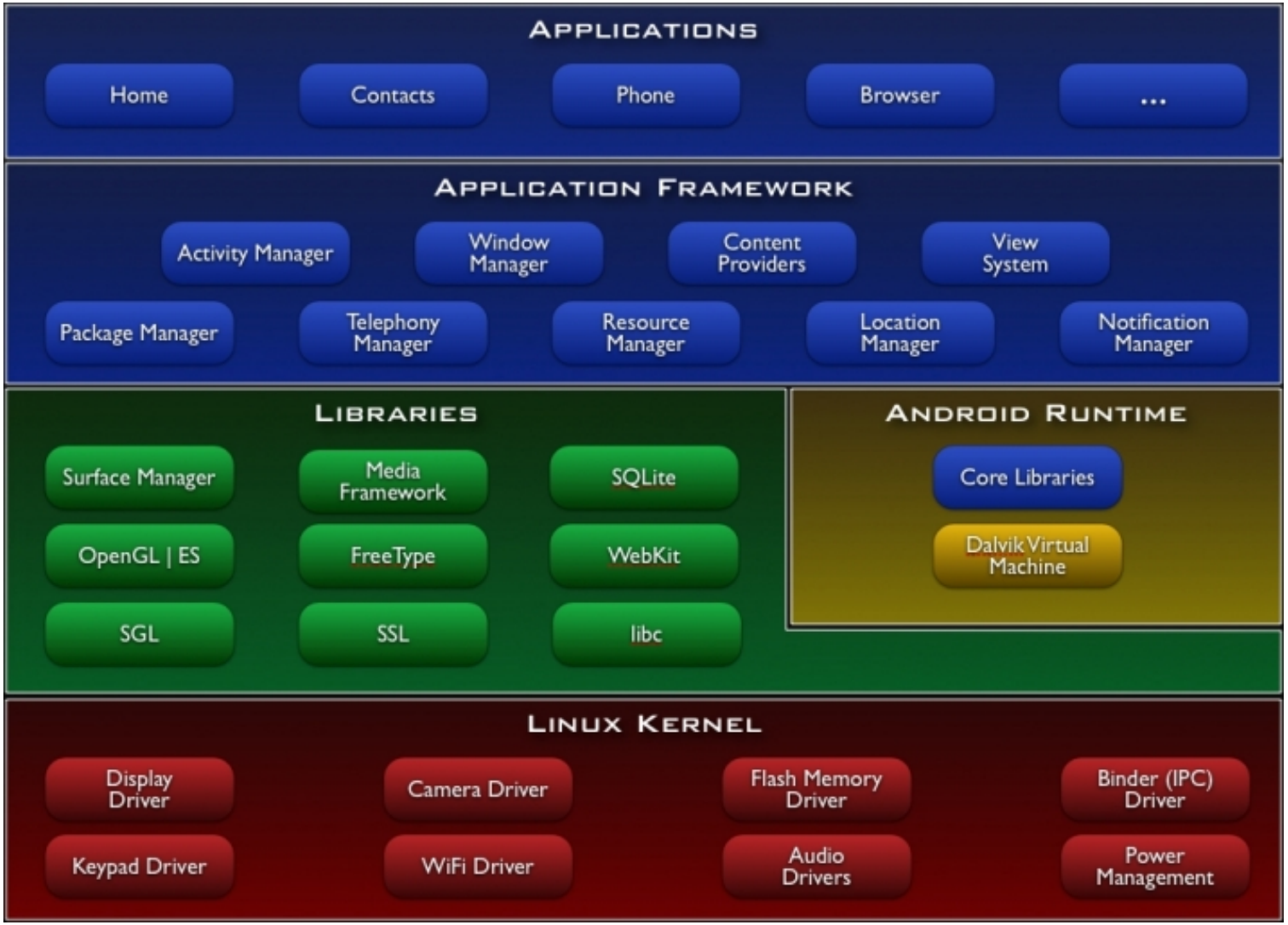
Un po' di architettura

The big picture



Studieremo a fondo

Studieremo per diletto

Largamente invisibili

Utile da sapere

Il Kernel

- Alla base di tutto c'è un **kernel Linux**
- Si tratta di un kernel completo, con tutte le primitive UNIX a cui siamo abituati
  - Processi, gestione della memoria, IPC, thread
  - Filesystem, utenti, diritti
  - Librerie, shell, comandi
  - Driver (sotto forma di moduli) per vari device
    - Tipicamente, quelli presenti in ogni particolare dispositivo
    - SD card, reti, telefonia, sensori, ecc.

Applicazioni native

- È possibile scrivere applicazioni che chiamano direttamente il kernel
  - Direttamente (via syscall) o via librerie (es., stdio.h)
  - Il codice deve essere compilato per il particolare processore in uso su un certo telefonino
    - In genere, ARM – ma non forzosamente
  - Deve poi essere “impacchettato” in un formato specificato per la distribuzione/installazione
    - Non troppo diverso dai vari .rpm, .deb, e simili
- Sconsigliato! (noi lo vedremo in fondo)

Dalvik

- La stragrande maggioranza delle applicazioni gira in una **macchina virtuale**: **Dalvik**
- È una versione di JVM con importanti differenze
  - Basata su registri (non su stack)
  - Set di istruzioni ottimizzato per risparmiare memoria e aumentare la velocità di esecuzione
  - Formato dei file eseguibili ottimizzato per risparmiare memoria
  - Eseguibile da più processi con una sola istanza
    - Tutto codice **rientrante** – sharing del codice di Dalvik via mmap()
  - **Non** sotto il controllo di Oracle (che infatti ha perso la causa)

Linux, Dalvik, e sicurezza

- Ogni App viene eseguita dal kernel Linux
  - **In un processo separato**
    - Che esegue Dalvik che esegue il bytecode dell'app
    - Controllo dei **permessi** di accesso alle risorse logiche fatto dalla VM (i permessi sono concessi dall'utente-umano)
  - **Con uno user ID distinto**
    - Tutti i file creati dall'applicazione appartengono al “suo” user ID; altre applicazioni non possono accedere alla “sua” directory, né leggere i “suoi” file
    - È possibile che applicazioni *amiche* condividano processo e user ID – occorre però che siano firmate dallo stesso autore
    - Controllo dei **diritti** di accesso alle risorse fisiche fatto dal kernel (i diritti **non** sono controllati dall'utente-umano)

Linux, Dalvik, e sicurezza

- Risultato complessivo
  - Notevole grado di **separazione** e **isolamento** delle Applicazioni
    - Ci può sempre essere un buco di sicurezza non patchato nel kernel Linux, ma l'uso dello stesso kernel usato per tutte le altre applicazioni rende l'eventualità remota
  - Android è un sistema piuttosto **sicuro**, ma...
    - Sono sempre possibili exploit basati sull'**ingegneria sociale**
      - **Una App convince l'utente darle particolari permessi**
      - **La App usa poi questi permessi per uno scopo diverso da quello pubblicizzato**
    - Molto più difficili gli attacchi veri

Linux, Dalvik, e sicurezza

- In ambiente UNIX tradizionale, abbiamo
  - Le applicazioni (comandi) appartengono al sistema
  - Più utenti (umani) usano il sistema
    - Mutuamente malfidati
  - Il dominio di protezione è l'**utente**
- Su Android invece
  - C'è un solo utente umano, fidato ma inaffidabile
  - Le applicazioni sono mutuamente malfidate
  - Il dominio di protezione è l'**applicazione**

Linux, Dalvik, e sicurezza

- I veri utenti (nel senso UNIX) sono **i programmatori delle varie App**
- Il proprietario del telefonino è (nel senso UNIX) quasi un **device** :-)
  - Non ha un suo userID
  - Non è proprietario di nessun file nel file system
  - Non è titolare di nessun processo
  - Non ha nemmeno un login e una password!
- L'utente non fa niente, se non tramite App!

DDD

- Di tanto in tanto, presenteremo degli approfondimenti tecnici
- Non è indispensabile seguire tutti i dettagli per programmare con successo su Android!
- Ma per gli informatici più incarogniti, si tratta di conoscenze utili
- Prima puntata: **Dalvik Internals**
  - Materiale gentilmente fornito da **Dan Bornstein** di Google