj okrenut na stranu, ekran isključen ili upaljeno



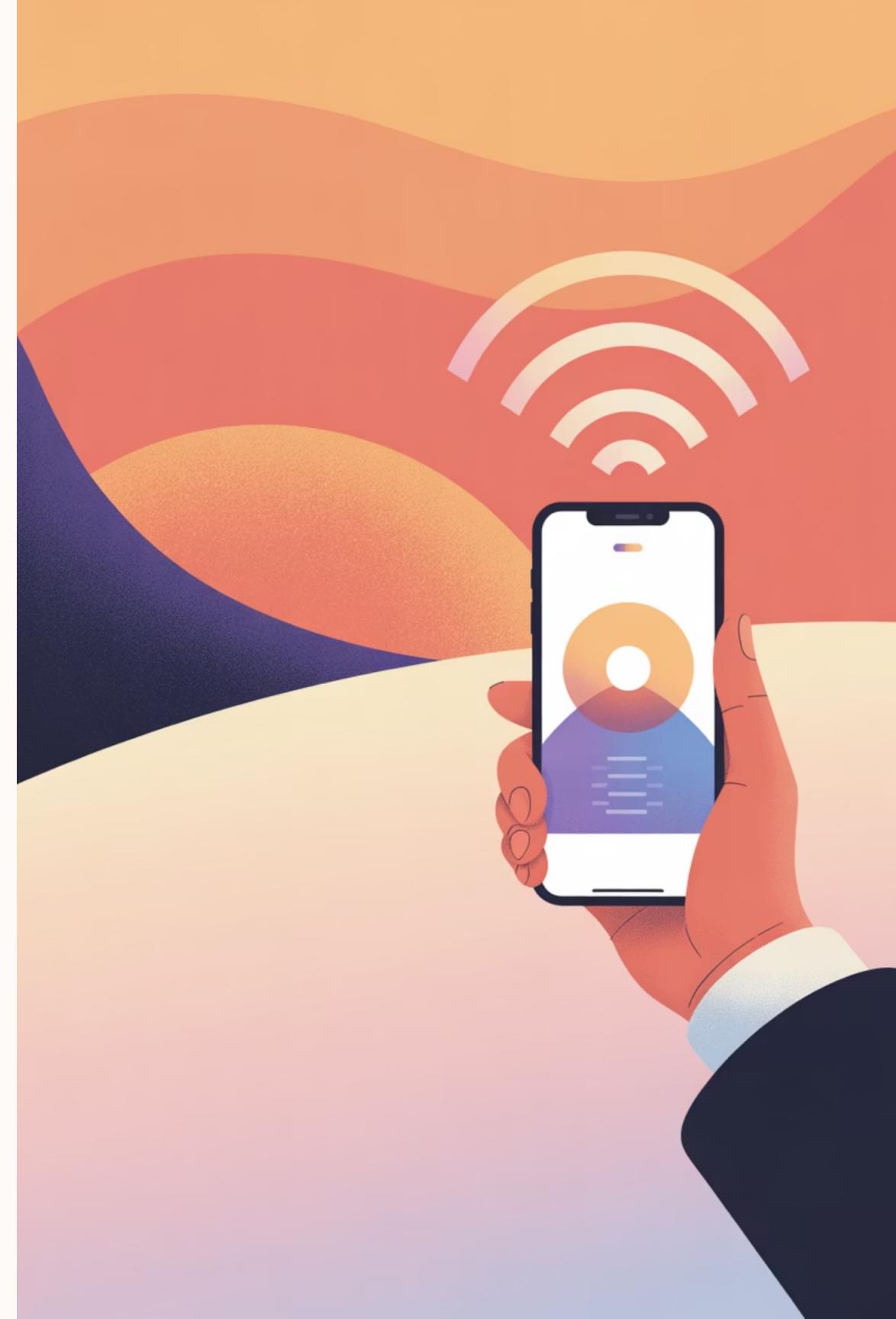
Stanje uređaja

Nadzor kritičnih događaja kao što su slaba baterija, dolazni telefonski poziv, promjena mrežne konekcije



Diskretni interfejs

Prijemnici emitovanja obično nemaju korisnički interfejs, ali mogu imati ikonu u statusnoj traci za indikaciju statusa



Intents

Intent je fundamentalna poruka koja traži određenu akciju od druge komponente Android sistema. Ovo je osnovni mehanizam za komunikaciju između komponenti aplikacija.

Intent sistem omogućava fleksibilnu i moćnu komunikaciju. To uključuje i "molimo pokrenite vašu aplikaciju" poruku koju sistem šalje kada korisnik klikne na ikonu vaše aplikacije na home screen-u.

01

Korisnik klikne ikonu

Sistem detektuje akciju korisnika

02

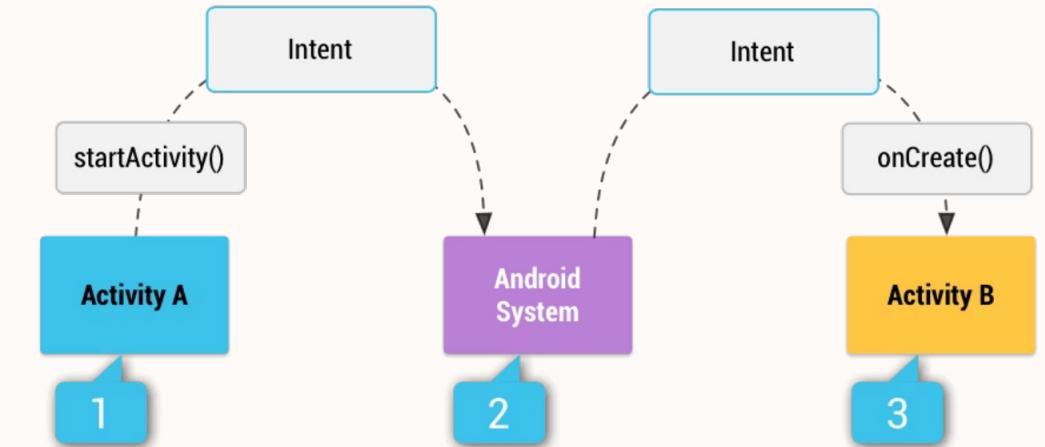
Intent se kreira

Sistemska poruka se priprema

03

Aplikacija se pokreće

Odgovarajuća komponenta reaguje



Intents omogućavaju modularnu arhitekturu gdje komponente mogu komunicirati bez direktnih referenci.

Povezane aplikacije

Zbog modularne prirode Android aplikacija koje se sastoje od Aktivnosti, Servisa i drugih komponenti, izuzetno je lako izgraditi funkcionalnosti vaše aplikacije koristeći postojeće sistemske komponente.

Korištenje kamere

Ako vaša aplikacija treba da snimi fotografiju, možete zatražiti od Aktivnosti kamere da obradi taj zahtjev i vrati rezultat u obliku slike

Pristup galeriji

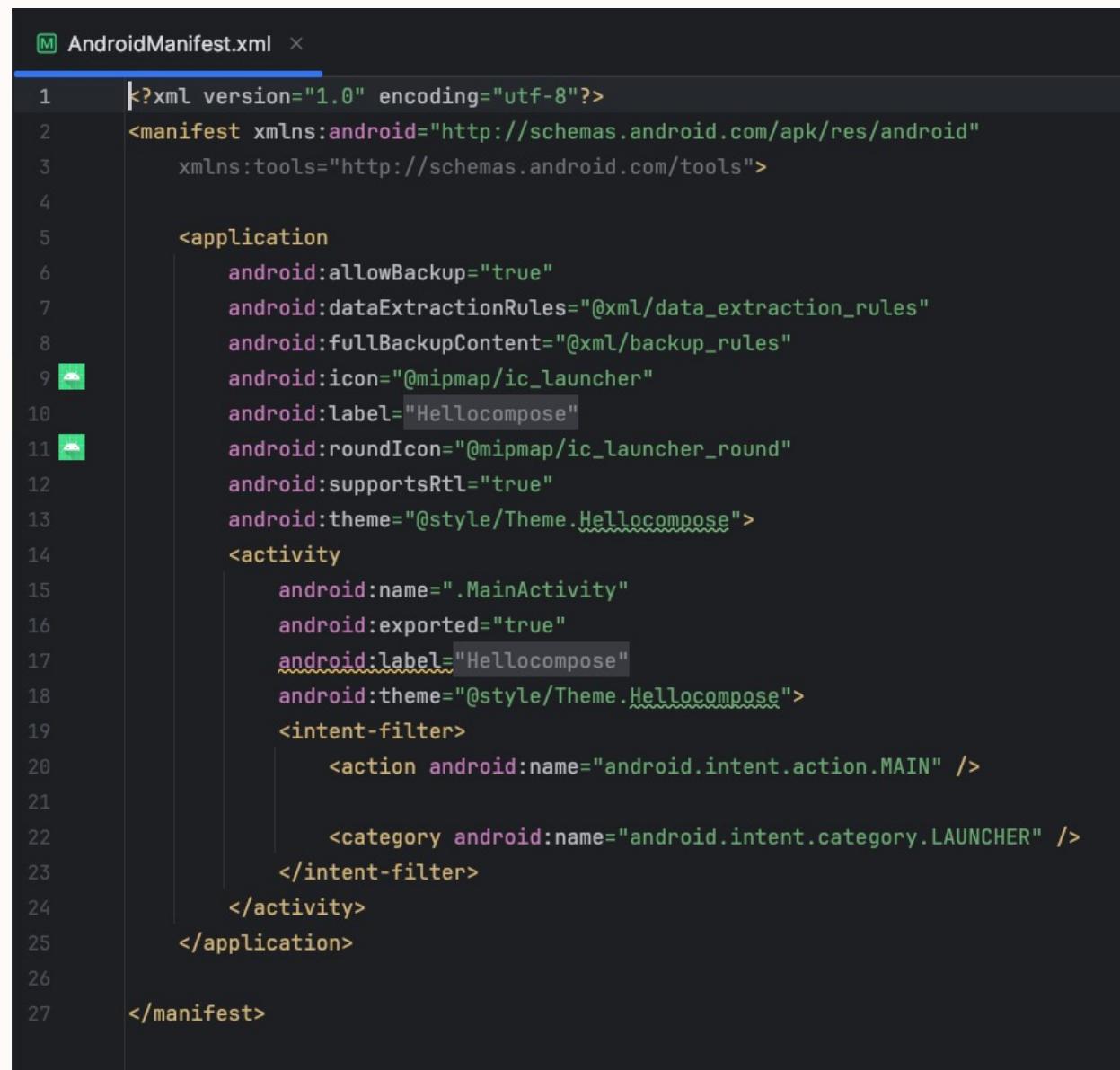
Otvaranje galerije slika putem Intent-a omogućava korisnicima da odaberu postojeće fotografije

Dijeljenje sadržaja

Intent sistem omogućava lako dijeljenje podataka između različitih aplikacija na uređaju

- ☐ **Ključna prednost:** Ovo se rješava korištenjem Intents - nema potrebe da sami implementirate složene funkcionalnosti kao što je fotografiranje ili snimanje videa.

Kako je to sve povezano?



```
<?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="Hellocompose"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Hellocompose">

        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:label="Hellocompose"
            android:theme="@style/Theme.Hellocompose">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

AndroidManifest.xml datoteka!

Ovo je centralna konfiguracijska datoteka koja povezuje sve komponente vaše Android aplikacije i definiše njene karakteristike.



Dozvole korisnika

Postavlja sve dozvole koje korisnik mora prihvati - internet, GPS, pristup kontaktima, kameri i drugim resursima



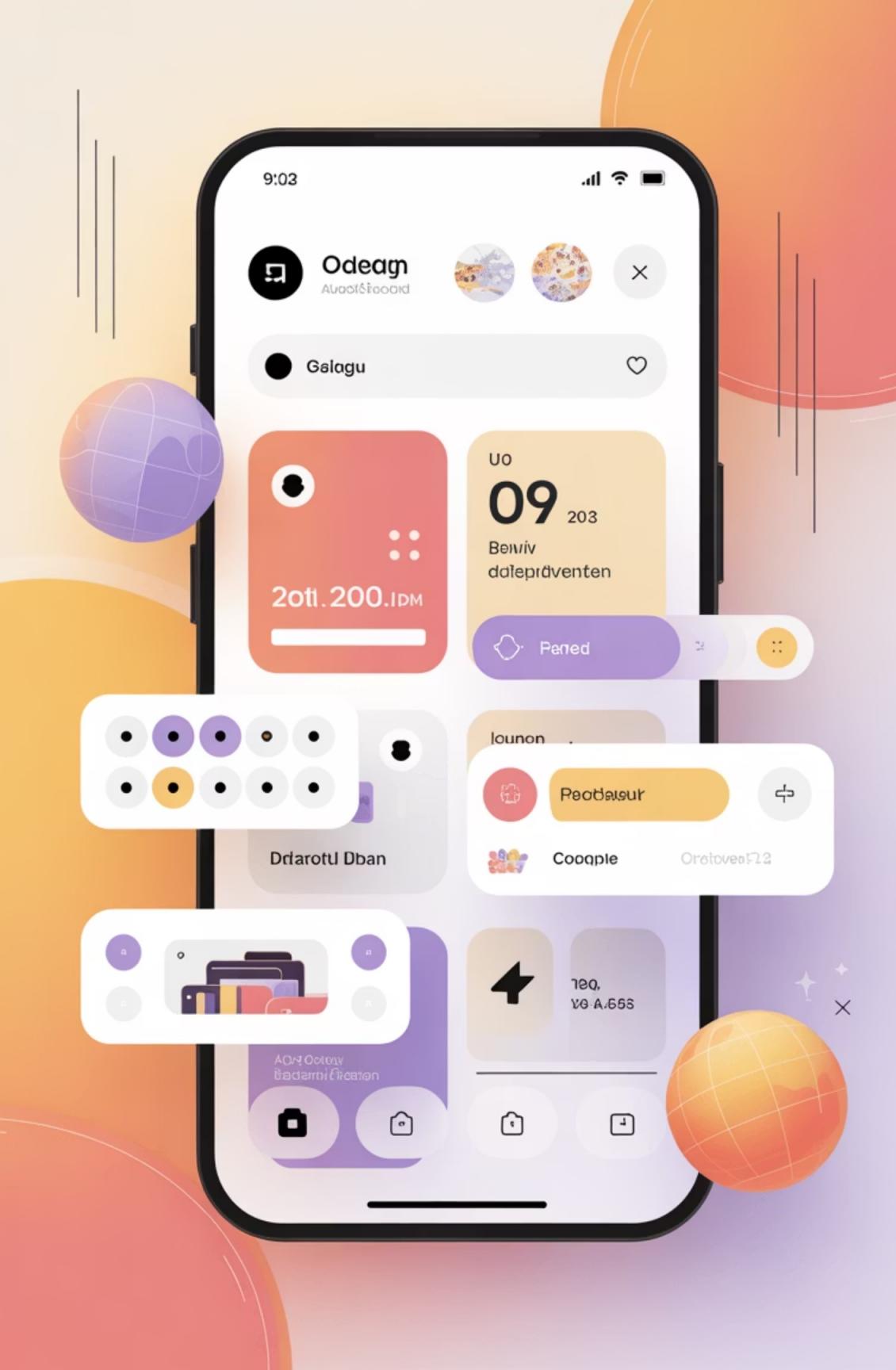
API konfiguracija

Deklariše nivo API-ja aplikacije, zahtijeva hardverske karakteristike koje su potrebne, i definiše potrebne biblioteke



Komponente aplikacije

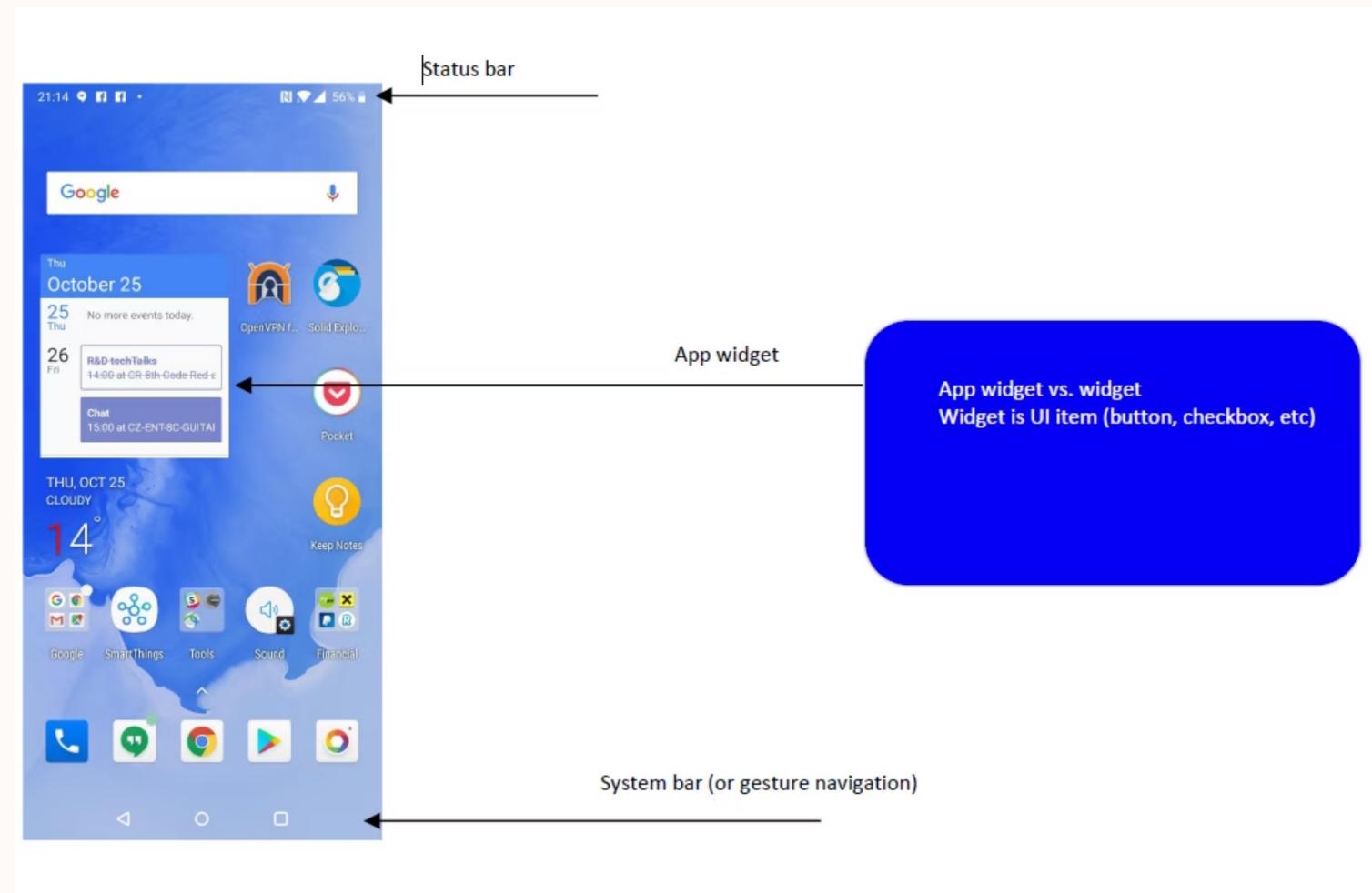
Određuje koje su aktivnosti, servisi i broadcast receiveri dio ove aplikacije



Android UI

Android korisnički interfejs (UI) je bogat ekosistem vizuelnih elemenata, navigacionih obrazaca i interaktivnih komponenti koji omogućavaju intuitivno korisničko iskustvo. Od launcher-a do notifikacija, svaki element igra važnu ulogu u komunikaciji između korisnika i aplikacije.

Launcher



Važna distinkcija

App Widget

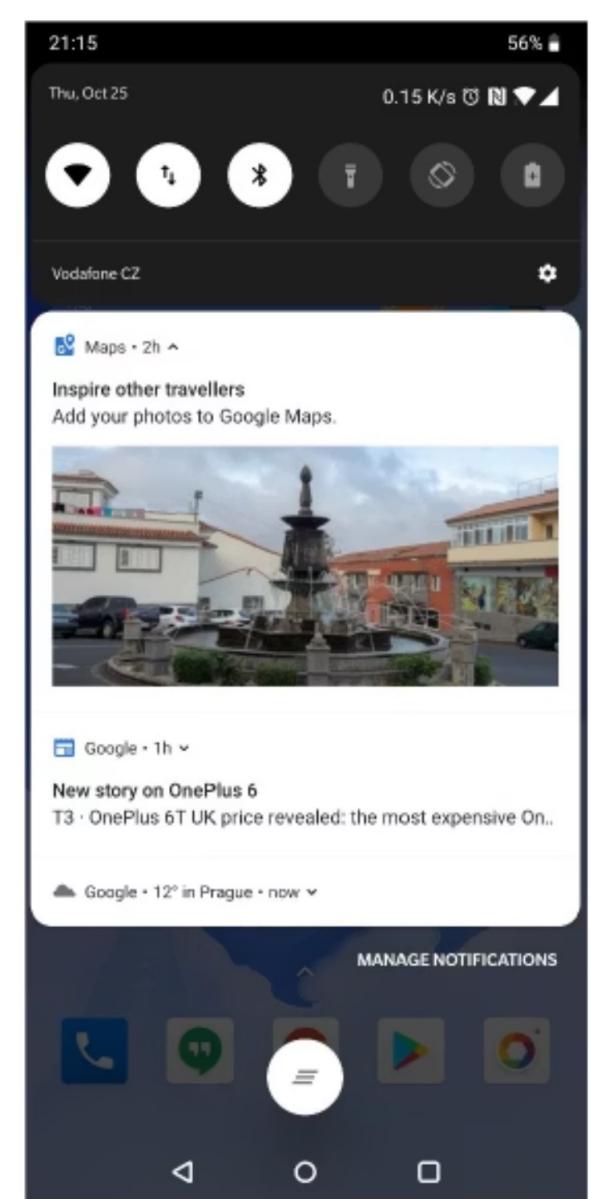
Mini aplikacija na home screen-u koja prikazuje informacije i omogućava brzu interakciju bez otvaranja pune aplikacije

Widget

UI element unutar aplikacije - button, checkbox, text field i drugi osnovni elementi interfejsa

Launcher je početni ekran Android uređaja gdje korisnici pristupaju svojim aplikacijama i widget-ima. To je centralno mjesto za organizaciju i pokretanje aplikacija.

Notifications



Notifikacije su esencijalni mehanizam za obavještavanje korisnika o važnim događajima - novi email, promjena statusa, poruke, i drugi relevantni podaci.



Osnovne notifikacije

Prikaz teksta, ikone i osnovnih informacija u notifikacionoj traci



Interaktivne akcije

Od API-16, notifikacije mogu sadržavati akcije koje korisnik može direktno izvršiti bez otvaranja aplikacije



Lockscreen prisustvo

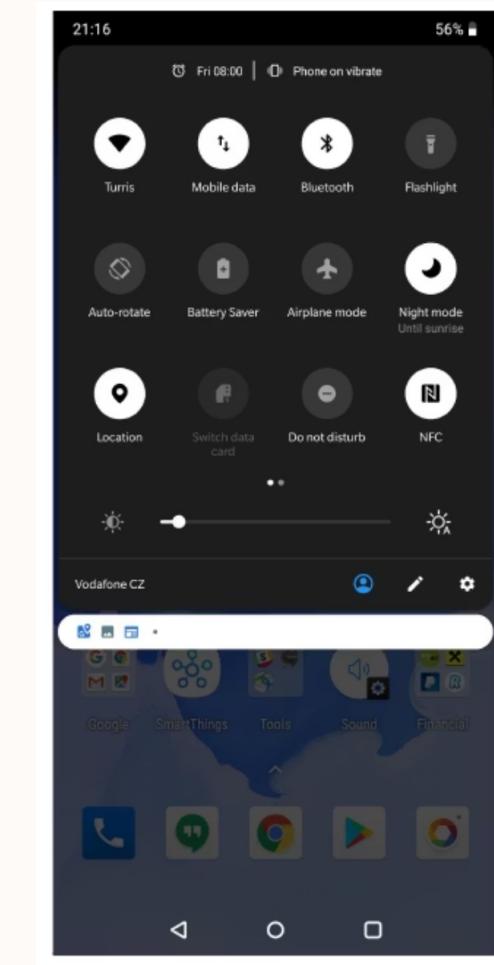
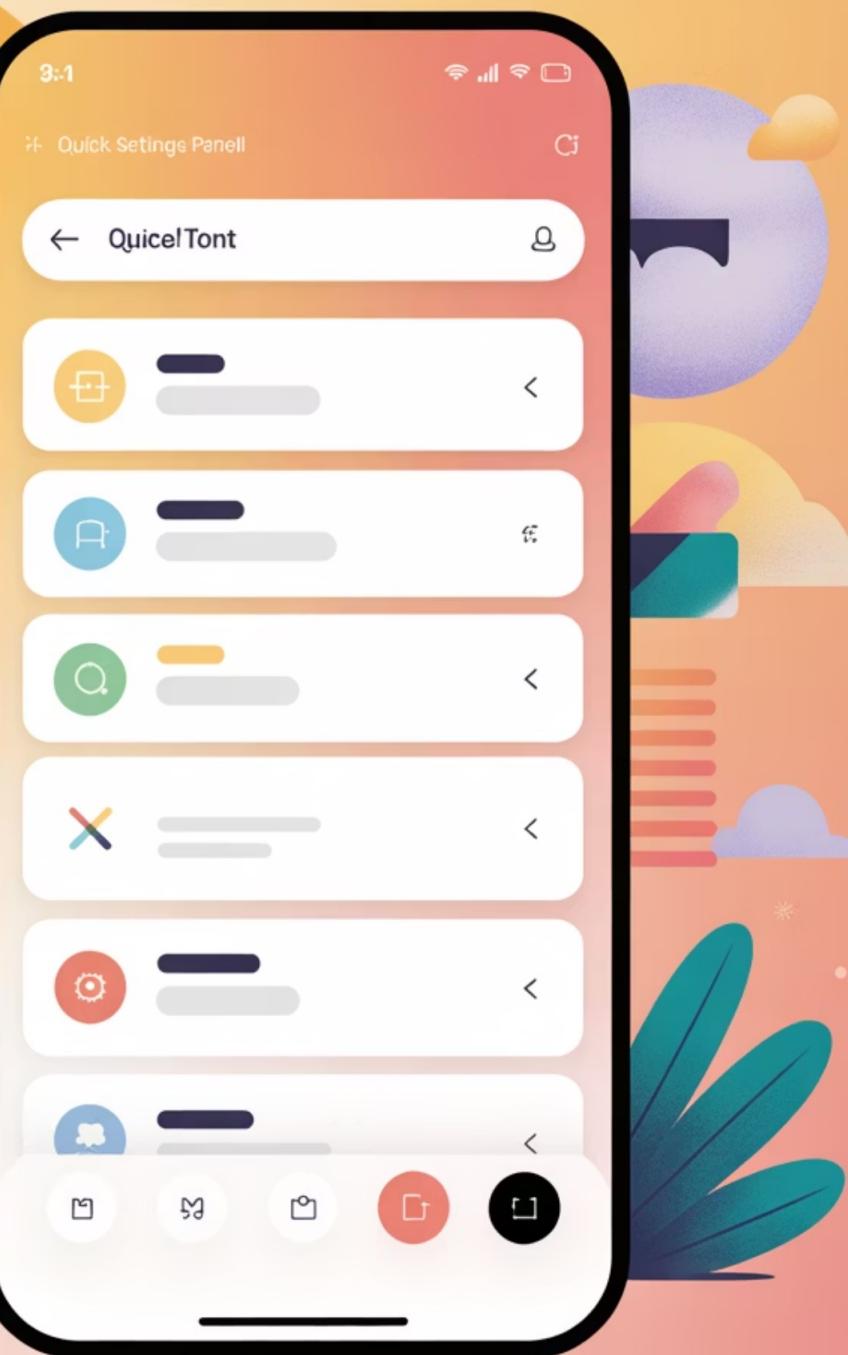
Notifikacije mogu biti vidljive na zaključanom ekranu za brz pregled informacija



Multimedijalni feedback

Opcionalni zvuk, vibracija i LED svjetlo privlače pažnju korisnika na različite načine

Quick Settings



Brza podešavanja

Quick Settings panel omogućava korisnicima brz pristup najčešće korištenim postavkama sistema - WiFi, Bluetooth, brightness, airplane mode i drugim kontrolama.

1

API-16

Quick Settings postaju dio AOSP (Android Open Source Project) i dostupni su na svim Android uređajima

2

API-24

Uvedene prilagođene akcije - aplikacije mogu dodavati vlastite tile-ove u Quick Settings panel za brz pristup specifičnim funkcijama

Ova evolucija omogućava aplikacijama da pruže korisnicima još bržu interakciju sa ključnim funkcionalnostima.

Navigacija unutar aplikacije

Navigacija je kritična komponenta korisničkog iskustva u Android aplikacijama. Dobro dizajnirana navigacija omogućava korisnicima da intuitivno pronađu željeni sadržaj i funkcije.

Action Bar

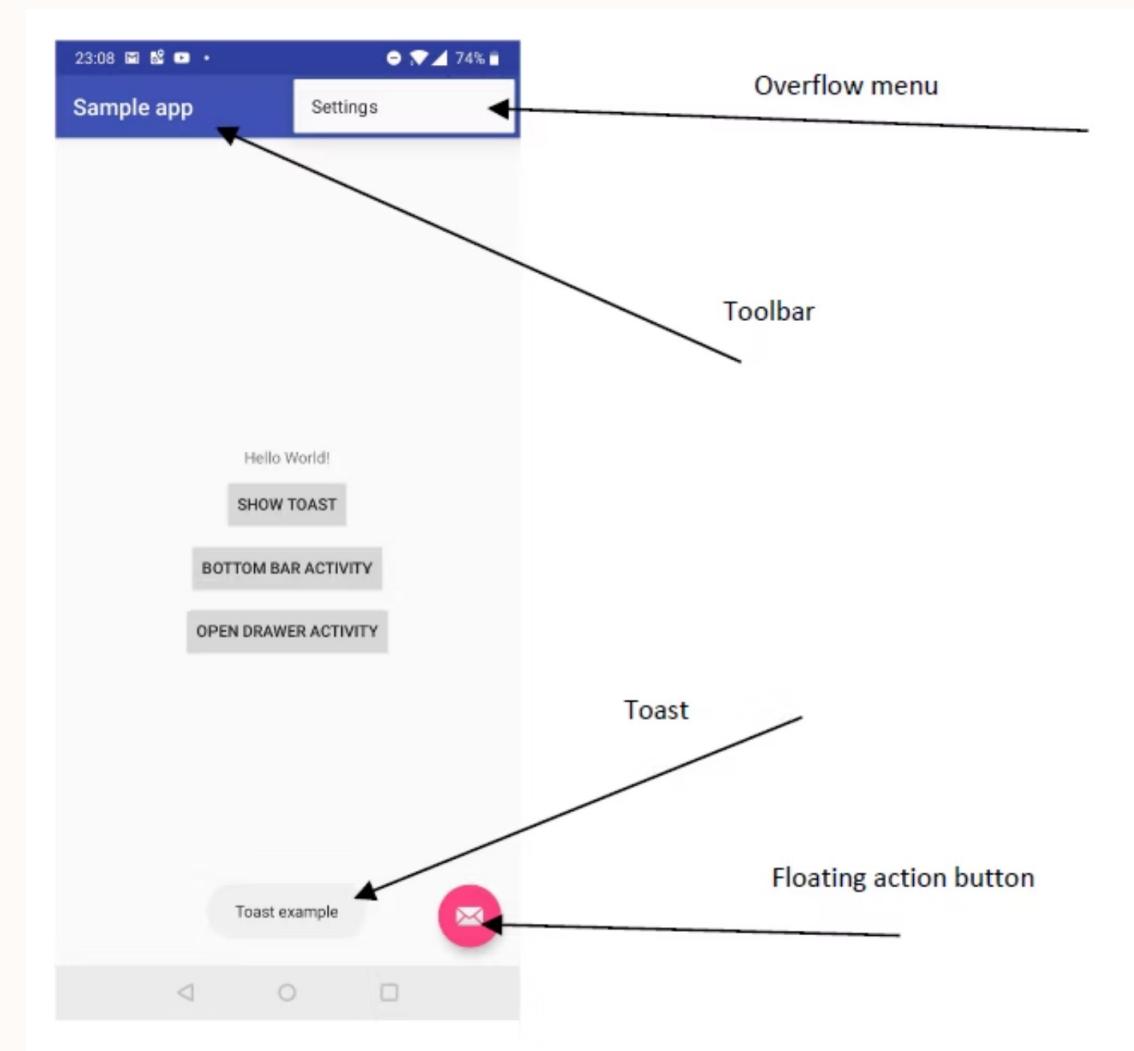
Gornja traka sa naslovom, ikonama akcija i navigacionim elementima

Tab Navigation

Tabovi omogućavaju brzo prebacivanje između različitih sekcija sadržaja

Gestures

Swipe i druge geste omogućavaju prirodnu navigaciju kroz ekrane



Različiti navigacioni obrasci se koriste zavisno od strukture aplikacije i korisničkih potreba. Kombinacija više pristupa često daje najbolje rezultate.

Navigacija - Bottom Bar

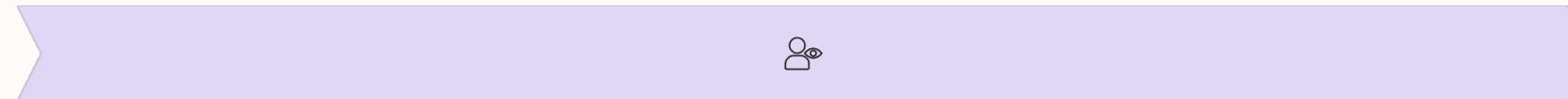
Bottom Navigation Bar

Bottom navigation je jedan od najpopularnijih navigacionih obrazaca u modernim Android aplikacijama. Smješten na dnu ekrana, omogućava brz pristup glavnim sekcijama aplikacije.



Pristupačnost

Lako dostupan palcem na većini veličina ekrana



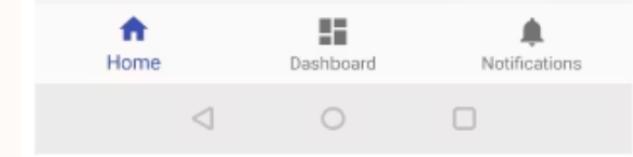
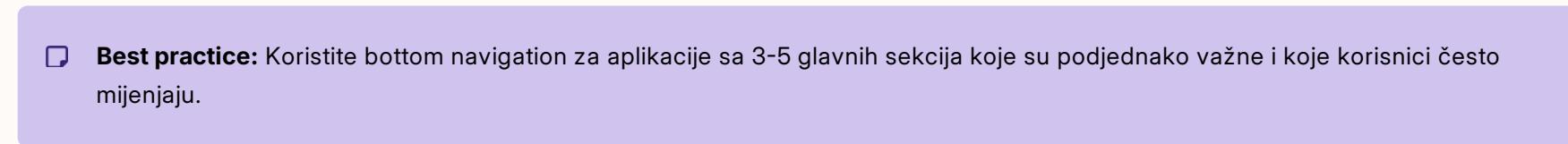
Vidljivost

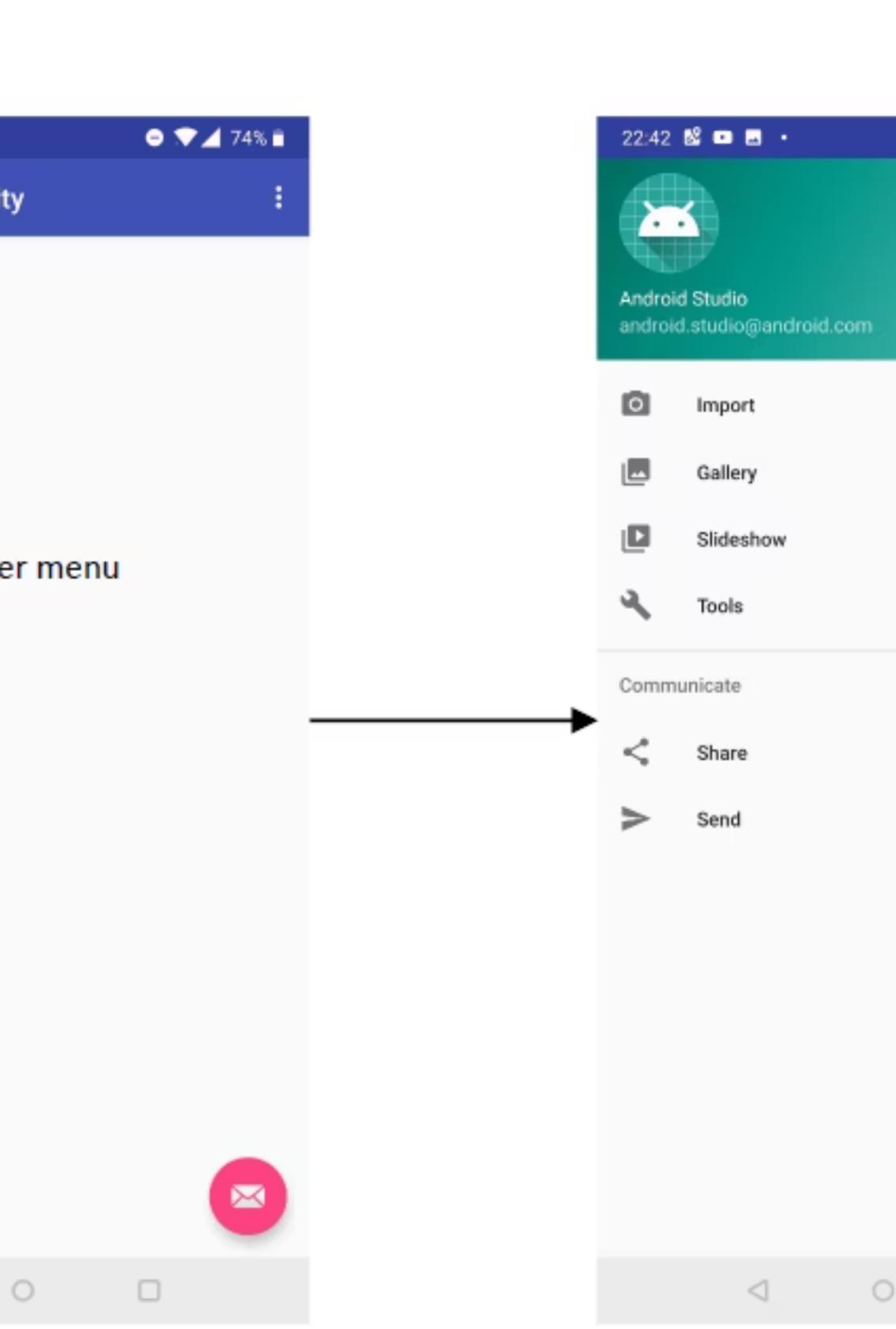
Uvijek prisutan, korisnici znaju gdje se nalaze u aplikaciji



Jednostavnost

Idealan za 3-5 top-level destinacija





Navigacija - Drawer

Navigation Drawer

Navigation drawer je panel koji se izvlači sa lijeve strane ekrana i sadrži navigacijske opcije. Ovo je idealno rješenje za aplikacije sa većim brojem destinacija ili složenom hijerarhijom.

Prednosti

- Podržava veliki broj navigacijskih opcija
- Čuva prostor na ekranu kada nije u upotrebi
- Može sadržavati korisničke informacije i postavke
- Fleksibilan za organizaciju različitih nivoa sadržaja

Kada koristiti

- Aplikacije sa više od 5 top-level destinacija
- Hijerarhijska struktura sa različitim nivoima
- Dodatne opcije kao što su postavke ili profil
- Manje frekventna navigacija između sekcija

Navigation drawer se obično aktivira swipe gestom sa lijeve ivice ili klikom na "hamburger" ikonu u gornjem lijevom uglu.

Material Design

Material Design je sveobuhvatni dizajnerski jezik razvijen od strane Google-a koji definiše principe, komponente i najbolje prakse za kreiranje konzistentnih i intuitivnih korisničkih interfejsa.



Materijal kao metafora

Inspirisan fizičkim svjetom - površine reflektuju svjetlo, bacaju sjene, i imaju taktilne kvalitete koje pomažu korisnicima da razumiju UI



Cross-platform

Dostupan na svim platformama - Android, iOS, Flutter, web aplikacije, omogućavajući konzistentan brand i UX



Material komponente

Bogata biblioteka ready-to-use komponenti koje ubrzavaju razvoj i osiguravaju kvalitet interfejsa

Material Design kontinuirano evoluira kroz verzije (Material Design 1, 2, i najnoviji Material You), ali osnovni principi ostaju konzistentni.

"Material is the metaphor. A material metaphor is the unifying theory of a rationalized space and a system of motion."

Posjetite <https://www.material.io/> za detaljnu dokumentaciju, komponente i resurse.

Vaša prva aplikacija

Sada kada smo razumjeli osnovne koncepte Android platforme, vrijeme je da kreiramo vašu prvu aplikaciju. Android Studio je službeni IDE (Integrated Development Environment) koji pruža sve alate potrebne za razvoj, testiranje i deploy Android aplikacija.



Otvoriti Android Studio

Nakon instalacije Android Studio-a, pokrenite aplikaciju. Dobit ćete ekran dobrodošlice sa nekoliko opcija za početak rada.

01

Pokrenite Android Studio

Kliknite na ikonu aplikacije na vašem računaru

02

Ekran dobrodošlice

Vidjet ćete opcije za kreiranje novog projekta, otvaranje postojećeg, ili pristup drugim funkcijama

03

Spremno za razvoj

Android Studio je sada spreman za kreiranje vaše prve aplikacije

- Napomena:** Prvi put kada pokrenete Android Studio, može biti potrebno nekoliko minuta za inicijalizaciju i konfiguraciju potrebnih komponenti.



Android Studio

+ Start a new Android Studio project

Open an existing Android Studio project

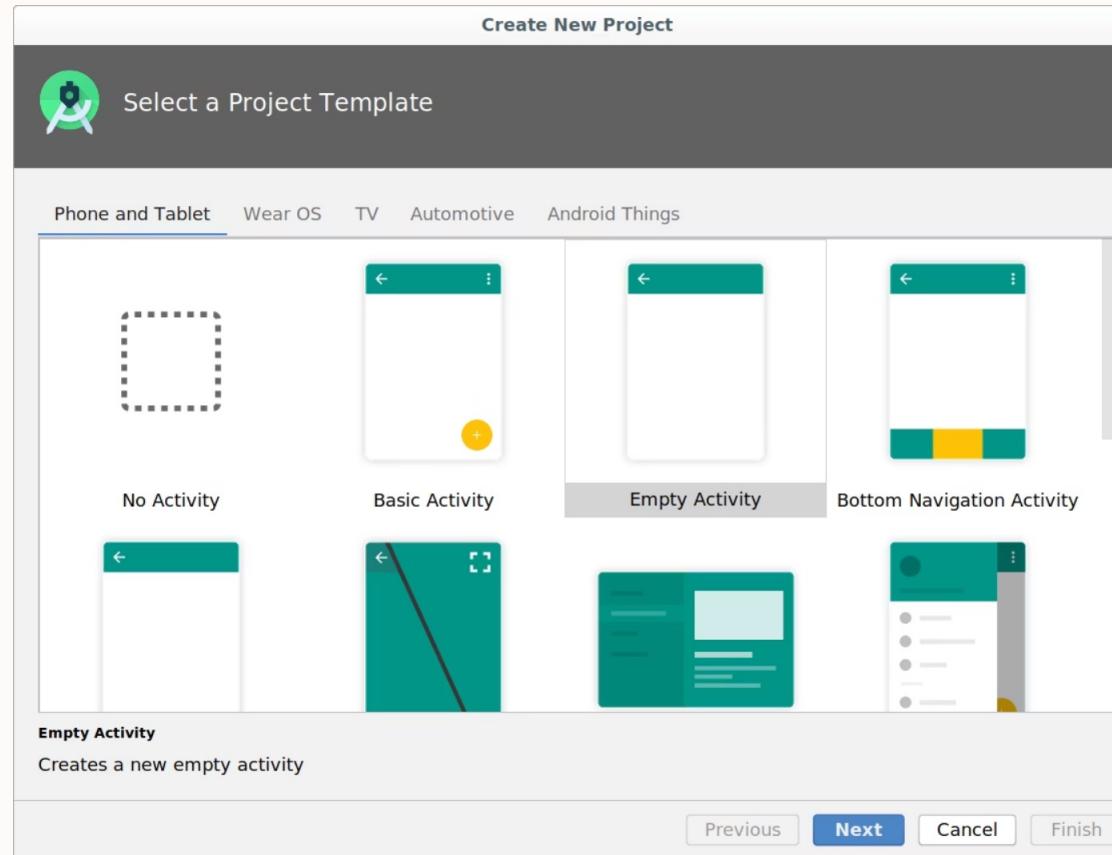
Check out project from Version Control ▾

Profile or debug APK

Import project (Gradle, Eclipse ADT, etc.)

Import an Android code sample

Kreirati novi projekat



Izbor template-a

Android Studio nudi različite template-e za početak projekta. Svaki template dolazi sa predefiniranom strukturom i osnovnim kodom.

- **Empty Activity**

Najjednostavniji template - dobra polazna tačka za učenje

- **Basic Activity**

Uključuje toolbar i floating action button

- **Bottom Navigation**

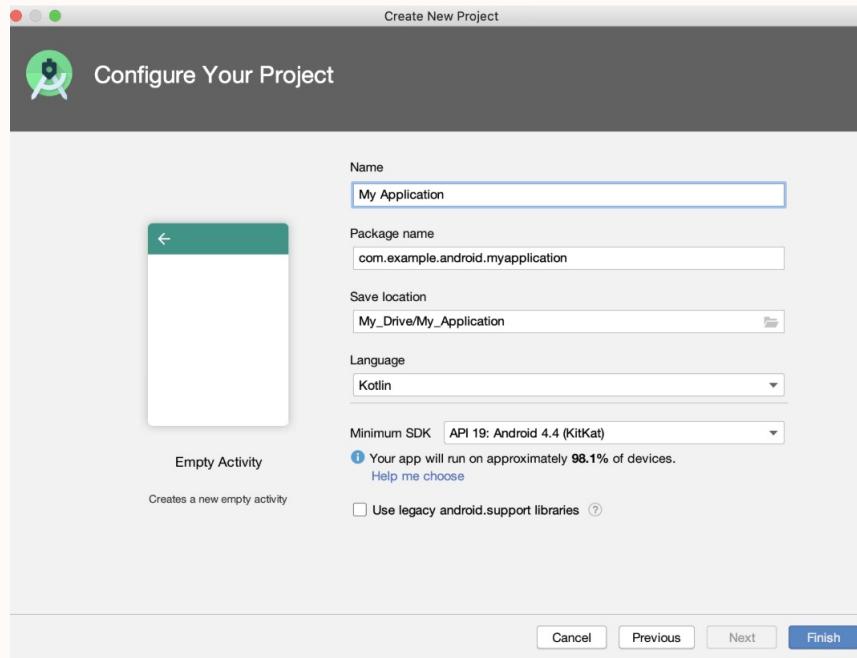
Template sa bottom navigation bar-om

- **Navigation Drawer**

Template sa navigation drawer-om

Za prvu aplikaciju, preporučujemo **Empty Activity** template jer omogućava da razumijete osnovnu strukturu bez dodatne kompleksnosti.

Unijeti detalje o projektu



Nakon izbora template-a, potrebno je unijeti osnovne informacije o vašem projektu. Ova konfiguracija definije identitet i osnovne parametre vaše aplikacije.

1

Name

Ime vaše aplikacije koje će biti vidljivo korisnicima (npr. "Moja Prva App")

2

Package name

Jedinstveni identifikator aplikacije, obično u reverse-domain formatu (npr. com.mojacompanija.mojaapp)

3

Save location

Direktorij na vašem računaru gdje će biti sačuvan projekat

4

Language

Programski jezik - Kotlin (preporučeno) ili Java

5

Minimum SDK

Najniži API level koji vaša aplikacija podržava (više o ovome na sljedećim slajdovima)

Android 8.1	27	O_MR1	Platform Highlights
Android 8.0	26	O	Platform Highlights
Android 7.1.1	25	N_MR1	Platform Highlights
Android 7.1			
Android 7.0	24	N	Platform Highlights

Android Releases & API Levels

Android platforma se razvija kroz verzije, a svaka verzija ima asocirani API level. Razumijevanje ovog sistema je ključno za odlučivanje koje uređaje vaša aplikacija može podržavati.

Historijski razvoj

- **Android 4.4 KitKat (API 19):** Optimizacije performansi
- **Android 5.0 Lollipop (API 21):** Material Design
- **Android 6.0 Marshmallow (API 23):** Runtime permissions
- **Android 7.0 Nougat (API 24):** Multi-window support
- **Android 8.0 Oreo (API 26):** Notification channels

Novije verzije

- **Android 9 Pie (API 28):** Gesture navigation
- **Android 10 (API 29):** Dark theme
- **Android 11 (API 30):** Scoped storage
- **Android 12 (API 31):** Material You
- **Android 13+ (API 33+):** Nastavak evolucije

Svaka nova verzija donosi nove funkcionalnosti, ali podržavanje starijih verzija znači širu dostupnost vaše aplikacije.

Izbor API Level za vašu aplikaciju

Postoje tri ključna API level-a koja trebate razumjeti prilikom konfiguracije vašeg Android projekta. Svaki ima specifičnu ulogu i implikacije.

Minimum SDK

Najniža verzija Android OS-a na kojoj će vaša aplikacija raditi. Uredaj mora imati najmanje ovaj API level da bi instalirao aplikaciju

Target SDK

API verzija sa kojom je aplikacija dizajnirana i testirana. Označava najvišu Android verziju sa kojom ste testirali aplikaciju

Compile SDK

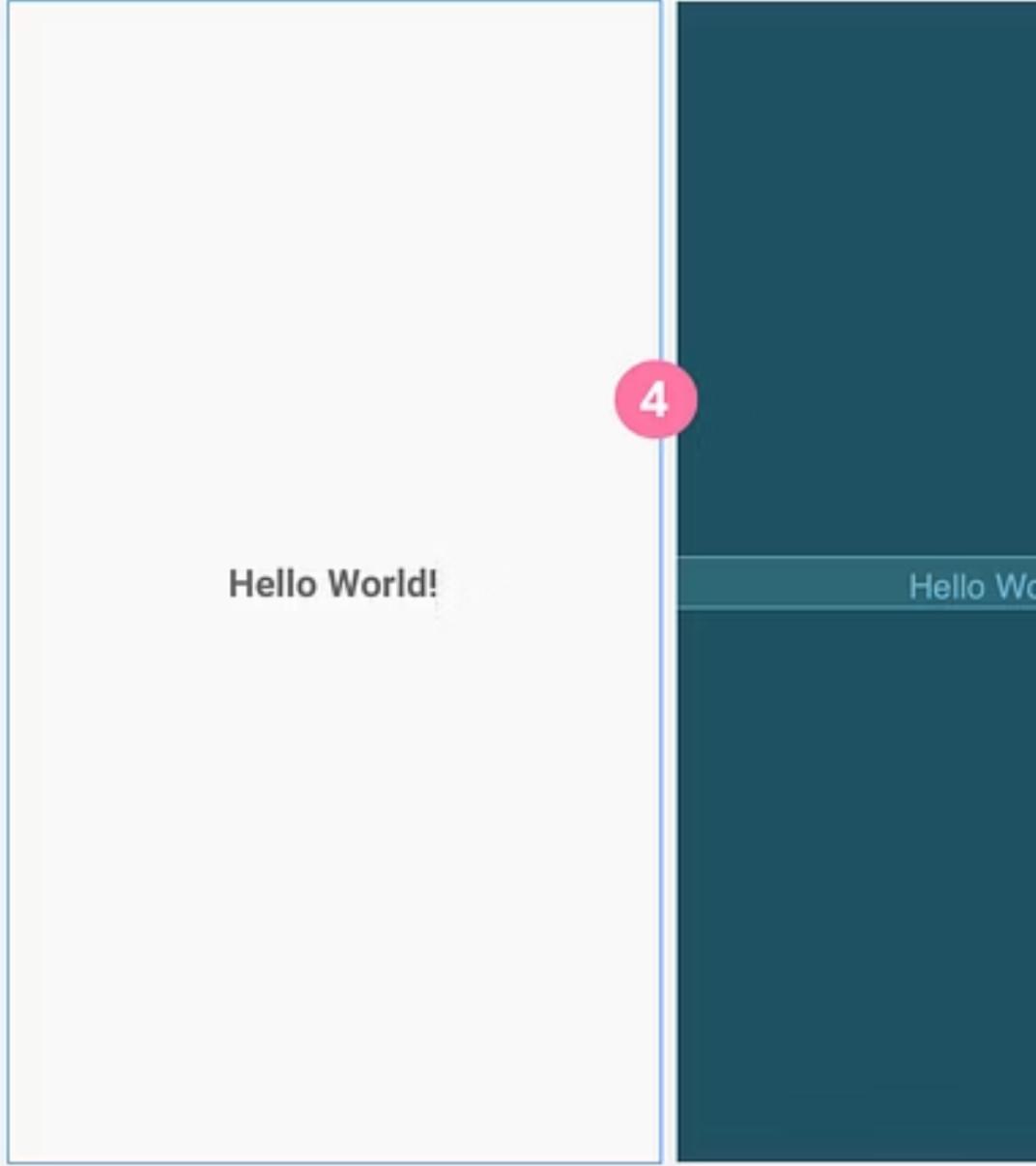
Verzija Android OS biblioteke sa kojom se aplikacija kompajlira. Omogućava korištenje najnovijih API funkcija tokom razvoja

Pravilo konfiguracije

```
minSdkVersion ≤ targetSdkVersion ≤ compileSdkVersion
```

API level identificuje verziju framework API-ja Android SDK-a i definiše koje funkcionalnosti su dostupne na određenom nivou platforme.

- Preporuka:** Postavite minSdkVersion na API 21 (Android 5.0) ili viši za pristup Material Design komponentama i modernim funkcionalnostima, dok još uvijek podržavate preko 95% aktivnih Android uređaja.



Android Studio

Nakon što kreirate projekat, Android Studio otvara radno okruženje sa nekoliko ključnih sekcija koje će vam pomoći u razvoju aplikacije.

Project Explorer

Lijevi panel prikazuje strukturu projekta - sve datoteke, resource-e, i kod vaše aplikacije organizovani u folder-e

Editor Window

Centralni prostor za pisanje koda, uređivanje XML layout-a, i pregled drugih datoteka projekta

Build & Run Toolbar

Gornja traka sa dugmadima za kompajliranje, pokretanje aplikacije, debugging, i pristup Android SDK Manager-u

Bottom Panel

Prikazuje logcat (system logs), terminal, build output, i druge korisne informacije tokom razvoja

Android Studio je zasnovan na IntelliJ IDEA platformi i pruža napredne funkcije kao što su intelligent code completion, refactoring alati, i integrirani version control.

Pokretanje aplikacije

Kada završite sa osnovnom konfiguracijom i imate jednostavan kod, spremni ste da pokrenete vašu prvu Android aplikaciju. Postoje dva načina za testiranje aplikacije.

Fizički uređaj

Povežite Android telefon ili tablet putem USB kabla. Omogućite USB debugging u Developer Options na uređaju. Ovo je najbolji način za testiranje stvarnog korisničkog iskustva.

1

Kliknite Run

Zeleno dugme "play" u toolbar-u

2

Izaberite uređaj

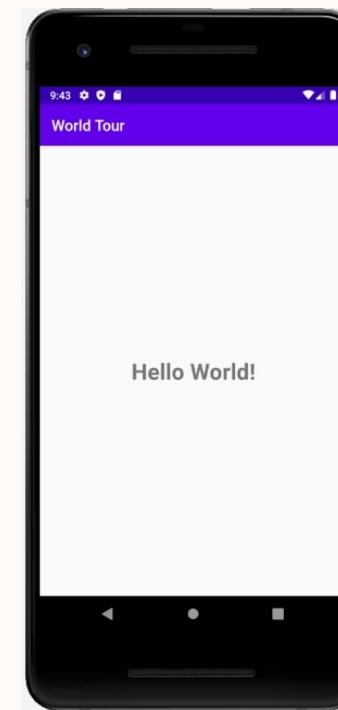
Fizički ili emulator

3

Aplikacija se pokreće

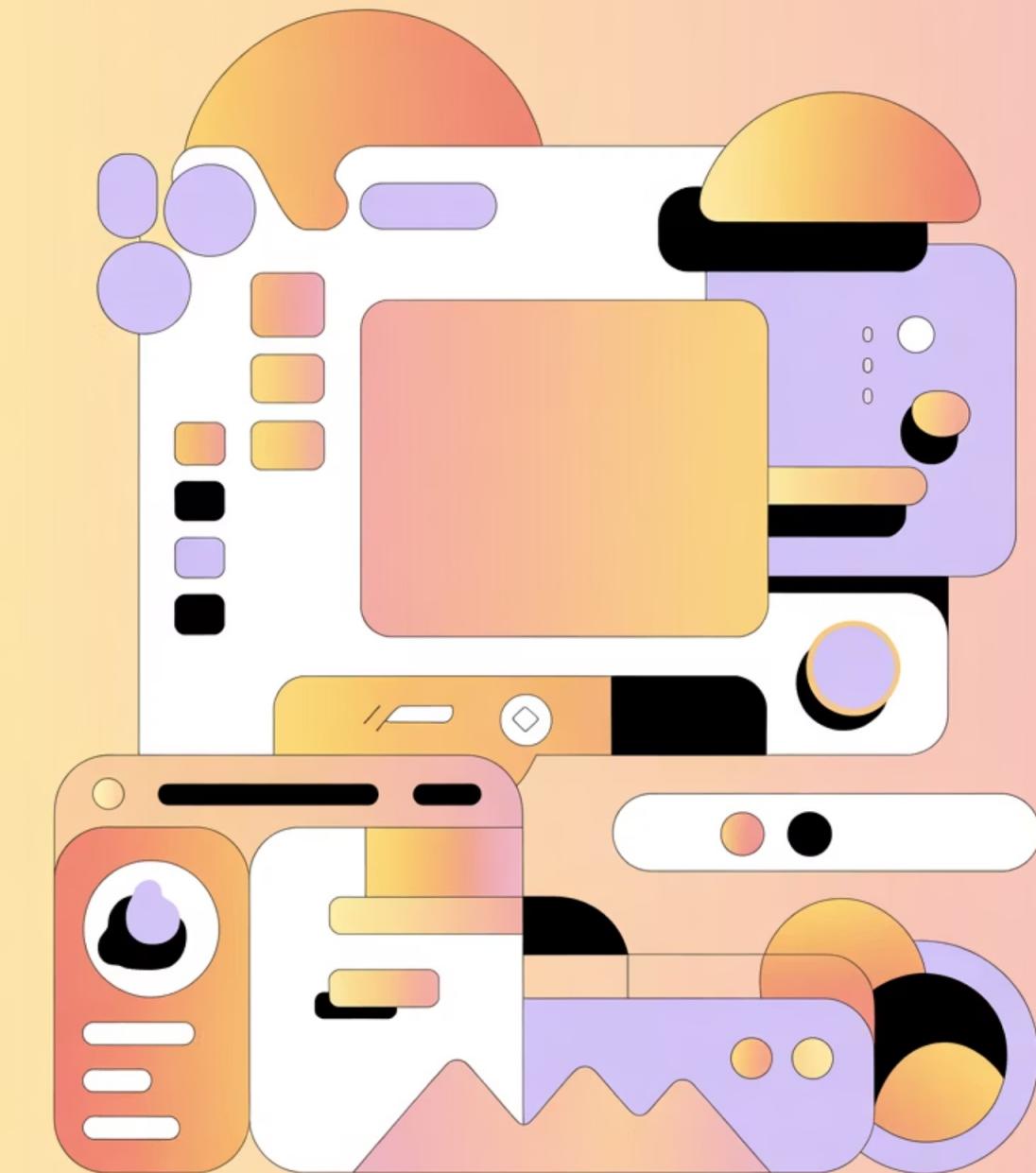
Vidjet ćete vašu aplikaciju

Čestitamo! Upravo ste kreirali i pokrenuli vašu prvu Android aplikaciju. Ovo je samo početak vašeg putovanja u Android razvoju.



Gradle

Alat za automatizaciju izgradnje Android aplikacija koji omogućava efikasno upravljanje projektom i zavisnostima.



Šta je Gradle?

Gradle je moderan sistem za automatizaciju izgradnje aplikacija koji revolucionira način na koji Android developeri upravljaju svojim projektima. Ovaj moćan alat omogućava kompletну kontrolu nad cijelim životnim ciklusom razvoja aplikacije.



Automatizacija izgradnje

Upravlja kompleksnim procesom izgradnje kroz seriju dobro definisanih zadataka koji se izvršavaju automatski.



Upravljanje zadacima

Kompajlira Kotlin izvorne datoteke, pokreće unit i integration testove, te instalira aplikaciju na fizički ili virtuelni uređaj.



Optimalan redoslijed

Inteligentno određuje pravilan redoslijed izvršavanja zadataka kako bi osigurao efikasan i ispravan build proces.

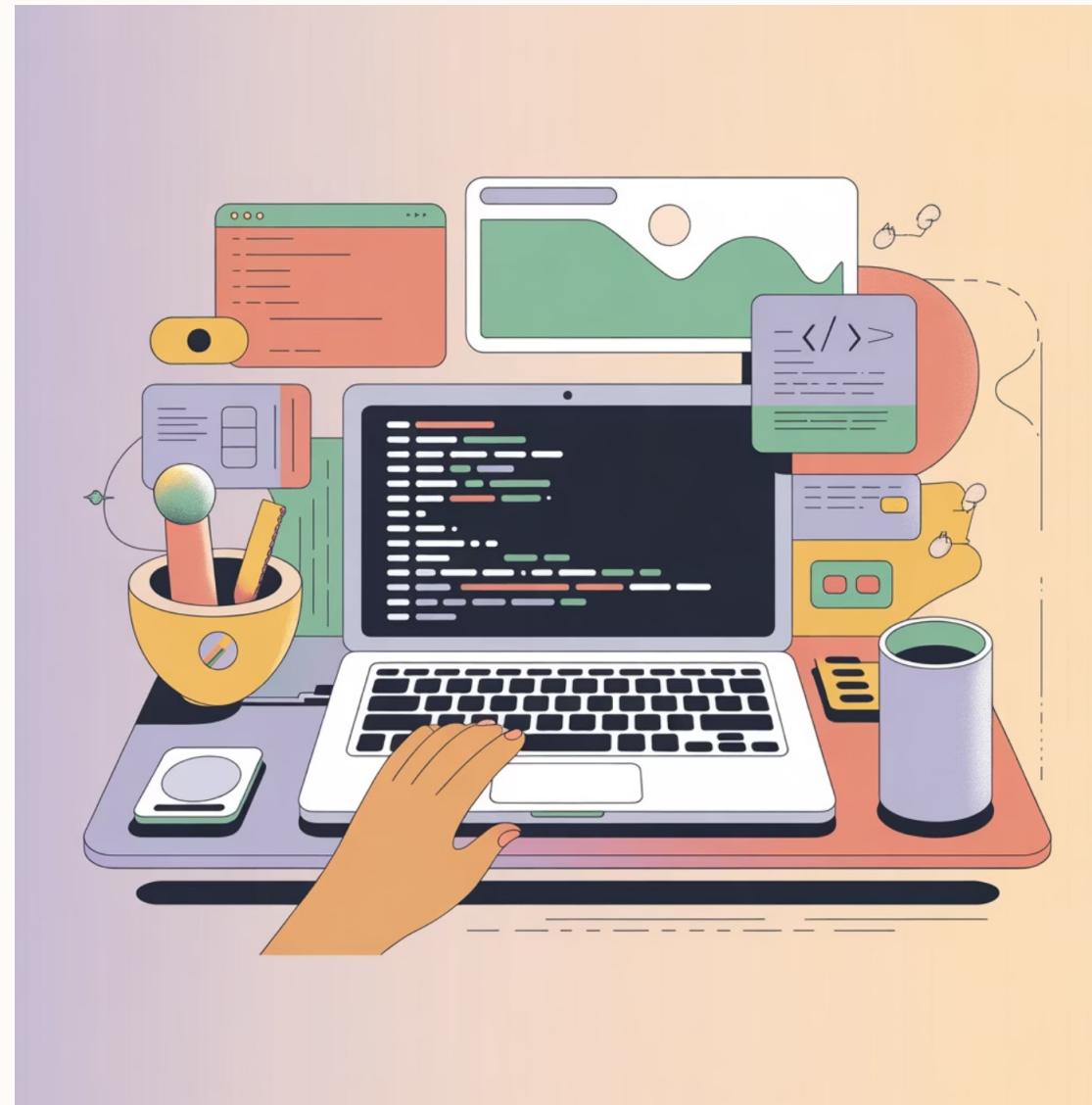


Upravljanje zavisnostima

Elegantno rješava ovisnosti između različitih projekata, modula i eksternih biblioteka trećih strana.

Uobičajeni Gradle zadaci

Tokom svakodnevnog razvoja, koristit ćete nekoliko ključnih Gradle zadataka. Razumijevanje ovih osnovnih komandi je esencijalno za efikasan rad.



Clean

Briše sve prethodno generirane build datoteke i artefakte iz projekta, omogućavajući potpuno čist build od početka. Koristan kada trebate rješiti probleme sa cache-om.



Tasks

Prikazuje kompletnu listu svih dostupnih Gradle zadataka u vašem projektu. Ovo je izuzetno korisno za istraživanje mogućnosti i razumijevanje strukture build sistema.

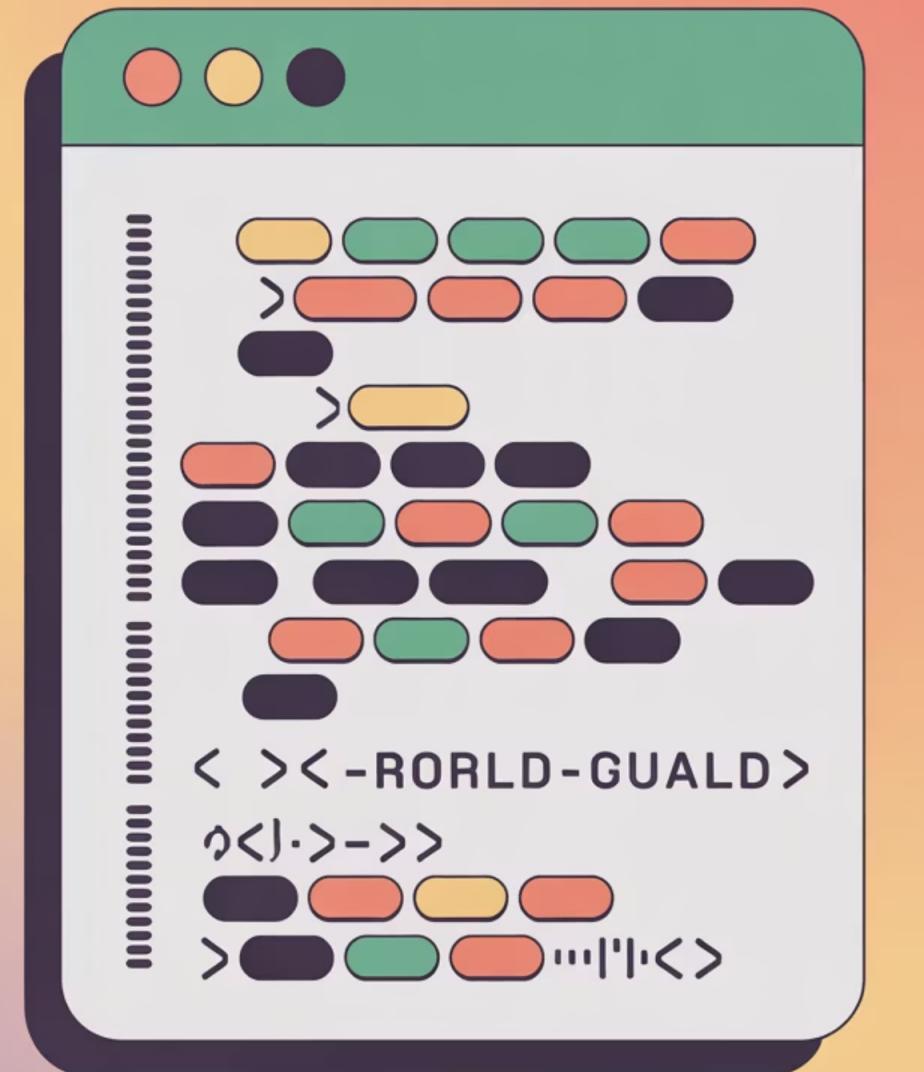


InstallDebug

Kompajlira debug verziju aplikacije i automatski je instalira na povezani Android uređaj ili emulator, pripremajući je za testiranje i razvoj.

Gradle build file

Build datoteka je centralno mjesto gdje se definije cijela konfiguracija vašeg Android projekta. Ova datoteka, obično nazvana `build.gradle`, sadrži sve bitne informacije o tome kako se vaša aplikacija gradi.



Deklaracija plugin-a

Definiše koje plugin-e aplikacija koristi za kompajliranje i packaging. Ovo uključuje Android plugin, Kotlin plugin i druge specifične alate.

Android properties

Postavlja osnovne Android parametre kao što su SDK verzije, build tools verzije, i aplikacijske identifikatore potrebne za kompajliranje.

Upravljanje zavisnostima

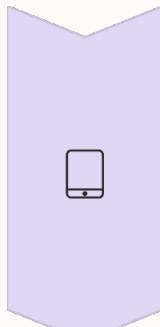
Specificira sve eksterne biblioteke i framework-e koje aplikacija koristi, omogućavajući automatsko preuzimanje i integraciju.

Povezivanje sa repository-jima

Definiše odakle Gradle preuzima zavisnosti - Google Maven, JCenter, ili custom Maven repository-je koje koristi vaša organizacija.

Plugins

Plugin-i su ključni elementi koji proširuju funkcionalnost Gradle-a i omogućavaju razvoj Android aplikacija. Oni obezbeđuju potrebnu infrastrukturu, alate i biblioteke za kompajliranje, testiranje i pakovanje vaše aplikacije.



Android Application Plugin



```
apply plugin: 'com.android.application'
```

Osnovni plugin koji omogućava kreiranje Android aplikacija. Bez njega ne možete kompajlirati APK ili AAB datoteke.



Kotlin Android Plugin



```
apply plugin: 'kotlin-android'
```

Dodaje podršku za Kotlin programski jezik, omogućavajući kompajliranje Kotlin izvornog koda u bytecode.



Kotlin Extensions Plugin



```
apply plugin: 'kotlin-android-extensions'
```

Omogućava korištenje Kotlin sintaksnih ekstenzija i olakšava pristup UI elementima bez eksplicitnog `findViewById` poziva.

- Napomena:** `kotlin-android-extensions` plugin je označen kao deprecated u novijim verzijama. Umjesto njega, preporučuje se korištenje View Binding ili Data Binding pristupa.

Android konfiguracija

Android blok u build.gradle datoteci definiše sve specifične postavke vezane za Android platformu. Ovdje se postavljaju verzije SDK-a, build alata, i osnovne informacije o aplikaciji.

```
android {  
    namespace = "com.example.hellocompose"  
    compileSdk = 36  
  
    defaultConfig {  
        applicationId = "com.example.hellocompose"  
        minSdk = 24  
        targetSdk = 36  
        versionCode = 1  
        versionName = "1.0"  
  
        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"  
    }  
}
```

compileSdkVersion

Verzija Android SDK-a koja će se koristiti za kompajliranje aplikacije. Određuje koje API-je možete koristiti u kodu.

buildToolsVersion

Verzija Android build alata koji se koriste za kompajliranje, packaging i signing aplikacije.

applicationId

Jedinstveni identifikator vaše aplikacije na Google Play Store-u i Android sistemu.

minSdkVersion

Najniža verzija Android OS-a koju aplikacija podržava. API level 19 odgovara Android 4.4 KitKat.

targetSdkVersion

Verzija Android-a za koju je aplikacija optimizovana i testirana. Utiče na ponašanje kompatibilnosti.

Dependencies

Sekcija dependencies je mjesto gdje deklarirate sve eksterne biblioteke koje vaša aplikacija koristi. Gradle automatski preuzima ove zavisnosti sa specificiranih repository-ja i integriše ih u vaš projekat.

```
dependencies {  
    implementation(libs.androidx.core.ktx)  
    implementation(libs.androidx.lifecycle.runtime.ktx)  
    implementation(libs.androidx.activity.compose)  
    implementation(platform(dependencyProvider = libs.androidx.compose.bom))  
    implementation(libs.androidx.compose.ui)  
    implementation(libs.androidx.compose.ui.graphics)  
    implementation(libs.androidx.compose.ui.tooling.preview)  
    implementation(libs.androidx.compose.material3)  
    testImplementation(libs.junit)  
    androidTestImplementation(libs.androidx.junit)  
    androidTestImplementation(libs.androidx.espresso.core)  
    androidTestImplementation(platform(dependencyProvider = libs.androidx.compose.bom))  
    androidTestImplementation(libs.androidx.compose.ui.test.junit4)  
    debugImplementation(libs.androidx.compose.ui.tooling)  
    debugImplementation(libs.androidx.compose.ui.test.manifest)  
}
```

implementation

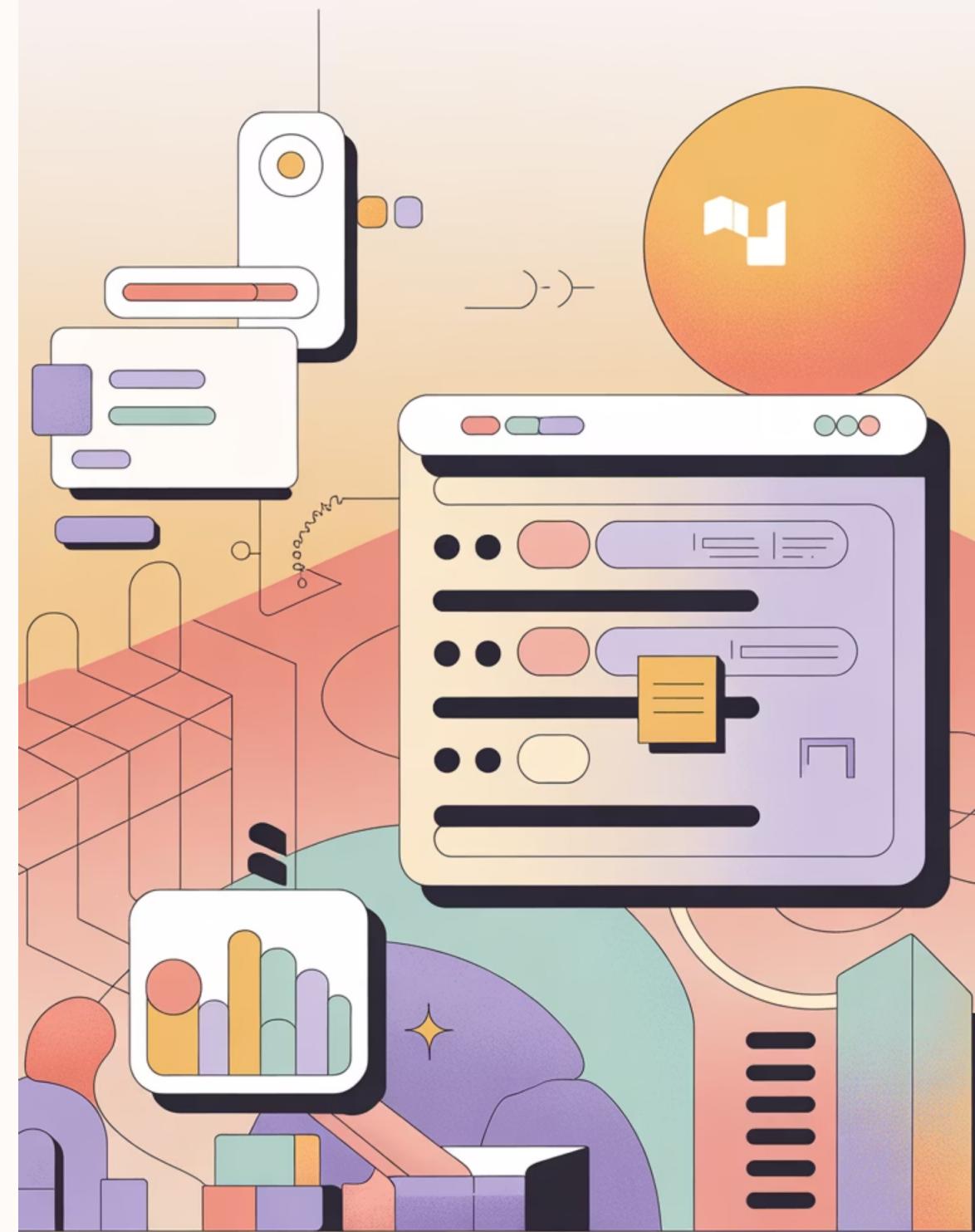
Standardna konfiguracija za zavisnosti koje su potrebne u runtime-u. Biblioteke deklarirane ovako nisu vidljive drugim modulima koji zavise od vašeg.

testImplementation

Zavisnosti potrebne samo za unit testove. Ove biblioteke se koriste tokom izvršavanja lokalnih testova na JVM-u.

androidTestImplementation

Zavisnosti za instrumentation testove koji se izvršavaju na Android uređaju ili emulatoru.



Repositories

Repositories definiše odakle Gradle preuzima zavisnosti vaše aplikacije. Ovo su serveri koji hostuju biblioteke i komponente koje koristite u svom projektu.

```
pluginManagement {  
    repositories {  
        google {  
            content {  
                includeGroupByRegex( groupRegex = "com\\\\.android.*")  
                includeGroupByRegex( groupRegex = "com\\\\.google.*")  
                includeGroupByRegex( groupRegex = "androidx.*")  
            }  
        }  
        mavenCentral()  
        gradlePluginPortal()  
    }  
}
```



Google Maven Repository

Službeni Google repository koji sadrži Android i Google biblioteke poput AndroidX, Material Design komponenti i Google Play services.

- **Best practice:** Uvijek navedite google() repository prije ostalih jer Android biblioteke trebaju prioritet. Redoslijed repository-ja utiče na brzinu preuzimanja zavisnosti.

Zaključak

