Desarrollo para dispositivos móviles con Android

Contenido

1 Introducción

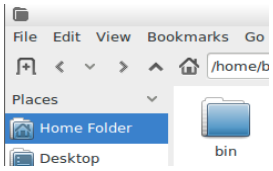
El sistema operativo **Android** es el corazón de miles de dispositivos móviles en el mundo, por delante de otros tan conocidos como **iOS** (Apple), **BlackBerry** (Research In Motion), o **Sailfish** (Jolla). Además, es capaz de escalar desde los teléfonos móviles de tipo *smartphone* (su primer objetivo), hasta tabletas, televisores e incluso relojes *smartwatch*. Es por todo esto que es el sistema operativo elegido para desarrollar en dispositivos móviles.

2 Instalación de las herramientas de desarrollo

Desde el sitio web de Google es posible descargar el software de desarrollo para **Android** (**ADK** o **Android SDK**). En el momento de escribir esta documentación, el entorno estrella para desarrollar para Android es *Android Studio*, parte de *IntelliJ IDEA* (un entorno dedicado a la programación en Java). Aunque el proceso con *Android Studio* es mucho más automatizado, en las siguientes líneas se explica cómo instalar el Android SDK por separado, que también puede ser utilizado desde otros entornos de programación, como por ejemplo *Eclipse*.

Los requisitos de una computadora para desarrollo en Android son: 2GB de RAM (8GB recomendados), SO Windows, Linux o Mac, y el JDK de Java instalado y actualizado.

2.1 Directorio de binarios



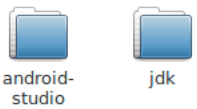
En este documento se asume que el usuario no tiene permisos de administrador en su máquina, así que se hará todo en el directorio local de su cuenta. Se aconseja, por tanto, siendo el usuario, $USR (tanto para Windows *c:\user\usuario* como para Linux */home/usuario*), crear el directorio *bin/*: *$USR/bin*. El objetivo es guardar ahí todos los binarios.

2.2 Instalación del JDK

El primer paso inexcusable es la instalación de un JDK actualizado[1]. Para ello, solo es necesario entrar en la página de Oracle y bajarse la última versión disponible del JDK. En Windows, con permisos de administrador, solamente será necesario bajarse el instalador, mientras que en Windows y Linux sin permisos se baja un archivo comprimido (.zip y .tgz, respectivamente), que se descomprimirá como *$USR/bin/jdk* (es decir, se descomprime en *$USR/bin*, y se le cambia el nombre).

Es muy conveniente crear la variable de entorno *JAVA_HOME* para que apunte a *$USR/bin/jdk*. En Windows, esto se puede realizar en "*opciones avanzadas >> variables de entorno*" de las propiedades del PC. En Linux, es necesario crear el archivo *$USR/.profile* (si no existe ya), y al final, añadir la siguiente instrucción: `export JAVA_HOME=~/bin/jdk`. A continuación, se guarda el archivo, con los cambios ya hechos. Nótese que esta variable solo estará disponible en el siguiente reinicio de la máquina.

2.3 Instalación de Android Studio

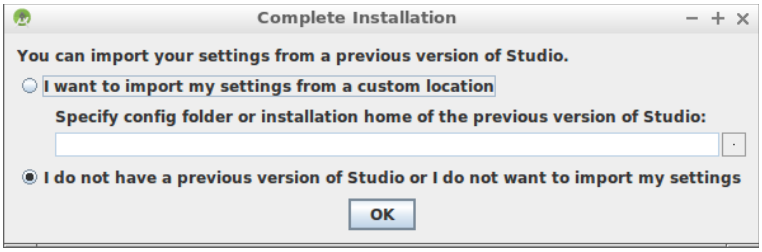


La instalación es ligeramente distinta para Windows y Linux: en el primer caso, el archivo comprimido a descargar ya incluye el Android SDK, mientras que en Linux, un wizard nos guiará en los pasos de instalación. En este documento, se seguirán los pasos a realizar en la distribución de Linux.

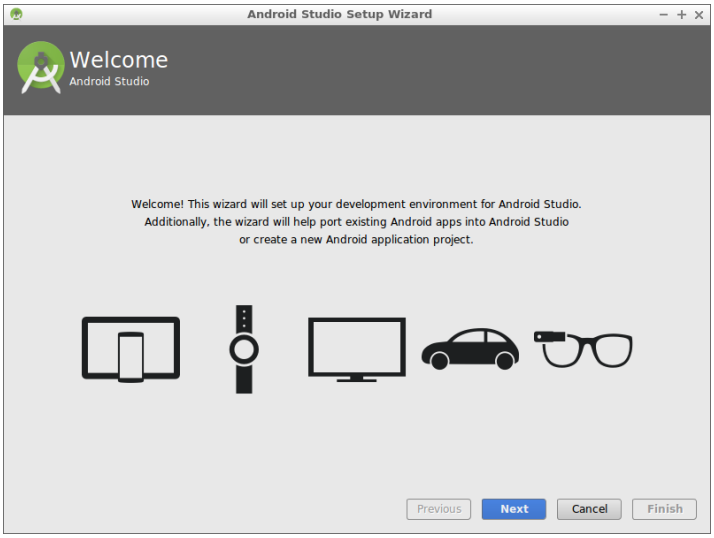
En cualquier caso, se descarga el archivo correspondiente al sistema operativo y se descomprime bajo *$USR/bin*, quedando como *$USR/android-studio*.

1 http://www.oracle.com/technetwork/java/javase/downloads/

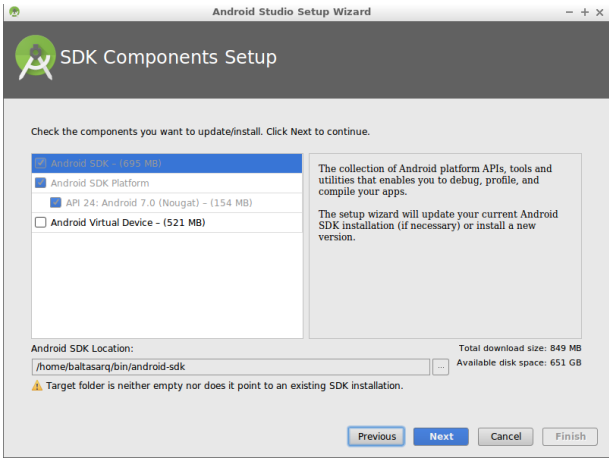
Para ejecutar **Android Studio**, será necesario entrar en *$USR/bin/android-studio/bin*, y ejecutar *studio.sh* o *studio.exe*, según el sistema operativo. Existe también un studio64.exe para equipos con Windows de 64bits. En el caso de Linux, puede ser necesario abrir un terminal en la carpeta antes mencionada, y ejecutar `"export JAVA_HOME=~/bin/jdk;./studio.sh &"`. La instrucción *export* será necesaria si no se ha reiniciado la máquina tras crear la variable JAVA_HOME. En otro caso, con *./studio.sh &* es más que suficiente.



En caso de que pida la carpeta de instalación de Java, deberemos introducir *$USR/jdk*.

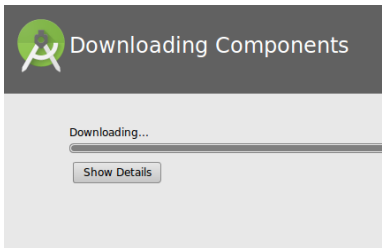


El primer paso es decidir si queremos importar una configuración previa o no. Normalmente, la opción por defecto es la más adecuada. Tras este paso, se llega a la pantalla de bienvenida del *wizard* que nos guiará en el resto de la instalación, la del Android SDK.



En la siguiente pantalla, se nos pregunta el tipo de instalación deseada: escogemos custom, de manera que podamos cambiar todas las posibles opciones, especialmente, el PATH de descarga del Android SDK.

Escogemos *$USR/bin/android-sdk* como carpeta de destino. El resto de opciones se dejan como están. Así, en la carpeta $USR/bin tendremos tres carpetas con los archivos binarios necesarios para las tres distribuciones: **Java JDK**, **Android Studio**, y **Android SDK**.



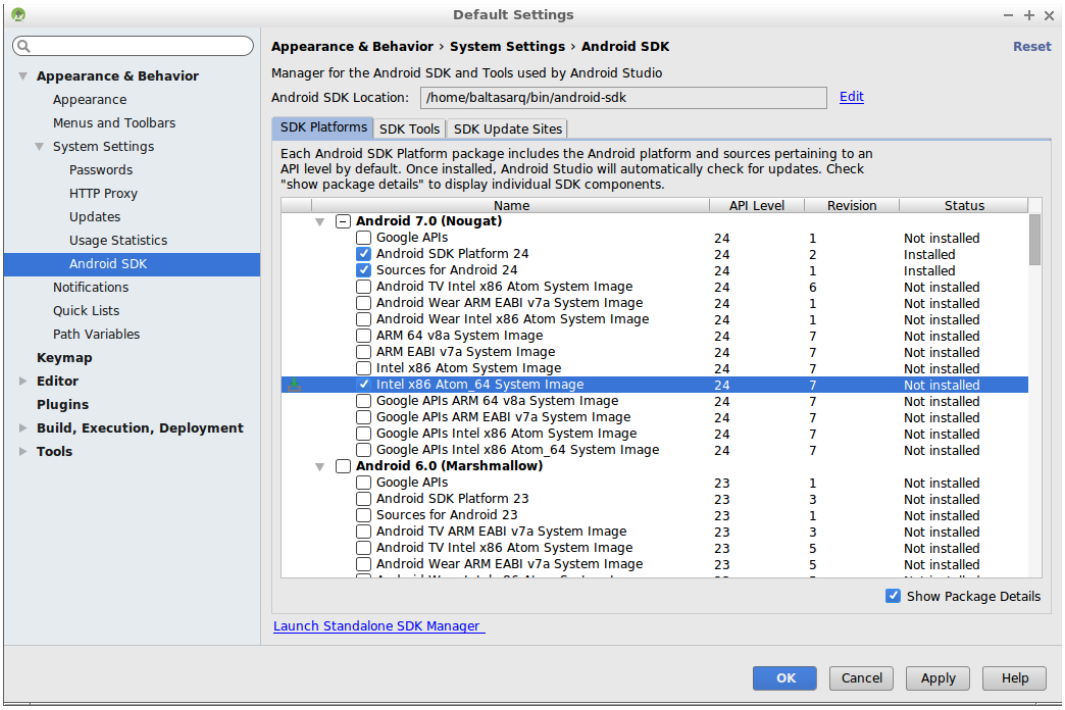
Tras un resumen de las opciones, **Android Studio** empieza a descargar el ADK. Este es un proceso que normalmente toma bastante tiempo, pues el ADK es bastante pesado. Tras este paso, Android Studio mostrará la pantalla de bienvenida, y será necesario realizar las últimas descargas y configuraciones, que afectarán a las posibilidades de distintos emuladores para probar las aplicaciones, así como los emuladores en sí mismos.

Una vez terminado el proceso de descarga, se llega a la pantalla de bienvenida normal de Android Studio.

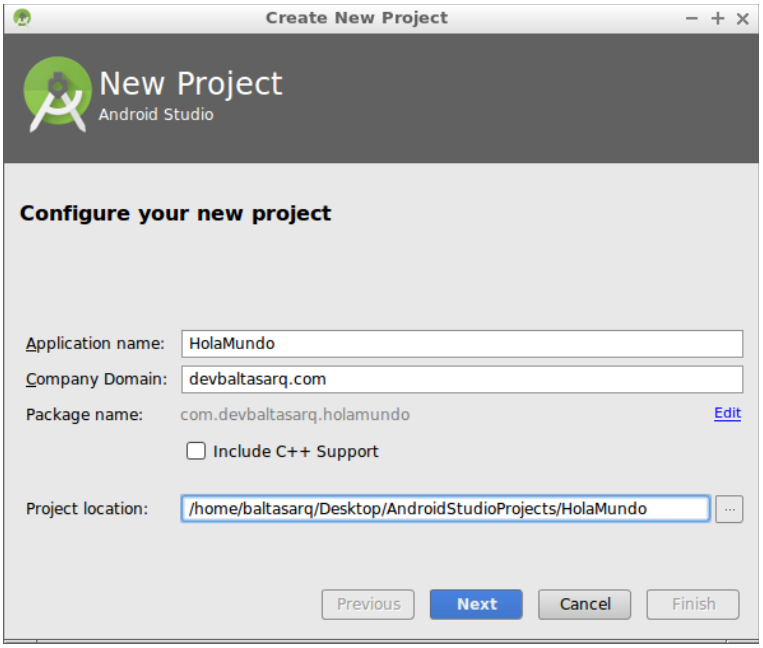
En la parte inferior, se despliega *configure*, y se seleccionar **SDK Manager**. En la siguiente ventana, se puede seleccionar *launch standalone SDK manager*, con lo que es posible lanzar el *SDK manager* de Google. En las siguientes secciones se trata el SDK Manager de Google, por lo que a continuación se explica cómo trabajar con el **SDK Manager** propio de **Android Studio**.

La ventana del gestor nos mostrará la última versión de Android ya instalada (la 7 en el momento de escribir este documento). En realidad, esto no es cierto, pues para poder ejecutar un emulador en el que probar los programas, es necesario bajarse una determinada imagen (para Intel Atom o ARM, básicamente). Así, seleccionaremos la opción *show package details* para poder escoger opciones de mayor detalle. Una vez hecho esto, podremos ver que la API efectivamente ya está instalada, aunque no está instalada ninguna imagen. Si el objetivo es desarrollar para teléfonos o tablets, con tener instalada la imagen *Intel x86 Atom System Image* o la *Intel x86_64 Atom System Image* (para equipos de 32 y 64 bits, respectivamente), es más que suficiente. Las Google APIs son imágenes que incluyen las API's para los servicios de Google, como Maps, GMail u otros. Si no se plantea desarollar para estos servicios, no son necesarias. Las Android TV son imágenes especializadas para televisores que incoporan Android como sistema operativo, mientras que Android Wear es la

imagen utilizada en smartwatches (relojes inteligentes).

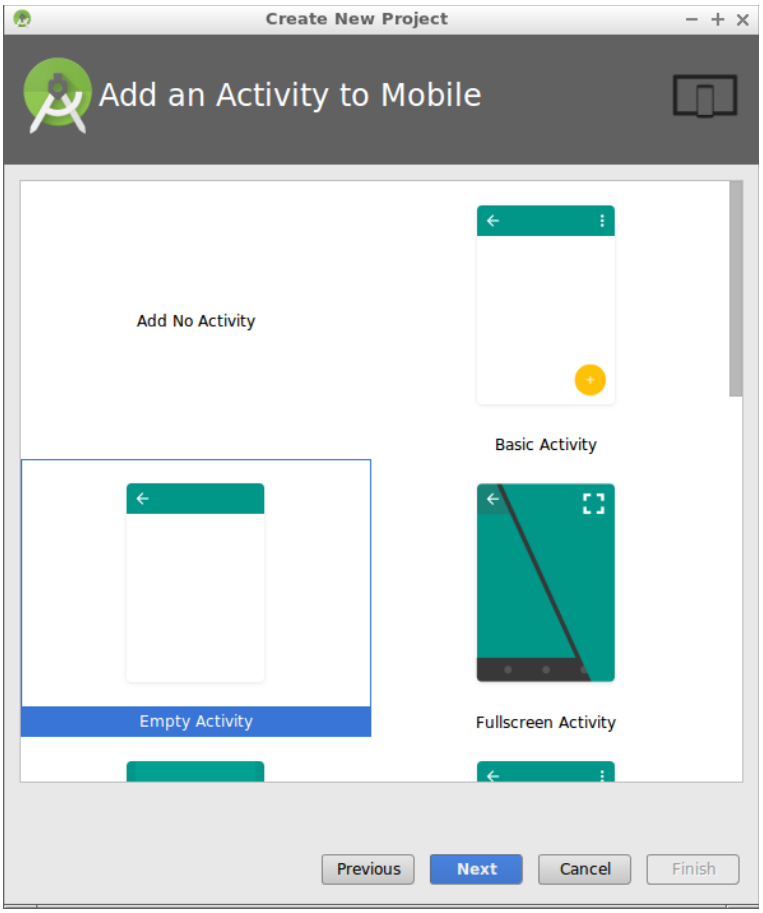


Tras aceptar el acuerdo de licencia, se descarga la imagen, con lo que ya es posible crear un emulador para un teléfono Android. Y para ello, crearemos una aplicación “hola, mundo”. De vuelta en la pantalla de bienvenida, seleccionamos la opción “Start a new Android Studio project”. En la siguiente pantalla rellenamos la información necesaria, como el nombre de la aplicación, que será *HolaMundo*, y el directorio donde se creará.

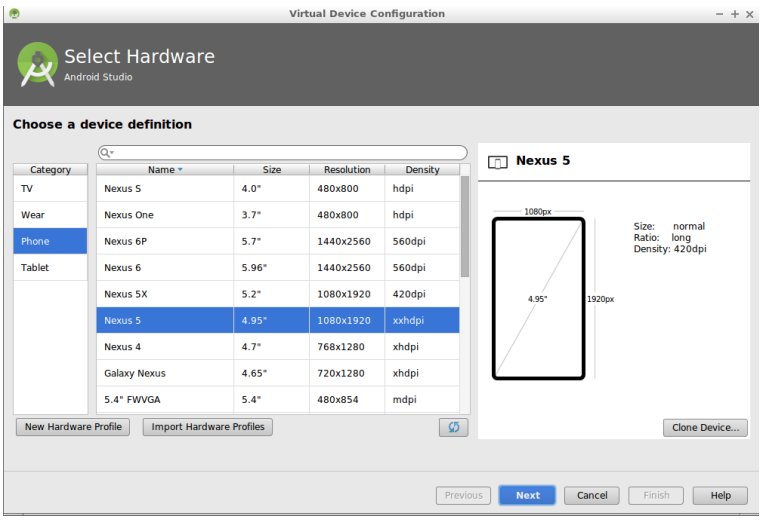


A continuación, se nos pregunta por el tipo de aplicación a crear: *Phone and Tablet* es la opción por defecto, mientras que también es posible desarrollar para otro tipo de dispositivos como televisores,

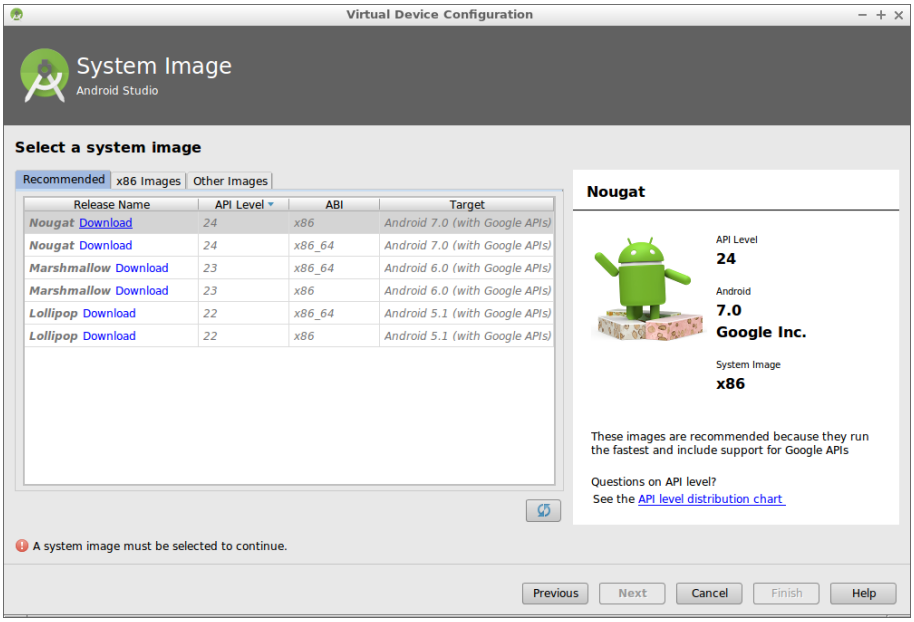
relojes inteligentes… Es también en este momento en el que se nos pregunta por la versión de Android mínima para la que trabajará la aplicación. La versión mínima ofrecida por el entorno es la más adecuada para que las aplicaciones desarrolladas funcionen en la mayor parte de los teléfonos existentes. Se pueden dejar todas las opciones por defecto y pulsar en *next* (siguiente).



En la siguiente pantalla es posible que Android Studio realice una descarga de componentes extra. Después, se selecciona una actividad (básicamente, una pantalla de Android) en la que comenzará la aplicación. Seleccionamos *empty activity* (actividad vacía) y pulsamos *next*. Allí, nos permite cambiar el nombre de la actividad, que por defecto es **MainActivity**. Pulsamos en *finish*



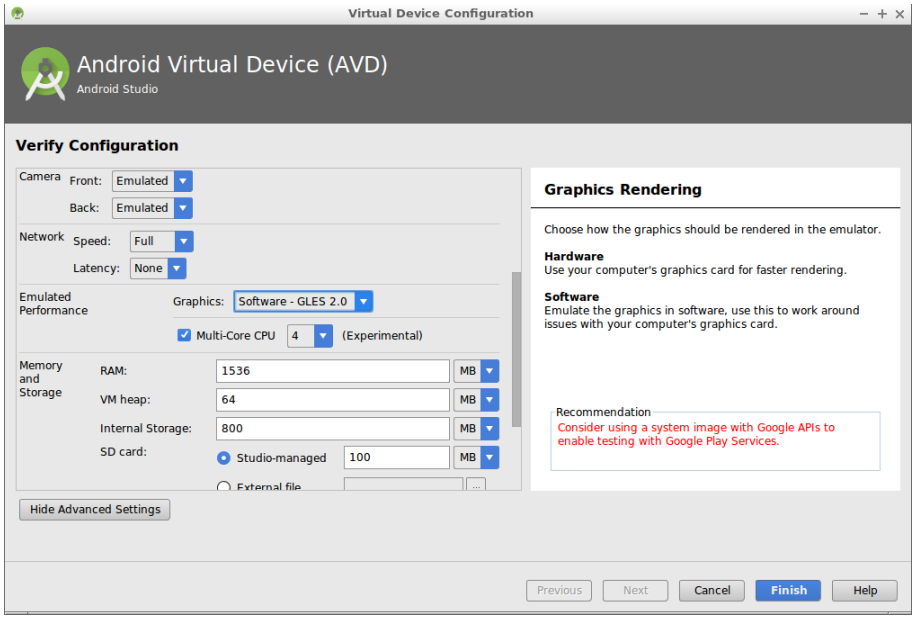
(terminar).



Después de un cierto tiempo, **Android Studio** habrá terminado la creación del proyecto, que podremos ejecutar pulsando en el símbolo play de la barra de herramientas. Como todavía no hemos creado ningún emulador, pulsamos en "*Create new virtual device*" en la ventana de elección de emulador. Entonces se abrirá una ventana con las especificaciones de diferentes modelos. Un modelo con un tamaño de pantalla similar a los móviles en venta hoy en día es el Nexus 5, con una pantalla de casi 5".

En la siguiente ventana, podremos seleccionar la imagen a utilizar en el emulador. Ya nos hemos bajado la imagen *Intel Atom x86* o *Intel Arom x86_64*, así que en la pestaña *x86 images* podemos seleccionarla y continuar. A continuación aparece la última pantalla que es de confirmación, en la que simplemente pulsamos *finish*.

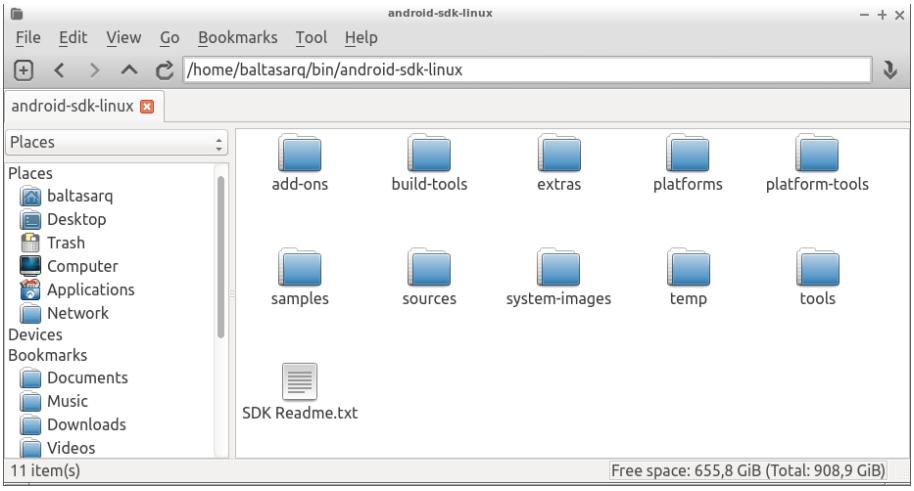
De vuelta en la ventana de selección de dispositivo para ejecución, seleccionamos la imagen que acabamos de crear. La ejecución de la aplicación se lanzará una vez que el emulador haya arrancado.



Es posible que el emulador no consiga arrancar, ya que por defecto, se hace una emulación por hardware, aprovechando las posibilidades de la máquina. Esto es problemático, especialmente si no se tienen permisos de administrador para modificar la configuración gráfica o para instalar los drivers necesarios. En ese caso, lo mejor es indicar que se realice una emulación de gráficos por software (opción *Graphics: software GLES*). Para ello, es necesario ir a *Tools* > *Android* > *AVD Manager*, seleccionar el emulador y pulsar en el lápiz para modificarlo. En la ventana dedicada a la edición del emulador, pulsamos sobre *Show advanced settings* y modificamos el apartado *graphics* como se ve en la imagen.

Aunque el emulador tarda bastante en arrancar, es posible dejarlo funcionando para ejecutar la aplicación cuando sea necesario.

2.4 Instalación de Android SDK

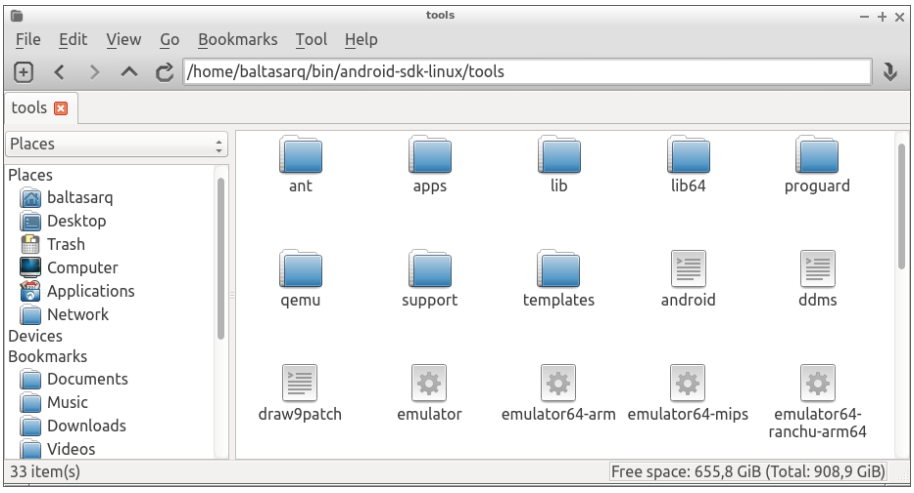


Este apartado describe pasos opcionales, véase al apartado anterior para una instalación "estándar".

Se descarga desde el sitio web de Android para desarrolladores[2], en "*download options (opciones de descarga)*", como "*SDK Tools only*", debiendo elegir el sistema operativo para desarrollar. Una vez descargado, se descomprime en una carpeta, por ejemplo, *bin/* dentro de la carpeta de usuario.

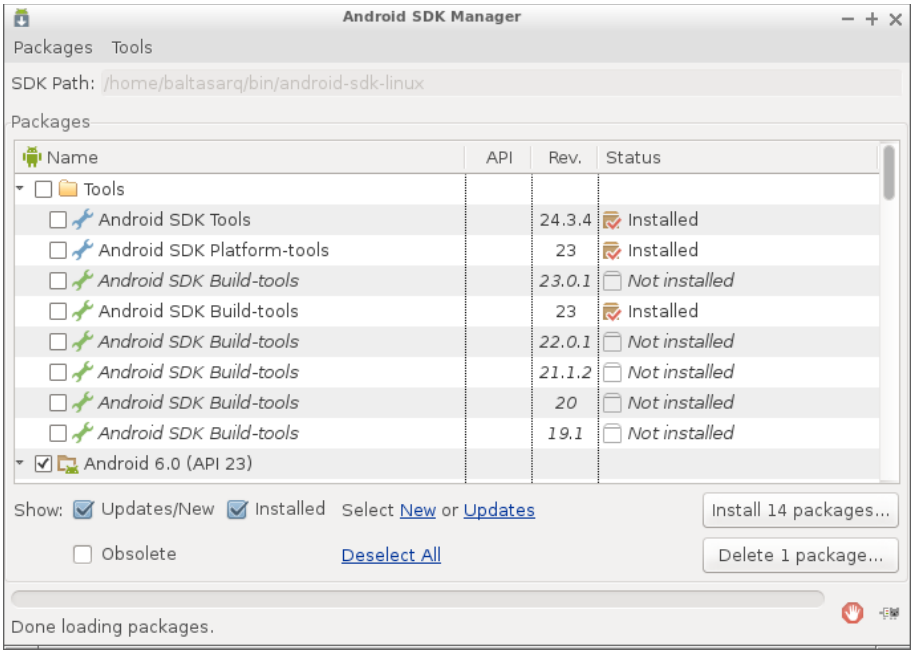
Tal y como explica el archivo *SDK Readme.txt*, en este momento no se dispone ni de lo mínimo para desarrollar. Es necesario lanzar el gestor del SDK para decidir qué librerías de API se desea descargar, así como crear un emulador adecuado (AVD o *Android Virtual Device*).

Nota importante: cada vez más, Google se inclina más a una sola descarga, del tipo "Get Android Studio (obtener Android Studio)". En caso de realizar esta descarga, se accede al Android SDK mediante el menú Tools >> Android. Por defecto, se incluye en Linux una carpeta con el Android SDK, mientras en Windows se instala en la carpeta oculta *c:\users\<usuario>\AppData\Local\Android* o similar.

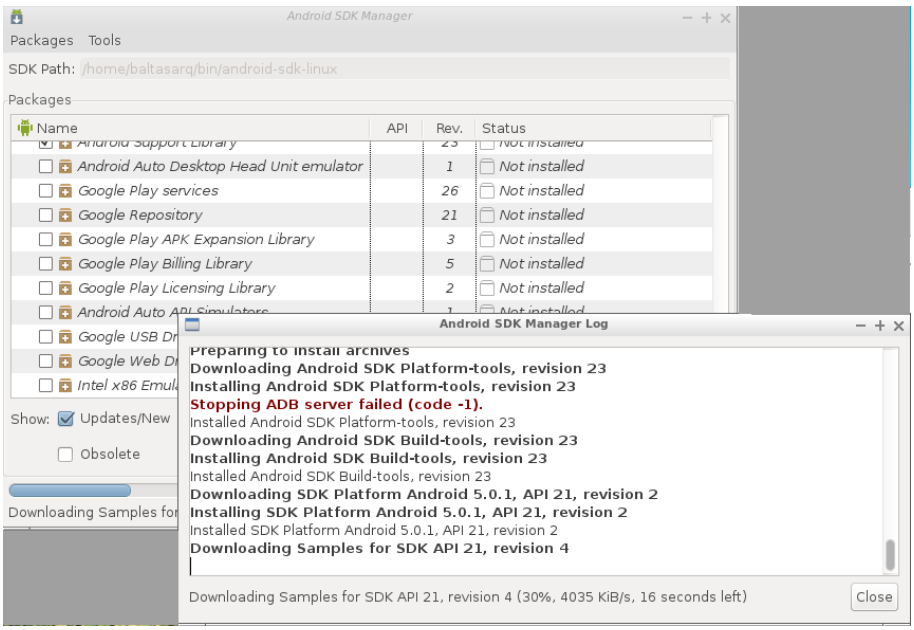


La subcarpeta *tools/* contiene el *Android SDK Manager*. En sistemas operativos como Linux, puede ser necesario añadir el atributo de ejecución tanto a los programas como a los *scripts* en esta carpeta. En cualquier caso, es necesario lanzar *android* para poder acceder al gestor del SDK.

2 http://developer.android.com/



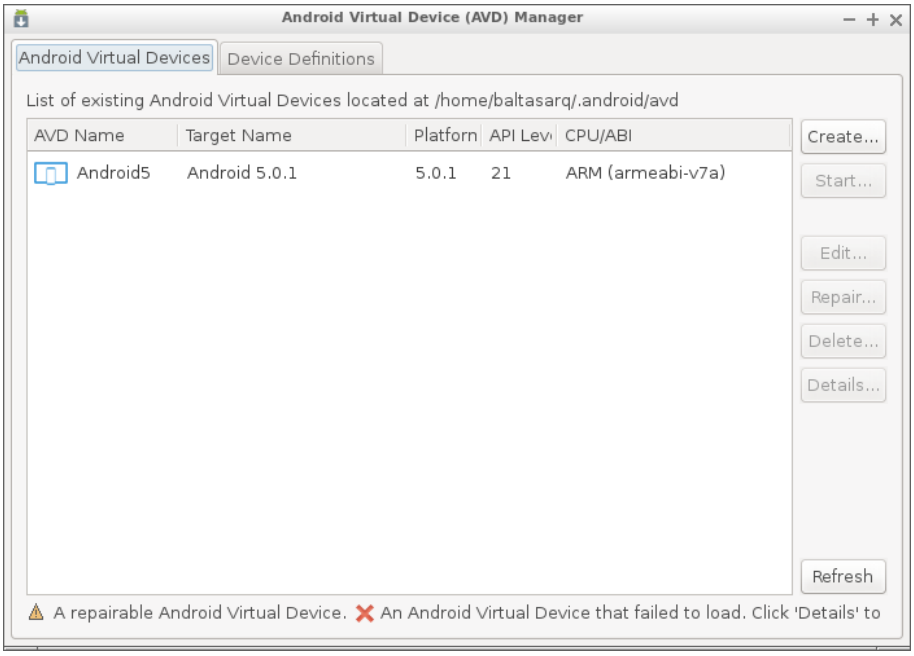
Desde esta aplicación es posible gestionar tanto los paquetes de desarrollo necesarios para construir aplicaciones para una determinada API, como los emuladores o AVDs. En un principio, es necesario seleccionar la API deseada. Por defecto, se selecciona la última (la 6.0 en el momento de escribir este documento). Dentro de los archivos de una API, se distingue entre Android APIs y Google



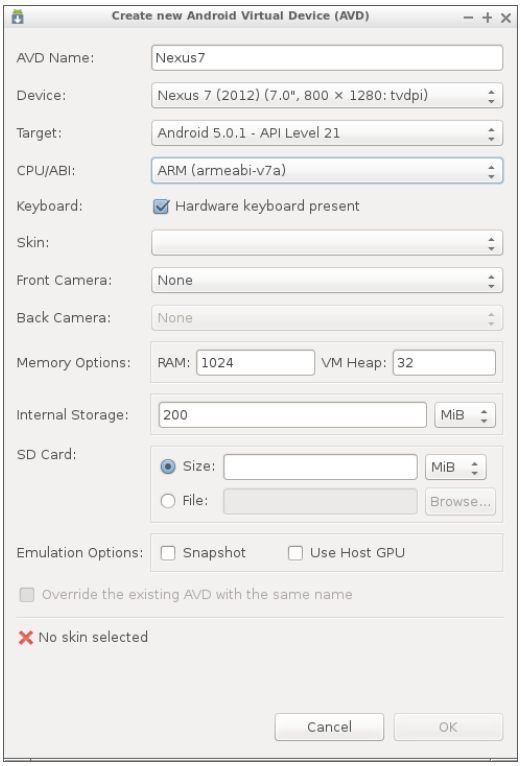
APIs. Las segundas incluyen los servicios de Google como Google Maps, y otros.

Una vez seleccionados los paquetes necesarios, se pulsa en *Install*, y se espera a que termine la instalación.

2.4.1 Creación de un emulador



Los emuladores o AVDs también se gestionan desde el *SDK Android Manager*, concretamente desde el menú *Tools >> Manage AVDs*. El gestor de AVDs permite crear nuevos emuladores (botón *Create* a la derecha), o modificar los ya existentes (*Edit*).



El primer campo a rellenar es *AVD Name*, el nombre del AVD, que puede ser cualquiera que deseemos, aunque es muy recomendable que sea expresivo en cuanto a qué tipo de dispositivo representa. Para poder ejecutar aplicaciones de la API seleccionada en el *Android SDK Manager*, es necesario que el AVD sea al menos de esa versión, o superior. El campo *device* permite seleccionar un dispositivo que será el que se emulará en concreto. El apartado Target permite seleccionar la API deseada. El contenido de este desplegable depende de las APIs que se hayan instalado, y se permite siempre elegir entre Android API y Google API. La primera es mucho más genérica, mientras que la segunda depende aplicaciones y servicios de Google, como Google Maps o Google Mail, por poner un ejemplo.

El resto de opciones tratan de la memoria y otros recursos dedicados al dispositivo. Es recomendable marcar la aceleración por hardware, “Use host GPU”, ya que hace que el emulador resulte mucho más usable. En caso de no funcionar correctamente, siempre se puede modificar el AVD y desmarcar esta opción.

Una vez dados estos pasos, el SDK está listo para usarse.

3 Depuración en el hardware real

Es perfectamente posible depurar una aplicación en un teléfono (o tablet) con Android[3], una posibilidad especialmente atractiva si el emulador es muy lento en el PC. Normalmente, esto funciona directamente, pero en cualquier caso aparecen aquí las instrucciones completas.

Para ello, es necesario activar las opciones de desarrollo en el dispositivo: en *Ajustes >> Acerca del dispositivo*, pulsa 7 veces en el número de compilación (*build number*) . Aparecerá la opción "*developer options* (opciones de desarrollo)" en el menú de ajustes. Activa "USB debugging (depuración por USB)" en opciones de desarrollo.

Desde Android Studio, localiza el archivo AndroidManifest.xml, ábrelo y añade la línea, si no existe, *android:debuggable="true"* a la etiqueta **Application**. Hay que tener en cuenta que la aplicación **no** debe ser depurable en el momento de su liberación (*release*), es decir, es necesario borrar esa línea en la versión final. Alternativamente, se puede abrir el archivo de construcción de *gradle* (la herramienta de construcción de aplicaciones), dejando su contenido de esta forma:

```
android {
    buildTypes {
        debug {
            debuggable true
        }
...
```

Una vez hecho esto, es necesario configurar el *driver* USB según el teléfono. Normalmente, todo es automático, solo es necesario hacer las configuraciones del siguiente párrafo en caso de que no funcione la instalación por defecto.

En Mac no es necesario hacer nada. En Windows solo es necesario seguir las instrucciones en "instalar driver OEM[4]". En Ubuntu Linux, es necesario modificar (como **root**, es decir, con *sudo*) el archivo */etc/udev/rules.d/51-android.rules*, añadiendo una línea con la sintaxis siguiente: *SUBSYSTEM=="usb", ATTR{idVendor}=="<vendor_id>", MODE="0666", GROUP="plugdev"*. La sintaxis ouede variar ligeramente en según qué versiones[5]. El campo "*<vendor_id>*" debe sustituirse por el id del vendedor, que aparece en una tabla en el enlace anterior. También es necesario darle al archivo ciertos atributos: *chmod a+r /etc/udev/rules.d/51-android.rules*.

4 Instalación y ejecución en el hardware real

La aplicación creada puede ser ejecutada en el dispositivo real de manera muy sencilla. Lo primero es instalar un navegador de archivos, si todavía no se dispone de uno, por ejemplo desde Google Play.

También es necesario, en *ajustes*, permitir la instalación de aplicaciones desde orígenes desconocidos.

Se conecta el dispositivo por USB, y se navega hasta la carpeta del proyecto, concretamente a la subcarpeta *out >> production >> HelloWorld*, y se copia el archivo *debug.apk* (o similar). Cópiese dicho archivo a la subcarpeta *Download* del dispositivo, por ejemplo.

Una vez copiado, se utiliza el navegador de archivos del dispositivo recientemente instalado y se pulsa sobre el icono del robot de Android (el icono por defecto de cualquier aplicación si no se especifica otro), y se permite la instalación. Solo resta abrir la aplicación desde el menú, como cualquier otra.

3 https://developer.android.com/studio/run/device.html
4 https://developer.android.com/studio/run/oem-usb.html
5 http://www.reactivated.net/writing_udev_rules.html

5 El emulador desde la línea de comando

Para trabajar con el emulador desde línea de comando, es necesario manejar los siguientes puntos:

- **El directorio de instalación de Android SDK**. Aunque en cualquiera de los sistemas operativos en los que es posible instalar Android es posible elegir la ruta donde instalar el Android SDK, por defecto, se crea en Linux una carpeta con el Android SDK, mientras en Windows se instala en la carpeta oculta *c:\users\<usuario>\AppData\Local\Android*, o similar.
- **La herramienta adb** (*Android Data Bridge*). Esta herramienta permite la comunicación con un emulador de Android, o con un terminal Android conectado mediante USB. Para distribuciones linux basadas en Arch, es necesario instalar el paquete *android-tools*, al igual que en Ubuntu, que también dispone del paquete *android-tools-adb*. Las versiones para Windows y Mac se pueden descargar de *developer.android.com*.
- **El emulador**, con al menos una imagen ADV ya creada.

Es interesante, por comodidad, crear una variable de entorno llamada ANDROID_SDK. En entornos derivados de UNIX, se consigue con *export ANDROID_HOME=~/bin/android-sdk*, suponiendo que el Android SDK está instalado en el subdirectorio *bin/* del directorio del usuario. En Windows es necesario establecer una variable de entorno local en el administrador del equipo, accesible desde el menú de inicio.

En entornos Windows, es necesario cambiar el texto $ANDROID_HOME por %ANDROID_HOME%. Tampoco se debe incluir un *ampersand* (‘&’) al final de la orden que lanza el emulador.

```
$ $ANDROID_HOME/emulator/emulator
emulator: ERROR: No AVD specified. Use '@foo' or '-avd foo' to launch a virtual
device named 'foo'

$ $ANDROID_HOME/emulator/emulator -list-avds
Nexus_5_API_24
Phablet
Pixel_API_26

$ $ANDROID_HOME/emulator/emulator -avd Pixel_API_26 &
```

Como se puede apreciar más arriba, la opción *-list-avds* lista todas las AVD’s disponibles, de manera que es posible escribir una de ellas, a continuación de la opción *-avd* para lanzar una de ellas.

Una vez que el emulador se está ejecutando, es necesario ponerse en contacto con él mediante la herramienta *adb*. Mediante esta herramienta, es posible instalar en el emulador una aplicación (en forma de archivo *apk*), mediante la opción *install*.

```
$ cd ~/Prys/Android/BurguerBuilderComplexListView/app/build/outputs/apk/
$ adb install app-debug.apk
```

La orden anterior fallará en el caso de que la aplicación ya esté instalada. En tal caso, se puede forzar la reinstalación con la opción *-r*.

```
$ adb install -r app-debug.apk
```

Una vez está instalada en el emulador (o en el dispositivo, adb también funciona sobre dispositivos reales conectados mediante USB), se puede iniciar la aplicación normalmente.

6 Referencias

- Documentación y recursos de Android para desarrolladores (accedido en sept. 2016) http://developer.android.com/
- Historia de Android (accedido en sept. 2016) http://www.android.com/history/