### Návrh a vývoj mobilních aplikací

#### Vývojové platformy

Operační systémy iOS a Android jsou dvě hlavní platformy pro vývoj mobilních aplikací, které mají svá vlastní vývojová prostředí, nástroje a ekosystémy. Existují také multiplatformové vývojové nástroje, které umožňují vytvářet aplikace pro více platforem současně.

**iOS** - **Vývojové prostředí**: Hlavním vývojovým prostředím pro iOS je Xcode, které poskytuje nástroje pro návrh uživatelského rozhraní, úpravu kódu, ladění a testování. - **Programovací jazyky**: Swift a Objective-C. - **Historie**: iOS byl poprvé představen v roce 2007 s prvním iPhonem. - **Současnost**: iOS se vyvinul v robustní ekosystém s pravidelnými aktualizacemi a podporou široké škály zařízení Apple včetně iPhonu, iPadu a Apple Watch.

**Android OS** - **Vývojové prostředí**: Hlavním vývojovým prostředím pro Android je Android Studio, které nabízí integrované nástroje pro návrh, kódování, ladění a testování. - **Programovací jazyky**: Java a Kotlin. - **Historie**: Android byl poprvé vydán v roce 2008 jako otevřená platforma pro mobilní zařízení. - **Současnost**: Android je nejrozšířenější mobilní operační systém s podporou široké škály zařízení od různých výrobců.

**Cross-platform** - **Vývojové nástroje**: React Native, Flutter, Xamarin, Unity. - **Výhody**: Možnost napsat kód jednou a nasadit ho na více platformách (iOS, Android). - **Nevýhody**: Může být náročnější na výkon a přizpůsobení nativních funkcí.

#### Vývojové prostředí

**iOS – Xcode** - IDE vyvinuté společností Apple pro vývoj aplikací pro iOS. Nabízí nástroj Interface Builder, který umožňuje grafický návrh uživatelského rozhraní, a integrované nástroje pro testování a ladění.

**Android – Android Studio** - Oficiální IDE pro vývoj aplikací pro Android. Poskytuje nástroje pro návrh uživatelského rozhraní, ladění a správu projektů.

#### Historie a současnost

**iOS** - **Historie**: Systém iOS 1.0 byl představen v roce 2007 s prvním iPhonem. Od té doby prošel mnoha změnami a aktualizacemi včetně přidání obchodu App Store, multitaskingu a podpory dalších zařízení jako je iPad a Apple Watch. - **Současnost**: Nejnovější verze iOS obsahuje pokročilé funkce jako ARKit pro rozšířenou realitu, nové bezpečnostní funkce a vyšší výkon.

**Android** - **Historie**: Systém Android 1.0 byl představen v roce 2008. S každou novou verzí přibývaly nové funkce jako podpora více obrazovek, vylepšené uživatelské rozhraní a pokročilé bezpečnostní funkce. - **Současnost**: Systém Android nabízí širokou podporu pro různá zařízení a formáty včetně smartphonů, tabletů, televizorů a automobilů.

#### Základy programování v Javě

Java je hlavní programovací jazyk používaný pro vývoj aplikací pro Android.

* **Syntaxe**: Java je objektově orientovaný jazyk s jasnou a jednoduchou syntaxí.
* **Třídy a objekty**: Základní stavební prvky jazyka Java. Třída je šablona pro objekty, které obsahují metody a atributy. Objekt je instance třídy.
* **Dědičnost**: Mechanismus, který umožňuje jedné třídě dědit atributy a metody jiné třídy.
* **Polymorfismus**: Schopnost objektů různých tříd zpracovávat stejné rozhraní.
* **Výjimky**: Mechanismus pro zpracování chyb během vykonávání programu.

#### Úvod do vývojového prostředí Android Studio

**LogCat**: Nástroj pro zobrazování protokolů spuštěných aplikací. Pomáhá při diagnostice problémů a ladění. Zobrazuje výstupy z různých úrovní protokolování jako ladění, chyby, informace atd.

**ADB (Android Debug Bridge)**: Nástroj pro komunikaci se zařízeními s Androidem. Umožňuje instalovat aplikace, spouštět příkazy a ladit aplikace.

**Android SDK**: Android SDK (Software Development Kit) obsahuje všechny nástroje potřebné pro vývoj aplikací pro Android včetně knihoven, rozhraní API a nástrojů pro ladění. Obsahuje také emulátory pro testování aplikací na různých verzích systému Android.

#### Senzory (WIFI, BT, GPS, GSM, kompas, gyroskop, akcelerometr)

Zařízení s Androidem často obsahují různé snímače, které umožňují vytvářet bohaté a interaktivní aplikace:

* **WIFI**: Umožňuje připojení k bezdrátovým sítím. API pro správu připojení, skenování dostupných sítí atd.
* **Bluetooth**: Umožňuje bezdrátovou komunikaci s jinými zařízeními. Podpora různých profilů jako klasického Bluetooth a Bluetooth Low Energy (BLE).
* **GPS**: Umožňuje určování polohy pomocí satelitů. API pro získání aktuální polohy, sledování polohy atd.
* **GSM**: Umožňuje komunikaci prostřednictvím mobilních sítí. API pro správu telefonních hovorů, posílání SMS atd.
* **Kompas**: Umožňuje určení směru. Na určení směru vzhledem k magnetickému poli Země používá magnetometr.
* **Gyroskop**: Měří orientaci a úhlovou rychlost. Používá se pro zjišťování otáčení a naklonění zařízení.
* **Akcelerometr**: Měří zrychlení a dokáže rozpoznat pohyb a otřesy. Používá se pro detekci pádů, změn orientace zařízení atd.

#### Android – struktura projektu

Struktura projektu Android zahrnuje několik klíčových komponent:

* **Manifest**: Soubor AndroidManifest.xml obsahuje důležité informace o aplikaci, například oprávnění, aktivity a služby.
* **Kód**: Obsahuje zdrojový kód aplikace, obvykle v průčincích java nebo kotlin. Hlavní soubory jsou třídy, které definují chování aplikace.
* **Zdroje (resources)**: Obsahuje soubory jako obrázky, rozložení, řetězce atd. Zdroje jsou uspořádány v průčincích jako například res/drawable, res/layout, res/values atd.
* **Gradle build toolkit**: Nástroj pro automatizaci procesu sestavování, správu závislostí a konfiguraci projektu.

#### Tvorba UI

Vytvoření uživatelského rozhraní (UI) v systému Android zahrnuje:

* **XML**: Slouží k definování rozložení a vzhledu aplikace.
* **LinearLayout**: Uspořádá prvky do řádku nebo sloupce. Používá se pro jednoduché a pravidelné uspořádání prvků.
* **RelativeLayout**: Umožňuje relativní umístění prvků vůči sobě. Používá se při složitějším uspořádání prvků.
* **Styly**: Definují vzhled a styl prvků aplikace. Styly lze definovat v souboru res/values/styles.xml.
* **Activity**: Základní stavební prvek aplikace, kterým je jedna obrazovka s uživatelským rozhraním. Každá aktivita má svůj vlastní životní cyklus a může obsahovat více fragmentů.
* **Fragment**: Modulární část uživatelského rozhraní, kterou lze použít v rámci aktivity. Fragmenty umožňují flexibilní a opakovaně použitelné komponenty uživatelského rozhraní.
* **Změny konfigurace**: Odstraňování problémů souvisejících se změnami konfigurace, jako je například otáčení obrazovky. Možnosti zahrnují použití ViewModel na zachování údajů během změn konfigurace.
* **Optimalizace pro různá zařízení**: Použití různých rozložení, zdrojů obrázků a velikostí text

u pro různé velikosti a rozlišení obrazovky. Responzivní design pomocí ConstraintLayout a dalších nástrojů.

#### Životní cyklus aplikace

**Activity Lifecycle**: Systém Android řídí životní cyklus aktivity prostřednictvím série stavů. Každý stav představuje určitou fázi života aktivity:

* **onCreate()**: Inicializace aktivity. Zde se obvykle nastavuje uživatelské rozhraní a inicializují se potřebné zdroje.
* **onStart()**: Aktivita se stává viditelnou pro uživatele, ale ještě není v popředí.
* **onResume()**: Aktivita se dostane do popředí a uživatel může interagovat.
* **onPause()**: Aktivita ztrácí fokus, ale je stále viditelná. Zde se obvykle ukládají změny, které by měly být trvalé.
* **onStop()**: Aktivita není viditelná. Systém může uvolnit zdroje.
* **onDestroy()**: Aktivita je zničena. Zde se uvolní všechny zbývající zdroje.

#### Komunikace mezi aktivitami

* **Intent (Záměr)**: Mechanismus pro zahájení nových činností nebo komunikaci mezi činnostmi. Záměry mohou být explicitní (zaměřené na konkrétní činnost) nebo implicitní (se všeobecným záměrem, který může být naplněn více aplikacemi).
* **BroadcastReceiver**: Komponent pro příjem a zpracování globálního vysílání. Vysílání mohou být systémová (např. změna stavu baterie) nebo definovaná aplikací.

#### Interakce s uživateli

* **Notifikace**: Informují uživatele o událostech mimo aplikaci. Zobrazují se v oznamovací liště. Systém Android poskytuje rozhraní API oznámení na vytváření a správu oznámení.
* **Toast zprávy**: Krátké zprávy, které se uživateli zobrazí na krátký čas. Neprerušují aktuální činnost uživatele.
* **SnackBar**: Rozšířená verze toastových zpráv s možností akce. Zobrazuje se ve spodní části obrazovky.
* **Dialogy**: Okna, která vyžadují interakci uživatele, jako například AlertDialog pro potvrzování akcí.

#### Trvalá data v aplikaci

* **SharedPreferences**: Ukládání jednoduchých dvojic klíč-hodnota údajů. Slouží pro ukládání nastavení a jednoduchých údajů.
* **Databáze SQLite**: Relační systém správy databází pro komplexnější ukládání údajů.
* **Správa souborů**: Ukládání a čtení údajů z interní nebo externí paměti zařízení.

#### Práce na pozadí

* **Threads**: Nízkoúrovňový mechanismus pro spouštění kódu na pozadí.
* **AsyncTask**: Jednodušší API pro spouštění úloh na pozadí a aktualizaci uživatelského rozhraní.
* **Services**: Komponenty pro dlouhodobé úlohy na pozadí, které nemají přímou interakci s uživatelem.

#### Android Design Guidelines

**Dokumentace**: Oficiální dokumentace k systému Android obsahuje pokyny pro navrhování aplikací podle zásad Material Designu. Navrženo tak, aby poskytovalo konzistentní uživatelské prostředí napříč zařízeními a platformami.

**Optimalizace**: Osvedčené postupy pro efektivní kódování, výkon a uživatelskou zkušenost. Efektivní využívání paměti a procesoru. Minimalizace spotřeby baterie. Optimalizace uživatelského rozhraní pro různé velikosti obrazovky.

**Používání knihoven**: Populární knihovny jako Retrofit (pro HTTP požadavky), Glide (pro práci s obrázky) a Room (pro práci s databázemi). - **Retrofit**: Knihovna pro jednoduchou komunikaci s rozhraním REST API. - **Glide**: Knihovna pro efektivní vyhledávání a zpracování obrázků. - **Room**: Abstrakce nad databází SQLite pro jednodušší správu databázových operací.

#### Komunikace se senzory (WIFI, BT, GPS, GSM, kompas, gyroskop, akcelerometr)

Systém Android poskytuje rozhraní API pro komunikaci se snímači, která aplikacím umožňují přístup k údajům z různých snímačů:

* **WIFI**: API pro správu připojení k bezdrátovým sítím, skenování dostupných sítí a správu připojení.
* **Bluetooth (BT)**: API pro komunikaci s jinými zařízeními prostřednictvím Bluetooth včetně klasického Bluetooth a Bluetooth Low Energy (BLE).
* **GPS**: API pro získávání aktuální polohy zařízení pomocí satelitů, sledování polohy a další funkce související s polohou.
* **GSM**: API pro správu mobilních sítí, telefonních hovorů, SMS a dalších funkcí souvisejících s mobilní komunikací.
* **Kompas**: Rozhraní API pro přístup k magnetometru zařízení s cílem určit jeho směr vzhledem k magnetickému poli Země.
* **Gyroskop**: API pro přístup k gyroskopu zařízení, který měří orientaci a úhlovou rychlost.
* **Akcelerometr**: API pro přístup k akcelerometru zařízení, který měří zrychlení a dokáže zjišťovat pohyb a otřesy.

#### Sdílení údajů mezi aplikacemi/servermi

* **Content providers**: Umožňují aplikacím sdílet údaje s jinými aplikacemi. Poskytují standardizované rozhraní pro přístup k údajům jako jsou kontakty, média a další.
* **SyncAdapter**: Mechanismus pro synchronizaci údajů mezi zařízeními a servery. Umožňuje automatizovanou a efektivní synchronizaci údajů.
* **REST API**: Rozhraní pro komunikaci mezi klientem a serverem pomocí požadavků HTTP. Používá se pro přenos údajů mezi aplikacemi a servery.

#### Podpisování aplikací a publikování v službě Google Play

* **Podpisování aplikace**: Proces zabezpečení pravosti a integrity aplikace před jejím zveřejněním. Podpísaná aplikace obsahuje digitální podpis, který ověřuje její původ a zaručuje, že s ní nebylo manipulováno.
* **Publikování v službě Google Play**: Proces nahrávání a distribuce aplikací prostřednictvím obchodu Google Play. Vývojáři si musí vytvořit účet Google Play Developer a nahrát soubor APK spolu s popisem aplikace, snímky obrazovky a dalšími informacemi.
* **Bezplatné vs. platené aplikace**: Rozhodnutí o strategii monetizace aplikace. Bezplatné aplikace mohou generovat příjmy prostřednictvím reklam nebo nákupů v aplikacích, zatímco platené aplikace vyžadují jednorázový nákup.
* **Reklamy**: Integrace reklamy na generování příjmů. Mezi oblíbené reklamní platformy patří Google AdMob, Facebook Audience Network a další.