UNIVERSIDAD NACIONAL DE LA PLATA

Sistemas Operativos

Práctica 6

Preguntas teóricas

Conceptos generales

1. ¿Qué es **Android**?
2. ¿Qué versión del *kernel* de **Linux** se esta utilizando para desarrollar Android?
3. ¿Qué aprovecha Android de Linux?
4. ¿Cuál es la mascota de Android?, ¿Cual es su etimología?
5. ¿Cuándo y por quién nació Android?
6. ¿Cuándo fue comprado por la empresa **Google**?
7. ¿Qué es la **OHA**?
8. ¿En qué año se lanzo la primer versión estable?
9. ¿Cuál es la última versión estable de Android?
10. ¿Con qué frecuencia se lanzan las versiones del *SDK* de Android?
11. Lea esta[1] página y conteste:
    (a) ¿Qué rol cumple la capa del *kernel de Linux*?
    (b) ¿Qué rol cumple la *HAL*?
    (c) ¿Qué rol cumple la capa de *librerias*?
    (d) ¿Qué rol cumple la capa de *Android Runtime*?
    (e) ¿Por qué fue reemplazada la *DVM*?, ¿Qué mejora?
    (f) ¿Qué rol cumple la capa de *Application Framework*?

Aplicaciones

1. ¿Qué componentes de una aplicación Android existen? ¿para que sirve cada uno?
2. ¿Qué es *Gradle*?, ¿Qué tarea permite realizar?
3. ¿A qué hacen referencia y para qué sirven las varibles *compileSdkVersion*, *minSdkVersion*, *targetSdkVersion* y *buildToolsVersion*? ¿tienen que tener alguna relación entre ellas?. ¿Y la variable *versionCode*? ¿que impacto tiene en la distribución de una aplicación a través de Google Play Store?

[1] https://developer.android.com/guide/platform/index.html

4. ¿Todas las aplicaciones tiene que estar firmadas digitalmente para ejecutar en un dispositivo?, ¿A través de que mecanismo se realiza la firma digital?, ¿A través de que herramienta se lo puede hacer?
5. ¿Qué es el *application sandbox*?
6. ¿Existe la posibilidad de que dos aplicaciones compartan el *userId*?, ¿Qué debe respetarse?
7. ¿Qué alternativas tienen las aplicaciones para compartir su datos?.
8. ¿Dónde se define el acceso a los recursos por parte de las aplicaciones?, ¿Pueden realizarce cambios en tiempo de ejecución?, ¿Qué excepcion se lanza si una aplicación intenta acceder a un recurso sobre el cual no definio permisos?
9. ¿En que momento el usuario le da permiso a la aplicación para que esta utilice los recursos del dispositivo? ¿Se pueden definir de manera dinámica?
10. ¿Qué es un y qué contiene un APK?
11. ¿Para qué sirve el archivo AndroidManifest.xml?. ¿Qué son los archivos *.dex*?
12. Detalle el proceso de firma (signing) de un APK.

Procesos

1. ¿Qué comandos de GNU/Linux tenemos disponibles en Andoid?
2. ¿Android cuenta con un área de intercambio?, ¿por qué?
3. ¿Qué alternativa tiene a la hora liberar memoria?, ¿Cómo se clasifica los procesos?
4. Lea este[2] artículo y responda:
   (a) Por defecto, ¿en cuántos procesos/threads corren los componentes de una aplicación?
   (b) ¿Los componentes de las aplicaciones pueden correr en procesos separados y/o en el mismo proceso?
   (c) ¿Se pueden crear n threads de un proceso?
   (d) ¿Los componentes de la vista (UI) de una aplicación son *thread-safe*?, ¿Qué reglas sigue el *Android's single thread model*?
5. ¿Qué es *DVM*?, ¿Por qué fue diseñada?. Compare este concepto con el de *JVM*. Adicional mente compare el concepto de *DVM* con el de *ART* en base a este[3] y este[4] artículo.
6. ¿Qué es y de qué se encarga *Zygote*?

Almacenamiento

1. ¿Qué alternativas tiene una aplicación para almacenar sus datos?. Detalle cada una.
2. ¿En dónde se almacenan las *shared preferences* de las aplicaciones?, ¿De qué tipo pueden ser?, ¿Una aplicación podría acceder a las *shared preferences* de otra diferente?
3. ¿En dónde se almacenan las bases de datos de las aplicaciones?

[2] http://developer.android.com/guide/components/processes-and-threads.html
[3] http://www.addictivetips.com/android/art-vs-dalvik-android-runtime-environments-explained-compared/
[4] https://en.wikipedia.org/wiki/Android_Runtime

File system

1. ¿Cuáles son los puntos de montaje principales del File System de Android?, ¿Qué contiene cada uno?
2. ¿Con qué tipos de memoria cuenta un dispositivo móvil generalmente?. Detalle cada uno y luego relacionelos con los tipos de file system que cada tipo podría soportar.
3. Detalle las caracteristicas del *YAFFS*.

Licencia

1. ¿Bajo qué licencias esta el *stack* de Android?, ¿Por qué?
2. ¿Por qué compañias como **Samsung** pueden cambiar la interface de sus propias versiones de Android?
3. ¿Cómo se llama la *libc* de Android?, ¿Por qué fue implementada?

Rooteo

1. ¿Qué significa rootear un dispositivo Android?
2. ¿Qué es el bootloader?, ¿De qué se encarga?, ¿Qué tipos existen?, ¿Qué consecuencias trae desbloquearlo?
3. ¿Qué es el *fastboot*?, ¿Qué acciones permite realizar?
4. ¿Qué es la *boot.img*?, ¿Con qué herramienta se la puede dividir y con cuál crear?, ¿Qué contiene?, ¿Qué nombre tiene la imagen del *kernel* de *Linux*?, ¿y la de *Android*?. ¿En qué formato está empaquetado y comprimido el *initramfs*?

Ejercicios prácticos

Requisitos

Para poder realizar los ejercicios prácticos deberá contar con un conjunto de herramientas.
La practica esta armada para ser realizada usando Android Studio, para simplificar la instalación de las herramientas, ya que la idea no es gastar demasiado tiempo en la puesta a punto del entorno para resolver la practica. Una vez instalado, desde el IDE *Android Studio*, podrá acceder casi todas las herramientas mencionadas en esta practica. Para ello, deberá primero crear un proyecto nuevo y cuando se abra la ventana del IDE tendra acceso a las herramientas mediante la pestaña *Tools*.
Si quisiera no instalar *Android Studio*, tendría que instalar todas las herramientas que se mencionan a continuación de manera separada.

- El *Java SE Development Kit*. Descarguelo desde la última versión desde esté[5] link. Elija la arquitectura correspondiente y el formato que desee:
  - Instálelo. Puede guiarse con esté[6] sitio.
- El *SDK* de Android. Descargue la última versión desde esta[7] página:

[5] http://www.oracle.com/technetwork/java/javase/downloads
[6] https://www.java.com/en/download/help/download_options.xml
[7] https://developer.android.com/studio/#downloads

- Si descargo *Android Studio*, podrá instalar todas las herramientas necesarias de manera simple y automatizada. Para realizar esta práctica, solo tendrá que asociar los binarios correspondientes a la variable de entorno *PATH*.
- De lo contrario descargue solo las herramientas de consola. Con lo cual, tendrá que desempaquetar y descomprimir los archivos correspondientes.
- Acceda al *Android SDK Manager*[8] [9]:

  ```
  # sdkmanager
  ```

- Descargue e instale/actualice los siguientes paquetes:
  - *Android SDK Tools*.
  - *Android SDK Platform-tools*.
  - *Android SDK Build-tools*.
  - Android 10.0(29) → *SDK Platform*.
  - Android Emulator.
- Para facilitar el desarrollo de los ejercicios, agregue las siguientes herramientas a la varaible de entorno *PATH*:
  - *<android-sdk-directory>/platform-tools/adb*.
  - *<android-sdk-directory>/platform-tools/fastboot*.
  - *<android-sdk-directory>/platform-tools/sqlite3*.
  - *<android-sdk-directory>/tools/android*.
  - *<android-sdk-directory>/tools/emulator*[10].
  - *<android-sdk-directory>/emulator/emulator*

- Si no instaló Android Studio, va a necesitar instalar *Gradle* de manera manual. Descarguela desde esté[11] link. Tenga en cuenta esta tabla[12].

Crear dispositivo AVD

- Desde el IDE *Android Studio*, cree un dispositivo virtual. Para esto, abra la ventana del IDE y haga click en el boton *Tools ->AVD Manager*. Puede encontrar una guia en este link. Una vez creado, podrá iniciarlo desde este entorno (Recomendado).

  También puede crear y manejar el dispositivo mediante los siguientes comandos (si aparece, ingrese la letra *n* cuando se le realice la pregunta "Do you wish to create a custom hardware profile [no]"):

  ```
  # avdmanager create avd --name emulador-so -k 'system-images;android-29;goog
  ```

  Nota: podrá ver la lista de *targets* ejecutando:

  ```
  # android list target
  ```

- Inicie el emulador (*avd - android virtual device*):

  ```
  # emulator -avd emulador-so
  ```

[8] Para ejecutar el comando directamente agregue la herramienta "*android*" a la variable *$PATH*.
[9] Si instalo Android Studio, podrá instalar las dependencias desde una ventana gráfica.
[10] Puede encontrarse con el nombre "emulator-x86".
[11] https://gradle.org/install/
[12] https://developer.android.com/studio/releases/gradle-plugin

Acceder al dispositivo AVD

Una vez iniciado un dispositivo AVD, se puede acceder a este mediante una shell. Desde una consola ejecute:

- Acceda a la shell del dispositivo:

```
# adb shell
```

Esto creara un cliente *adb* e iniciara el servidor *adb*, el cual se comunicará con el demonio *adbd* del dispositivo ejecutandose en *background*. Una vez dentro del dispositivo, podra ejecutar el comando *su* para cambiar de modo usuario a modo root.

SQLite

Dentro de la shell del dispositivo (habiendo accedido mediante la adb shell), puede ejecutar el comando sqlite3 para manejar bases de datos sqlite3.
Para ver la ayuda del entorno, ejecute:

```
# sqlite3
sqlite> .help
```

Mediante este comando, se puede acceder a las distintas bases de datos de las aplicaciones.
Para realizar la próxima parte, deberá crear un nuevo AVD que cumpla ciertos requerimientos.

- Acceda a la configuración del dispositivo (Requiere estar en un dispositivo con Android Lollipop(5.0/5.1), puede crear otro AVD que cumpla este requerimiento):

```
# sqlite3 /data/data/com.android.providers.settings/databases/settings.db
```

- Liste todos los valores de la tabla *system*:

```
sqlite> select * from system;
```

Se puede apreciar el volumen del tono de llamada, el de las notificaciones, etc.

- Responda:

1. ¿Qué pasa si abrimos la configuración del dispositivo desde la interface gráfica (*Menu ->Settings ->Sound*) y modificamos el volumen del tono de llamada o de notificaciones?
2. ¿Y qué pasa si ejecutamos el comando *delete from system;*?, ¿Sigue funcionando el sistema operativo?, ¿Qué pasa con las configuraciones?

Almacenamiento

Utilice el mismo dispositivo AVD que para la etapa de Sqlite3 (Android 5.1 o menor).

- Acceda, mediante *SQLite*, a la información persistida por el browser. Analice la misma y borre los *bookmarks*.
- Identifique la página de inicio del browser (**Tip:** se encuentra almacenada a través de las *shared preferences*).

Tipo de memoria y File system

- ¿Qué tipo de memoria tiene el dispositivo virtual creado? (**Tip**: utilice el comando *mount* o liste el directorio `/dev/block/`)
- Agreguele almacenamiento *SD* al dispositivo AVD, si es que no lo creo con una SDCard. Abra el AVD Manager, y edite las preferencias del dispositivo para agregar la SDcard. Si desea hacerlo por comandos:

```
Use la herramienta mksdcard dentro del directorio
Tools para crear la tarjeta:
# mksdcard -l mySdCard 1024M mySdCardFile.img
Y agreguela al AVD:
# emulator -avd emulador-so -sdcard mySdCardFile.img
```

  ¿Aprecia algún otro tipo de memoria? ¿Por qué?
- ¿Qué file systems existen en el dispositivo virtual creado? (**Tip**: utilice el comando *mount*)

Aplicaciones

En Android, la forma de ejecutar aplicaciones es mediante la herramienta *am*. La misma provee la funcionalidad necesaria para ejecutar activities, services, entre otras cosas.
Para ver su ayuda, ejecute:

```
# am
```

- Ejecute la actividad *main* de la aplicación *settings*:

```
# am start -a android.intent.action.MAIN
  -n com.android.settings/.Settings
```

- ¿Cómo funciona?:
  - El primer parámetro inica que se va a hacer. En este caso, se ejecuta una aplicación (*start*).
  - El segundo parámetro inica el tipo de acción que se ejecutará (*android.intent.action.MAIN*). **Nota**: para ver más tipos de acciones visite está[13] página.
  - El tercer parámetro es la actividad que se desea mostrar (*com.android.settings/.Settings*).
- Inicie la aplicación del navegador del dispositivo, accediendo al sitio de la Facultad de Informatica:

```
# am start -a android.intent.action.VIEW -d http://info.unlp.edu.ar
```

Procesos y Usuarios

- Abra, desde la interface gráfica, el browser que viene instalado por defecto. ¿Cómo identificamos al proceso? ¿Podemos matarlo? ¿Cómo?
- Entendiendo el *sandbox*:
  1. En *Android Studio*, cree dos proyectos:

[13] http://developer.android.com/guide/topics/intents/intents-filters.html

- nombre/package: uno/unlp.so.android.uno
- nombre/package: dos/unlp.so.android.dos

2. Si existe, elimine el bloque de ámbito productivo (release), del archivo *build.gradle* dentro del directorio raíz de cada uno de los proyectos y/o módulos de los mismos:

```
buildTypes {
  release {
      runProguard false
      proguardFile getDefaultProguardFile('proguard-android.txt')
  }
}
```

3. Agregue al archivo *AndroidManifest.xml*, permisos de *Internet* para la aplicación uno:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="unlp.so.android.uno" >
    ...
    <uses-permission android:name="android.permission.INTERNET" />
    ...
</manifest>
```

4. Construya las aplicaciones mediante el IDE (Build ->Build Bundle(s) / APK(s) ->Build APK(s)).
   Tambien puede hacerlo mediante la herramienta *Gradle*, posicionándose en el directorio raíz de cada proyecto (requiere conexión a Internet):

```
# gradle assembleDebug
```

   Podrá ver los *.apk* generados en *(*<directorio-raíz-proyecto>/app/build/outputs/apk/debug/).
   Además podrá saber que tareas ya vienen configuradas a través de *Gradle* ejecutando lo siguiente:

```
# gradle tasks
```

5. Instale las aplicaciones en el dispositivo (requiere que el dispositivo/emulador este conectado/iniciado):

```
# adb install <debug-apk-generado-proyecto-uno>
# adb install <debug-apk-generado-proyecto-dos>
```

6. Ejecute cada una de las aplicaciones.
7. Acceda al dispositivo mediante el *adb* y responda:
   1. ¿Qué se información puede extraer como resultado de ejecutar los siguientes comandos?:

```
$ adb shell
# ps ( o ps -A )
# dumpsys package unlp.so.android.uno
# dumpsys package unlp.so.android.dos
```

8. Como posiblemente ya sepa, el *userId* del usuario *root* en Linux es el *0*. En la explicación se dijo que cada aplicación Android es un usuario Linux, es decir que tiene un *userId* único. Debera acceder a un archivo en el cual, entre otras cosas se define la configuración de usuarios y grupos para el sistema operativo Android. Acceda a la siguiente página[14] y responda:

[14] https://android.googlesource.com/platform/system/core.git/+/master/libcutils/include/private/android_filesystem_config.h

1. ¿Cuál es el *userId* del usuario *system*?
2. ¿A partir de qué *userId* Android concede indentificación para las aplicaciones de usuario?
3. ¿Qué *userId* se le asigna al demonio adb (*adbd*)?

Nota: El que desee hacerlo puede leer esté[15] artículo en donde se explica intrínsecamente la asignación del *userId* al momento de instalar una aplicación en Android.

[15] http://users.encs.concordia.ca/ clark/papers/2012_spsm.pdf