Android - Introduzione

Programmazione Mobile

A.A. 2021/22

M.O. Spata

Per Iniziare

- Per qualunque tipo di sviluppatore (desktop, enterprise, web o mobile) Android rappresenta una opportunità nuova per la progettazione e l'implementazione di applicazioni innovative destinate a dispositivi mobili.
- Android è uno stack software open source, che include un sistema operativo, uno strato middleware, applicazioni native, un set di librerie e di API per la scrittura di applicazioni destinate a dispositivi mobili.
- I moderni “smartphone” sono sempre più con uno stile accattivante ma soprattutto potenti: sono dotati di buone capacità di calcolo e di svariate funzionalità (video-camera, media-palyer, GPS, touch-screen, accelerometro) grazie alle diverse periferiche di cui sono dotati.

Piattaforme per dispositivi mobili

- Le piattaforme di sviluppo e le applicazioni software, "purtroppo" non hanno saputo tenere il passo dell'evoluzione della tecnologia e dei dispositivi mobili.
- Le tecnologie e le piattaforme dei dispositivi mobili rappresentano di fatto ambienti chiusi costruiti su sistemi operativi proprietari con strumenti di sviluppo proprietari
- Gli stessi dispositivi "privilegiano" le applicazioni native rispetto a quelle scritte dagli utenti, creando di fatto una barriera verso la speranza degli sviluppatori di creare applicazioni che sfruttino pienamente le potenzialità di hardware sempre più potente.

Piattaforma android

- In Android, le applicazioni native e quelle cosiddette "terze-parti" ovvero scritte da sviluppatori, sono basate sullo stesso set di API ed eseguite nello stesso ambiente di run-time.
- Tali API consentono l'accesso alle funzionalità del dispositivo e quindi di fatto accesso all'hardware, ai servizi location-based, alle attività map-based, etc....

Cosa vedremo durante questo corso

- Questo corso aiuterà a comprendere come usare le API suddette per creare una applicazione Android.
- Nel seguito di questa lezione saranno introdotte alcune linee guida per lo sviluppo di app per dispositivi mobili e saranno presentate alcune funzionalità e caratteristiche del framework di sviluppo di Android

Un po’ di storia

- Gli sviluppatori, per sistemi embedded e/o dispositivi mobili, sono stati "costretti" a studiare e comprendere il funzionamento dell'hardware dei dispositivi target e scrivere codice C o C++ per pilotarli e/o gestirli. Questo sforzo doveva essere fatto per ogni dispositivo o famiglia di dispositivi.
- Questo approccio non può essere considerato valido per restare al passo con l'evoluzione tecnologica.

Un po’ di storia

- Piattaforme più "recenti" quali Symbian, ad esempio, sono state create per fornire agli sviluppatori la possibilità di distribuire le stesse applicazioni a più dispositivi dalla caratteristiche "simili". Questo ha permesso di fare avvicinare sempre più sviluppatori al mercato mobile.
- Queste piattaforme forniscono qualche accesso alle periferiche del dispositivo ma richiedono la scrittura di complicate e complesse funzioni C o C++ basate pesantemente per altro su API proprietarie spesso di difficile comprensione.

Un po’ di storia

- La difficoltà di sviluppo si amplifica se si vuole sviluppare codice per dispositivi differenti che fanno uso di periferiche e funzionalità particolare (es. GPS).
- Grossi passi in avanti sono stati fatti con l'introduzione e l'uso di MIDlets. Queste ultime, classi java sviluppate per il mobile, vengono eseguite su una Java Virtual Machine e permettono di astrarre il livello hardware, consentendo allo sviluppatore di progettare e sviluppare software per diversi dispositivi che supportano Java RunTime.
- Prezzo da pagare: l'utilizzo di MIDlets, limita l'accesso alle risorse hardware. Le MIDlets sviluppate sembrano sempre più applicazioni desktop ridisegnate per funzionare su uno schermo più piccolo.

Il “Futuro”

- Windows Mobile ed Apple IPhone, forniscono oggi ambienti di sviluppo per applicazioni mobili molto semplici ed efficienti.
- Purtroppo, però, essi sono stati costruiti su sistemi operativi proprietari e privilegiano applicazioni native a scapito di quelle create da terze parti, e comunque pongono delle severe restrizioni sulla possibilità di comunicazione tra applicazioni e sull'accesso ai dati nativi relativi al "telefono".

Era del mobile

- Evoluzione dei dispositivi mobili:
  - furono telefoni.. Ora "a volte" sono anche telefoni;
  - da dispositivi "sfortunati" (schermi microscopici a cristalli liquidi, tastiera minimale)..
  - ..a dispositivi «smart» e «phone», con schermo (multi)touch, rete, GPS, …
  - da sistemi embedded custom scritti dalle aziende produttrici di dispositivi mobili…
  - ..ad aziende che si preoccupano di supportare un sistema operativo oramai standard de-facto (uno dei).
- Il mobile, i tablet, i futuri dispositivi di uso quotidiano, i «Very personal computers»…
  - …promettono di divenire gli strumenti più diffusi per la grande massa.. Avendo già oggi enorme diffusione.

Dal punto di vista dello sviluppatore

- Chi ha sviluppato app per dispositivi mobili di "vecchia" generazione sa che:
  - il panorama dei dispositivi mobili è enorme,
  - testare una applicazione su tutti i dispositivi è un lavoro titanico,
  - l'hw è svariato, ma anche i so/framework sono tanti e molto diversi da utilizzare per sviluppare applicazioni,
  - il supporto a java (JME) da parte del mobile ha permesso l'esistenza di applicazioni (quasi) portabili..
  - ..ma non del tutto e non sufficientemente robuste.
- Cosa serve?
  - un sistema operativo unico, modulare, customizzabile dai produttori di dispositivi, che supporti tutte le periferiche presenti su mobile, robusto, aperto, …
  - un framework standard di sviluppo che permetta la realizzazione di applicazioni portabili e "controllate".

Android offre

- nuove possibilità e prospettive per le applicazioni mobili grazie a un ambiente di sviluppo aperto basato su un kernel Linux open source;
- accesso all'hardware disponibile per tutte le applicazioni attraverso una serie di librerie di API;
- supporto completo per la comunicazione e l'interazione tra applicazioni ed i relativi dati

Android non è:

- Una Implementazione di Java ME: le applicazioni Android sono scritte utilizzando il linguaggio Java ma non sono eseguite attraverso una Java ME virtual machine; Classi java e eseguibili non gireranno nativamente su Android.

Google vs Apple

- La risposta di Google all'iPhone: iPhone è un dispositivo proprietario cosi come lo è la piattaforma software, entrambi rilasciati da una unica azienda (??);
- Android è invece uno stack software open source prodotto e supportato da Open Handset Alliance e progettato per funzionare su ogni sistema mobile con le caratteristiche richieste.

Ecco cosa è android

- Android è un sistema operativo per dispositivi mobili sviluppato da Google, progettato principalmente per sistemi embedded quali smartphone e tablet, con interfacce utente specializzate per televisori (Android TV), automobili (Android Auto), orologi da polso (Wear OS), occhiali (Google Glass), ed altri dispositivi.

Background Java

- Java è un linguaggio portabile:
  - il codice sorgente (alto livello di astrazione) viene compilato in linguaggio macchina (basso livello di astrazione, bytecode) per una macchina virtuale il cui microprocessore non esiste;
  - una macchina virtuale (JVM) interpreta il bytecode per la CPU su cui la JVM viene eseguita;
  - il Just In Time compiler (JIT) traduce «al volo» (e ottimizza) il bytecode, fornendo performance migliori rispetto alla compilate per CPU «generiche».
- Java è Object Oriented:
  - esistono i concetti di classe, interfaccia, oggetto, enumerazione, annotazione, package, polimorfismo riferimento, composizione, meccanismo, pattern, …

Riguardiamo assieme alcuni concetti

- Classi, classi astratte, interfacce:
  - diversi livelli di astrazione:
    - classi → implementazione concreta
    - classi astratte → implementazione parziale
    - interfacce → dichiarazione di capacità
  - un occhio all'ereditarietà e al polimorfismo:
    - ereditarietà = riciclo del codice
    - polimorfismo = riciclo con modifiche
    - incapsulamento: è proprio legato al concetto di "impacchettare" in un oggetto i dati e le azioni che sono riconducibili ad un singolo componente.

La vecchia virtual machine Dalvik

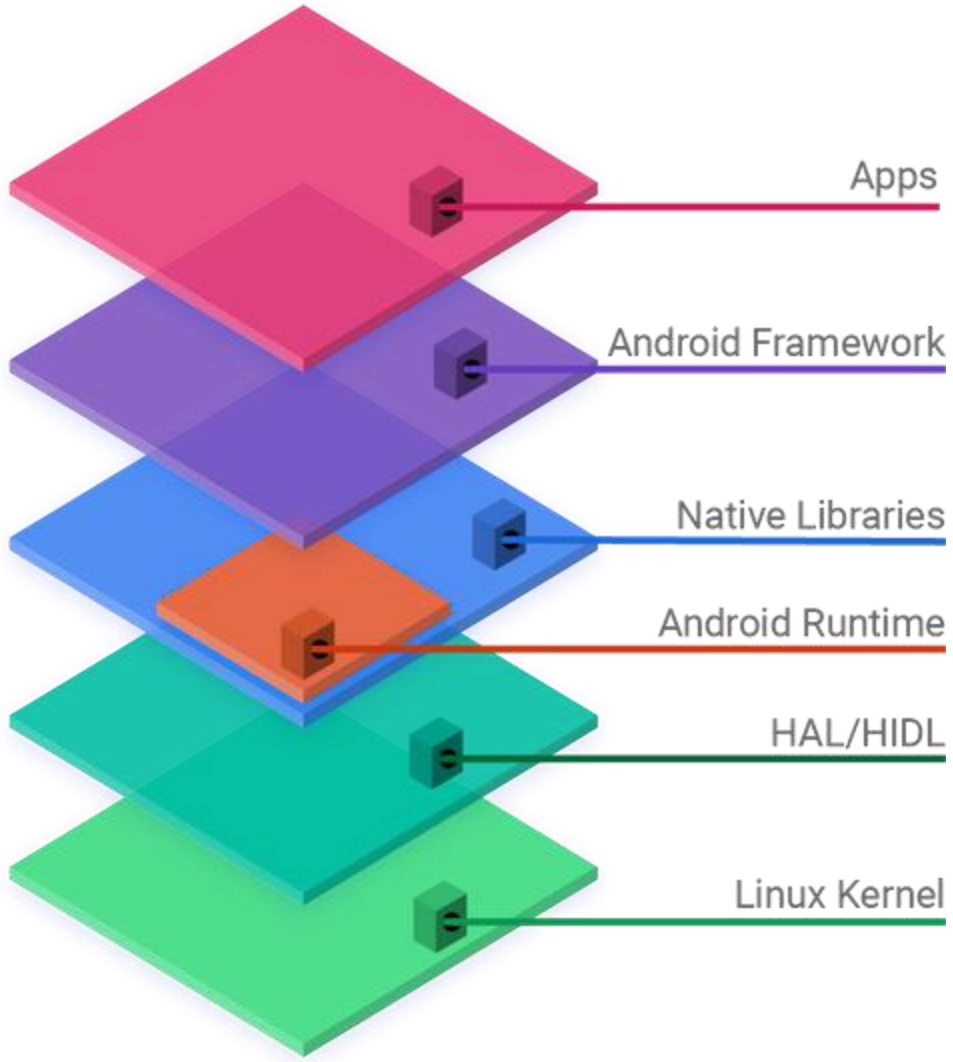
- E’ basata su tecnologia JIT (just-in-time): ogni app quindi viene compilata solo in parte dallo sviluppatore e sarà poi di volta in volta compito di un interprete software (proprio la Dalvik) eseguire il codice e compilarlo definitivamente in linguaggio macchina in tempo reale, per ogni esecuzione dell'app stessa.
- Questo ovviamente incide sulle prestazioni anche se permette una maggiore versatilità nello sviluppo di applicazione che girano su più piattaforme.

ART – Android Run Time

- E’ basata su tecnologia AOT (Ahead-Of-Time) che esegue l'intera compilazione del codice durante l'installazione dell'app e non durante l'esecuzione stessa del programma (JIT). Un vantaggio quindi in termini di prestazioni e gestione delle risorse.
- Tuttavia questo incremento di prestazioni e gestione delle risorse in fase di esecuzione incide con un maggior tempo per l'installazione di un app.
- Si tratta comunque di tempi quasi impercettibili, specie in vista di hardware sempre più potente

Cosa ci OFFRE Android

- Ci fornisce:
  - un framework comune programmabile in un linguaggio di programmazione unico e portabile,
  - strumenti di sviluppo (SDK) e di emulazione standard,
  - librerie di utilità comune contenenti API di base,
  - una «struttura» per le apps «incasellandole in un framework comune», non disomogenee tra loro,
  - strumenti per gestire periferiche, comunicazione, sicurezza, storage, interazione fra apps…
- Prevede la realizzazione/utilizzo di:
  - Attività → porzioni di GUI
  - Servizi → attività "di background"
  - Persistenza → astrazione sulla persistenza dei dati (capacità che hanno i dati di un programma a sopravvivere all'esecuzione del programma stesso che li ha generati)
  - Notifiche → da mostrare all'utente permettendo interazione



Applicazioni Native

- I "telefonini" Android vengono normalmente forniti con una suite di applicazioni preinstallate che includono almeno:
  - Un Client per la posta elettronica compatibile con Gmail ma non limitato ad esso.
  - Un'applicazione per la gestione di SMS.
  - Una suite PIM (Personal Information Management) che include: calendario e lista di contatti pienamente integrati con i servizi on-line di Google. La versione mobile di Google Maps compatibile al 100% che include StreetView.
  - Un Browser web basato su web-kit.
  - Un client di IstantMessaging.
  - Un lettore musicale ed un viewer per immagini.
  - Un client per l'acceso al marketplace di Android.

Applicazione native di android

- Tutte le applicazioni native di android sono state scritte in Java utilizzando l'SDK di Android e sono eseguite nell’AndroidRunTime environment.
- I dati memorizzati ed utilizzati dalle applicazioni native su android, esempio i dettagli dei contatti, sono anche disponibili per le applicazioni terze-parti.
- Analogamente tali applicazioni possono gestire eventi quali l'arrivo di una chiamata o di un nuovo SMS

Applicazioni di terze parti

- Il vero appeal dell'ambiente di sviluppo su piattaforma Android sta nelle API offerte agli sviluppatori e ad alcune caratteristiche di seguito riportate:
  - Nessuna tassa/fee per licenza, distribuzione e sviluppo (si paga una sola volta l'iscrizione alla comunità di sviluppatori Android).
  - Accesso "completo" all'hardware.
  - Utilizzo delle reti disponibili per effettuare e ricevere chiamate e/o SMS e per scambiare dati nella rete mobile.
  - Set completo di API per servizi location-based (GPS).
  - Android include API per il forward ed il reverse geocoding che consentono di trovare le coordinate di un indirizzo ed un indirizzo date le coordinate.
  - Controllo completo dell'hardware per il multimedia (camera, microfono, altoparlante): playback e registrazione dalle periferiche.
  - API per accelerometro e bussola.

Background Services:

- I moderni dispositivi mobili sono per natura multitasking.
- Ma la limitata dimensione dello schermo, e a volte scelte implementative che non prevedono l'esecuzione in background, limitano il numero di applicazioni che possono essere eseguite "contemporaneamente".
- Android supporta l'esecuzione in background di applicazioni e servizi. Questo consente di lanciare delle applicazioni che eseguono operazioni senza input diretto dell'utente.
- Un applicazione che fa uso di questo servizio potrebbe essere per esempio event-driven.

Condivisione dei dati e comunicazione interapplicazione

- Android fornisce tre tecniche che consentono di trasmettere informazioni e dati da una applicazione per essere utilizzati altrove:
  - **Notification**: rappresenta la tecnica standard secondo la quale un dispositivo "allerta" un utente. Utilizzando le API opportune è possibile attivare per esempio Allarmi Audio, attivare la Vibrazione attivare il Led o usare la barra di stato delle notifiche.
  - **Intents**: fornicono un meccanismo per il "messagepassing" all'interno di applicazioni e tra applicazioni. Utilizzando tale tecnica è possibile mandare in broadcast delle azioni per consentirne la gestione da parte di altre applicazioni (es. risposta ad una chiamata).
  - **ContentProviders**: fornisce una via per accedere in maniera "managed" al database privato di una applicazione.

SQLite Database

- Android fornisce un database relazionale per ogni applicazione grazie all'utilizzo di SQLite.
- Con questo servizio è possibile memorizzare e reperire informazioni e dati in maniera rapida, efficiente e sicura.
- Per default ogni database è accessibile dalla sua applicazione. I contentProviders forniscono,come accennato, un meccanismo per condividere questi database.

Gestione Ottimizzata di processi e memoria

- In maniera simile a Java e .NET, Android usa il suo ambiente runtime e la virtual machine per la gestione della memoria di una applicazione.
- Diversamente dalle altre tecnologie Android gestisce anche la vita di un processo.
- Android assicura la responsività di una applicazione stoppando o killando processi per liberare risorse richieste da applicazioni con più alta priorità. Fare attenzione a questi particolari durante la fase di progettazione di una applicazione.

Supporto per grafica e media

- Supporto ottimizzato per grafica 2D e 3D grazie ad OpenGL.
- Media Libraries.
- Un framework di sviluppo orientato alle componenti ed al relativo riuso.
- Possibilità di rimpiazzare le applicazioni native con applicazioni utente custom.

Architettura di Android