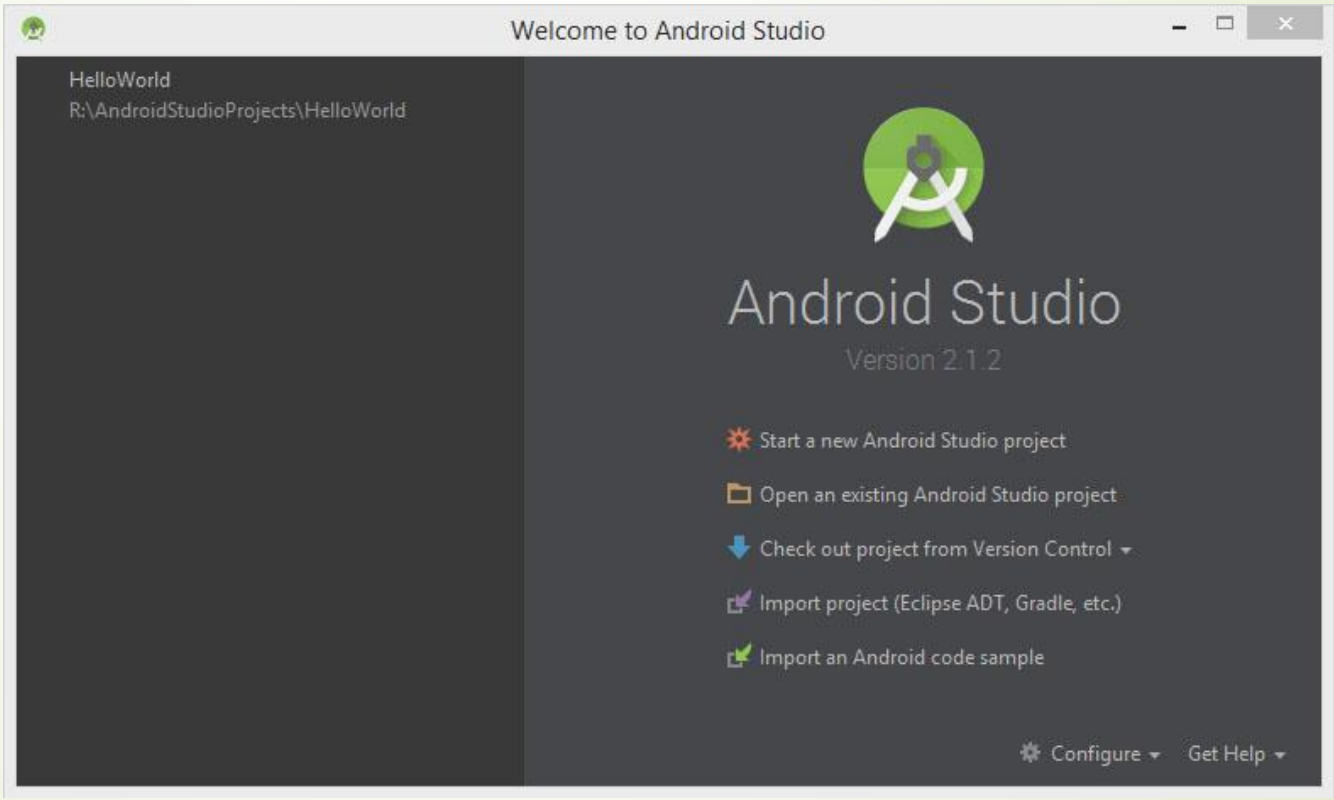
Android

Android Studio

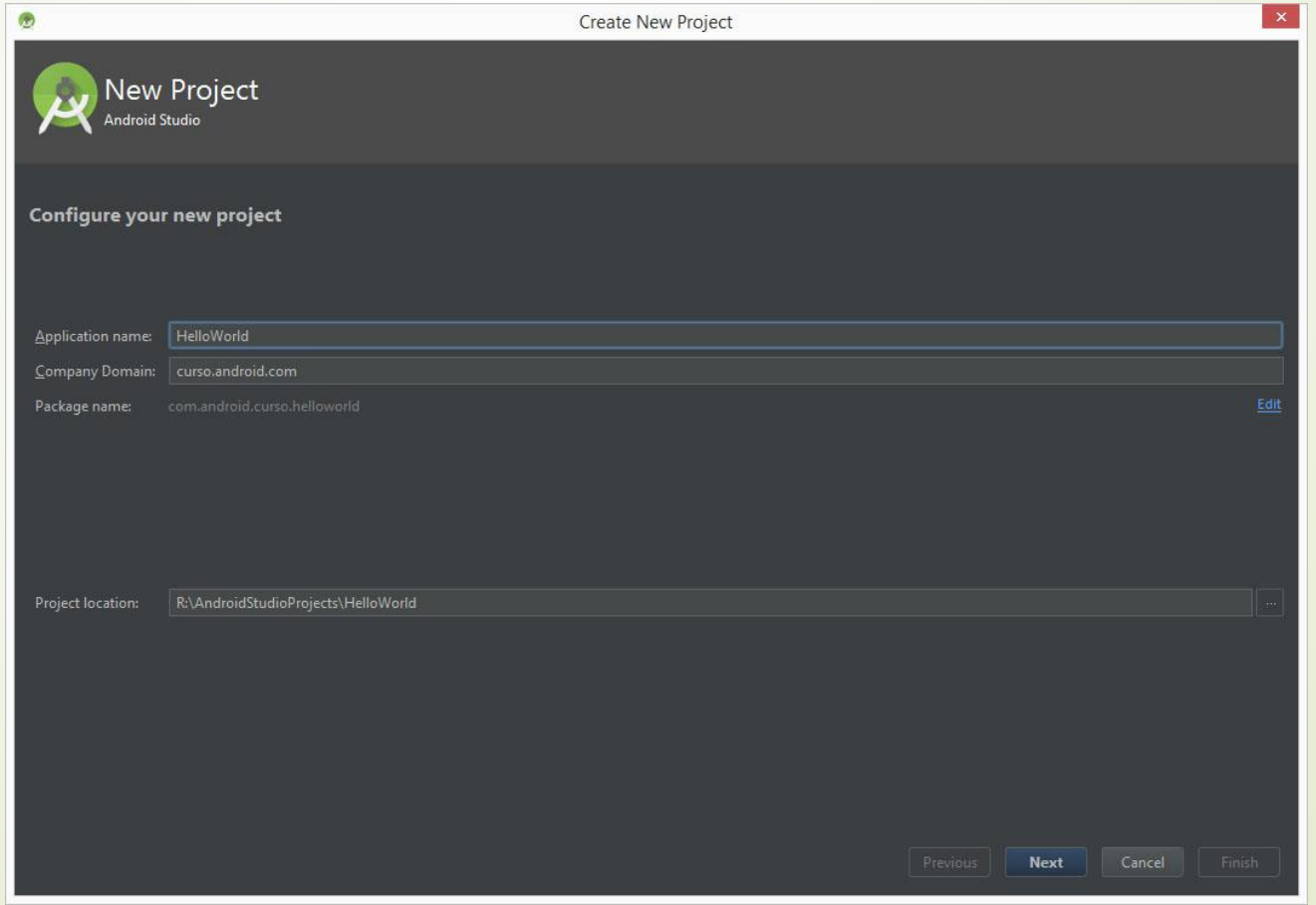


Android Studio

- La primera vista que nos muestra el IDE Android Studio está compuesta por dos partes:
  - A la izquierda nos parecerá nuestro listado de proyectos.
  - A la derecha tendremos distintas opciones
    - Crear un nuevo proyecto.
    - Abrir un proyecto.
    - Importar proyectos desde un control de versiones como Git o Subversion.
    - Importar un proyecto (Eclipse, Gradle).
    - Importar ejemplos de Android.
    - Configuración.
    - Ayuda
- Si la última vez que trabajamos con Android Studio no seleccionamos la opción de cerrar proyecto, nos abrirá el proyecto directamente en lugar de mostrarnos esta vista.

Crear un nuevo proyecto

- Seleccionamos crear un proyecto nuevo:

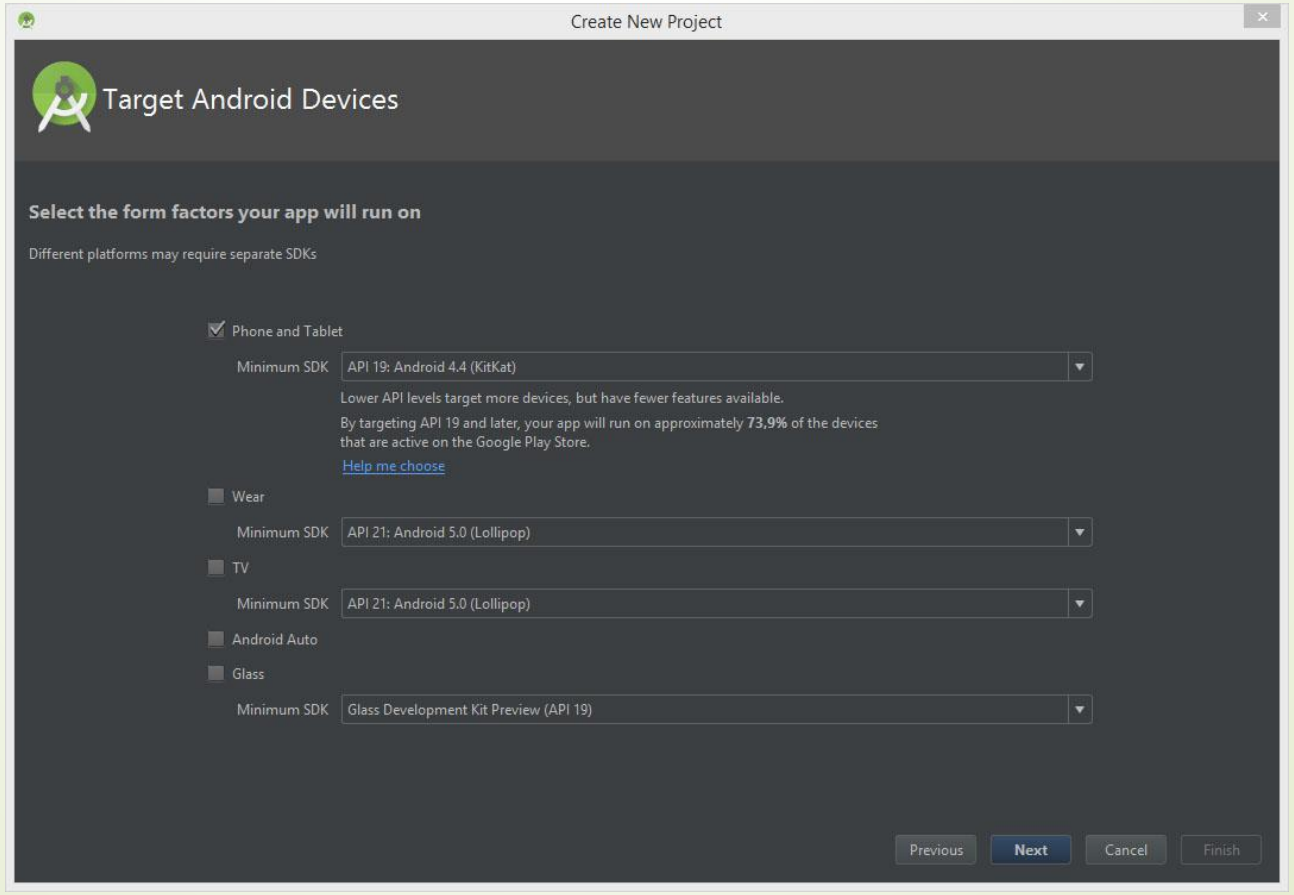


Crear un nuevo proyecto

- Lo primero que nos pregunta al crear un nuevo proyecto es:
  - Nombre de la aplicación.
  - Dominio de la compañía.
  - Package name (Se crea automáticamente a partir del nombre de aplicación y del dominio de la compañía. Se puede editar).
  - La ruta donde se guardara el proyecto.

Crear un nuevo proyecto

- Selección de las plataformas:

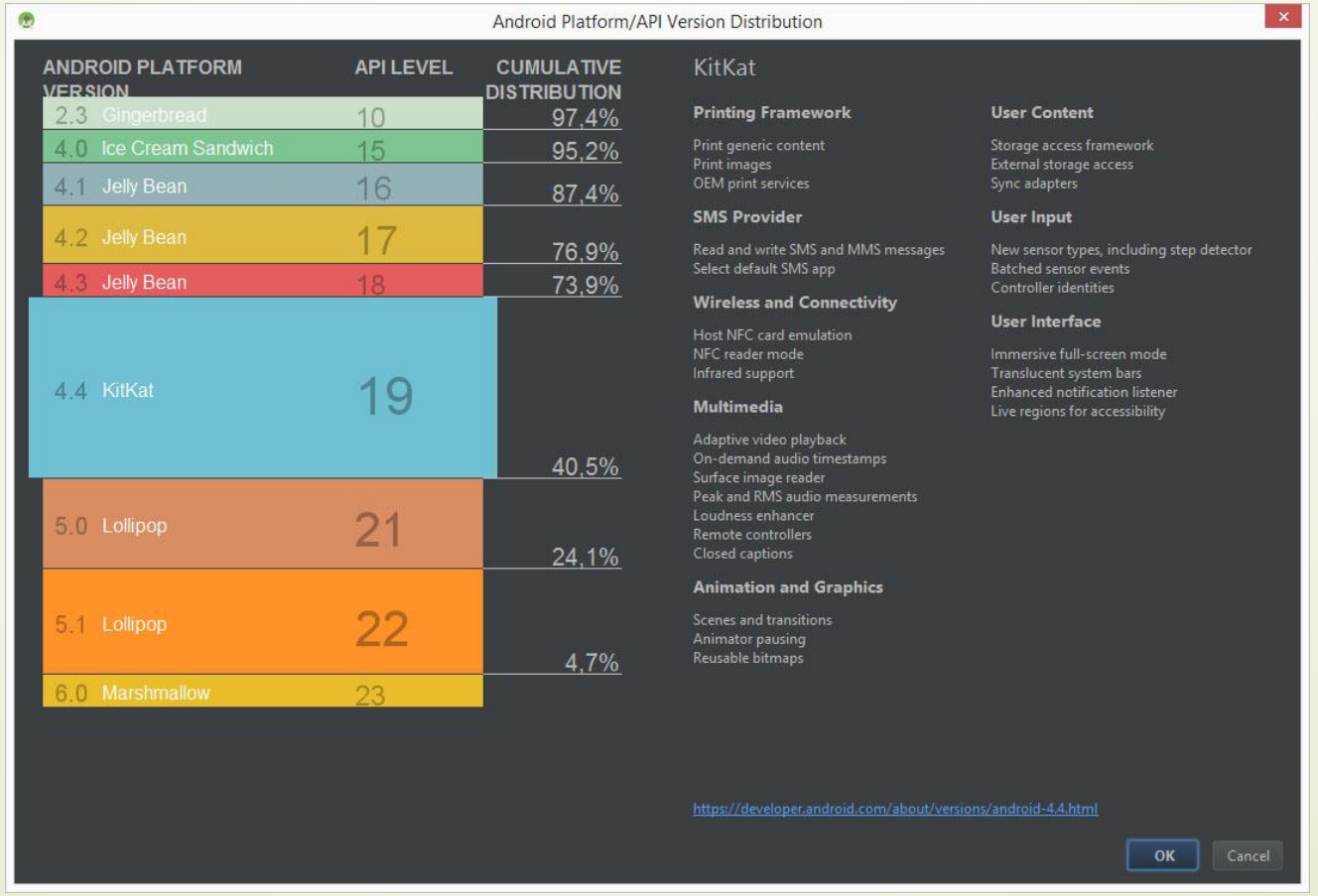


Crear un nuevo proyecto

- Aquí podremos configurar las plataformas y APIs que utilizara nuestra aplicación.
- Plataformas:
  - Teléfonos y Tablets
  - Wear
  - TV
  - Android auto
  - Google Glass.
- SDK mínimo determinara las versiones de Android que necesitaran como mínimo los dispositivos para poder ejecutar la aplicación

Crear un nuevo proyecto

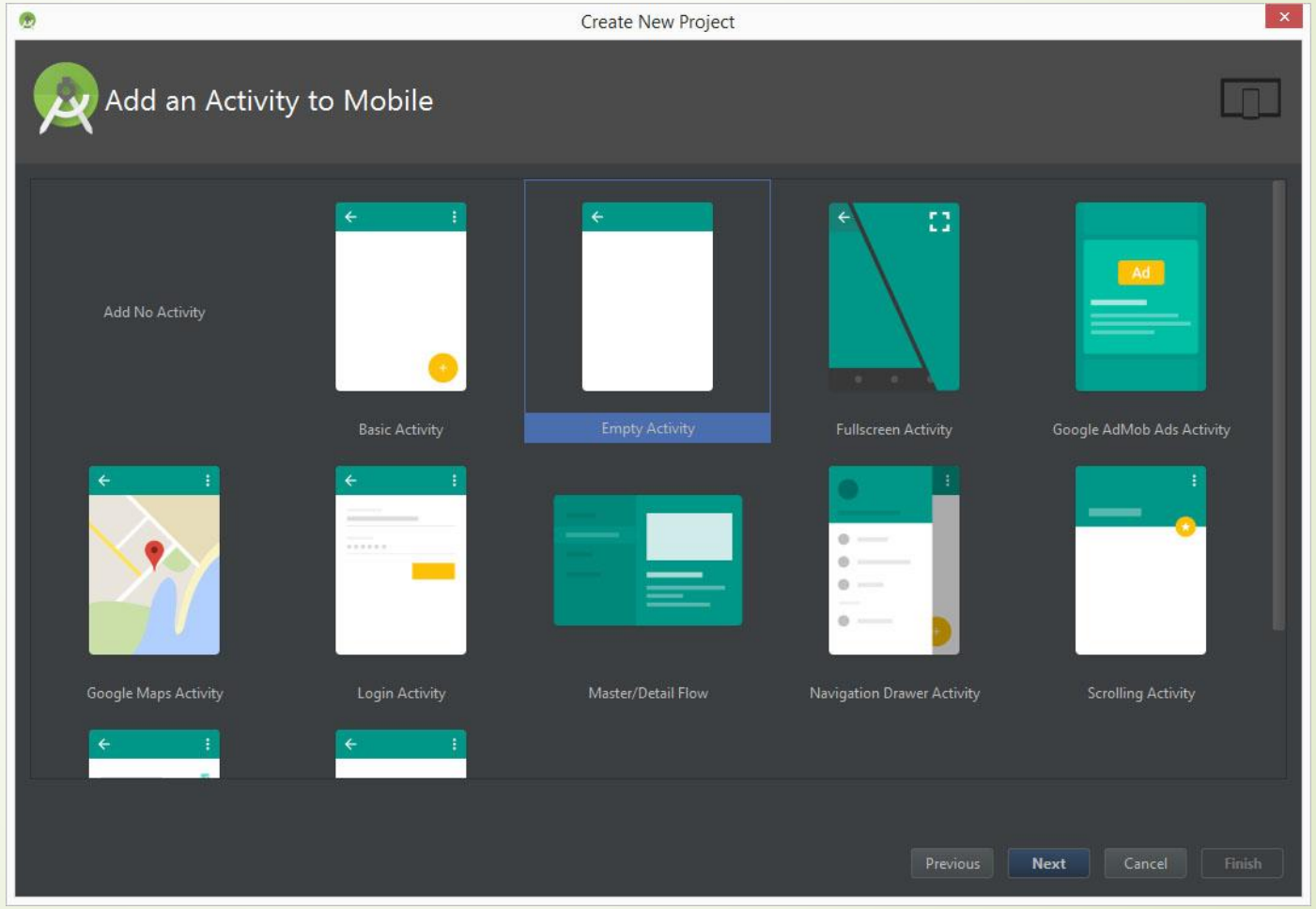
- Help me choose



Crear un nuevo proyecto

- Help me choose
- Nos mostrara un porcentaje de dispositivos que ejecutaran actualmente cada versión de Android.
- También podemos ver las novedades de cada versión.

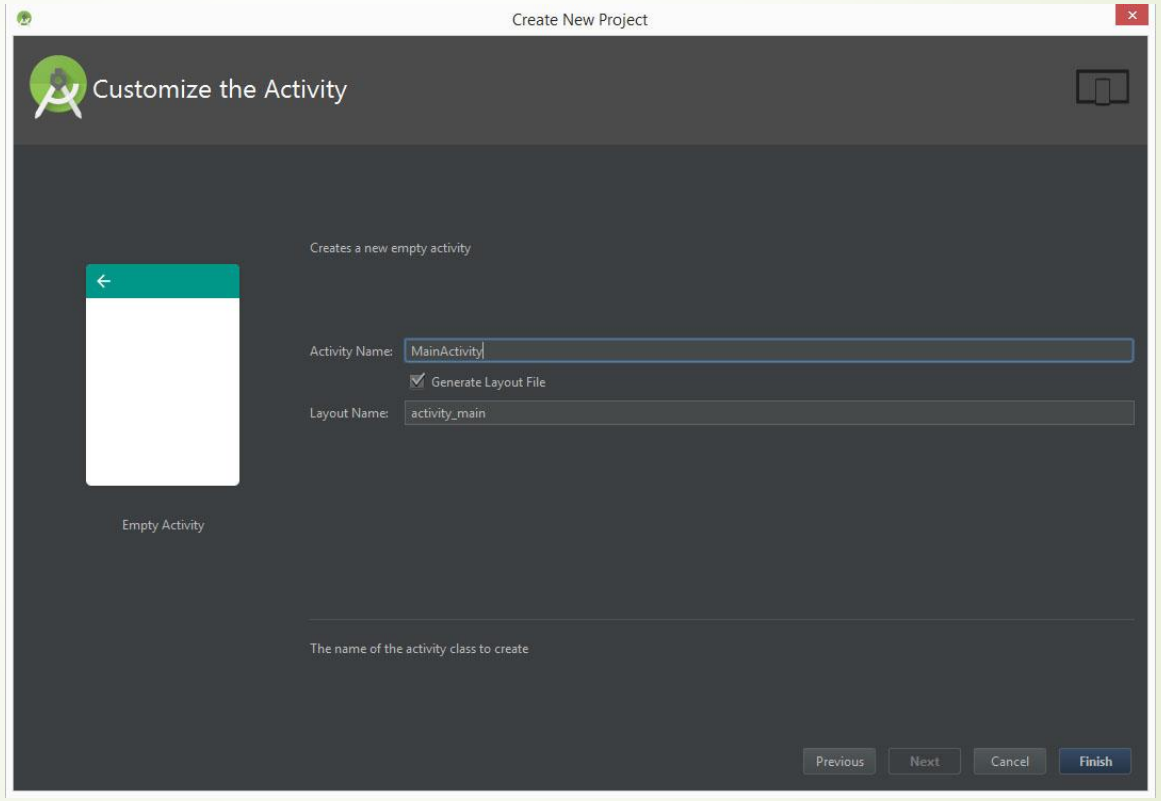
Crear un nuevo proyecto



Crear un nuevo proyecto

- En la siguiente pantalla tendremos que elegir el tipo de actividad principal.
- Tipos:
  - Basic Activity
  - Empty Activity
  - Fullscreen Activity
  - Google AdMob Ads Activity
  - Google Maps Activity
  - Login Activity
  - Maste/Detail Flow
  - Navigation Drawer Activity
  - Scrolling Activity
  - Setting Activity
  - Tabbed Activity

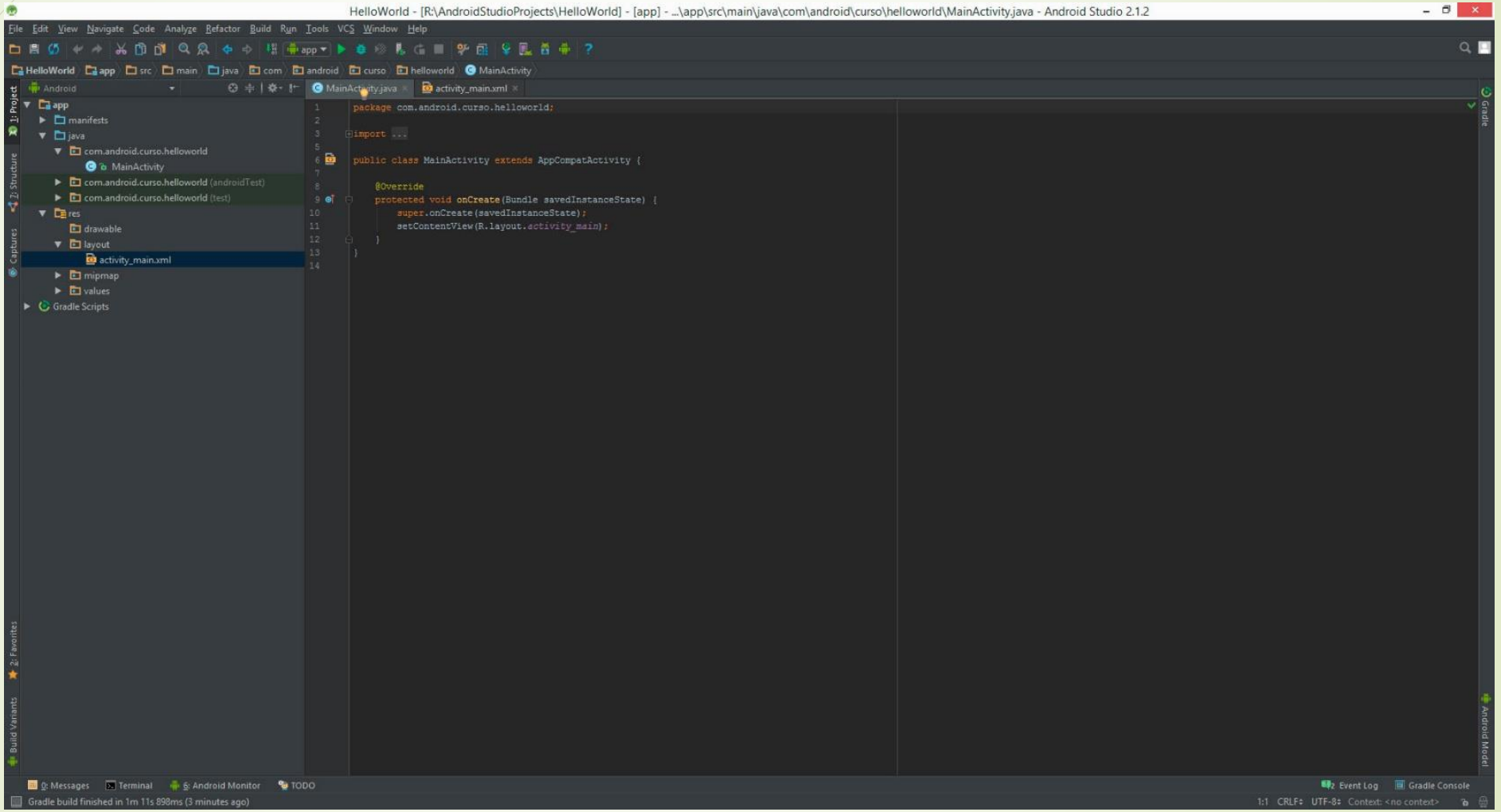
Crear un nuevo proyecto



Crear un nuevo proyecto

- En la última pantalla indicaremos el nombre de la clase java y del layout de la activity.
- Si el tipo de actividad contiene fragments podremos indicar su nombre también.
- Cuando todo este configurado pulsamos Finish para terminar, Android Studio creará el proyecto con los elementos básicos necesarios.

Crear un nuevo proyecto

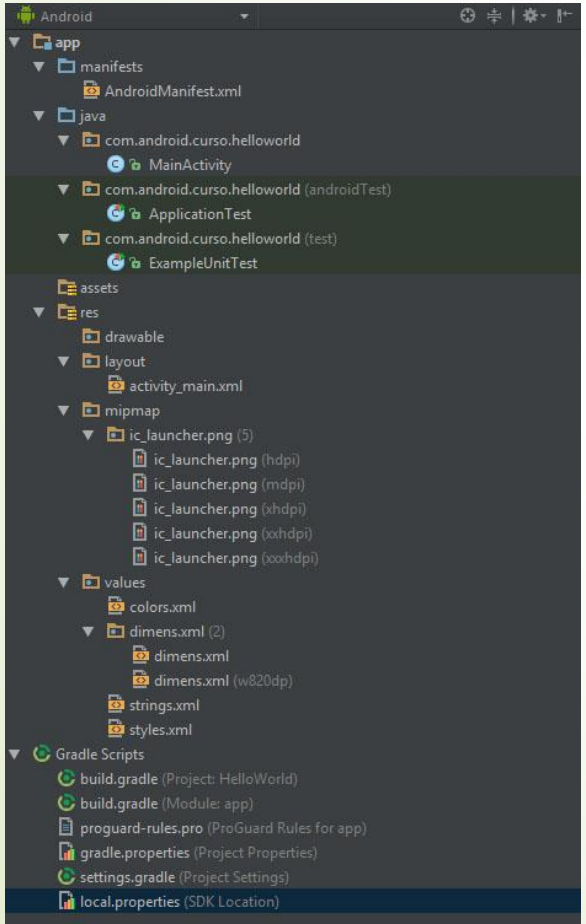


Android Studio

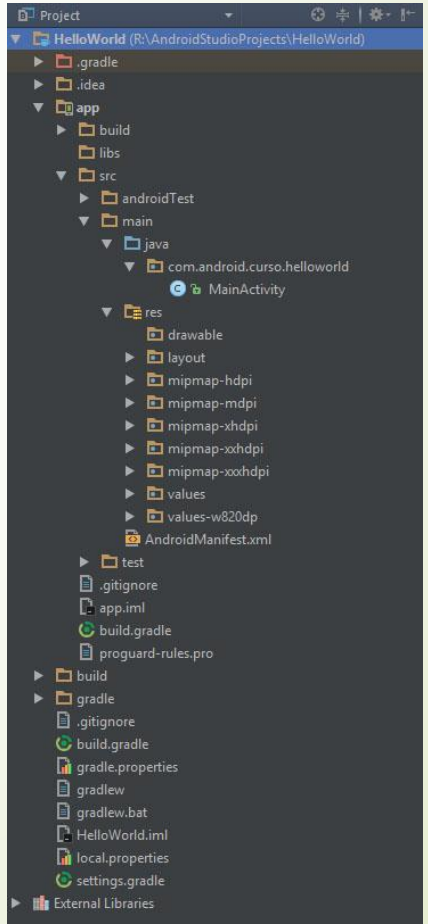
- En la parte izquierda de nuestro IDE veremos los elementos que tiene nuestro proyecto actualmente.
- En la parte de arriba de esta estructura de elementos podremos seleccionar la forma en la que se nos están mostrando estos elementos. Por defecto se muestra con la forma de Android.

Android Studio

Android



Project

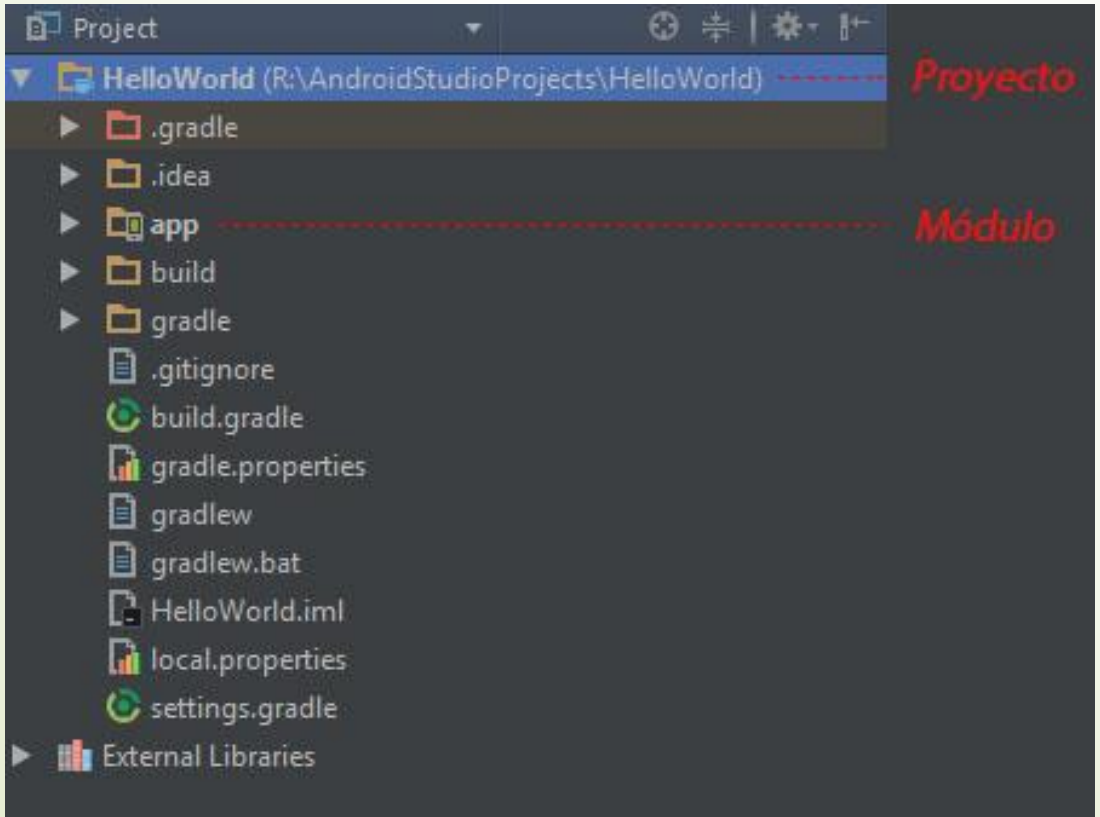


Android Studio

- Los conceptos principales son:
  - Proyecto: Es única y engloba a todos los demás elementos. Dentro de un proyecto podemos incluir varios módulos.
  - Módulo: Pueden representar aplicaciones distintas, versiones diferentes de una misma aplicación, o distintos componentes de un sistema (aplicación móvil, aplicación servidor, librerías,….). En la mayoría de los casos únicamente se tendrá un módulo.

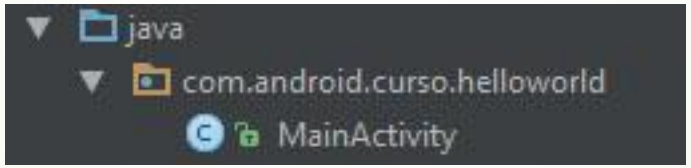
Android Studio

- En el ejemplo anterior el proyecto es HelloWorld y contiene el módulo app el cual contendrá todo el software de la aplicación.



Android Studio

- Contenido principal:
  - Carpeta/app/src/main/java: Esta carpeta contendrá todo el código fuente de la aplicación, clases auxiliares, etc. Inicialmente, Android Studio creará por nosotros el código básico de la pantalla (actividad o activity) principal de la aplicación, que en el ejemplo es MainActivity, y siempre bajo la estructura del paquete java definido durante la creación del proyecto.



Android Studio

- Contenido principal:
  - Carpeta /app/src/main/res/: Contiene todos los ficheros de recursos necesarios para el proyecto: imágenes, layouts, cadenas de texto, etc. Los diferentes tipos de recursos se pueden distribuir entre las siguiente subcarpetas.
    - /res/drawable/: Contiene las imágenes y otros elementos gráficos usados por la aplicación. Para poder definir diferentes recursos dependiendo de la resolución y densidad de la pantalla del dispositivo se suele dividir en varias subcarpetas:
      - /drawable (recursos independientes del la densidad).
      - /drawable –ldp (densidad baja).
      - /drawable –mdpi (densidad media).
      - /drawable –hdpi (densidad alta).
      - /drawable –xhdpi (densidad muy alta).
      - /drawable –xxhdpi (densidad muy muy alta).

Android Studio

- Contenido principal:
  - Carpeta /app/src/main/res/:
    - /res/mipmap/: Contiene los iconos de lanzamiento de la aplicación (el icono que aparecerá en el menú de aplicaciones del dispositivo) para las distintas densidades de pantalla existentes. Al igual que en el caso de las carpetas /drawable, se dividirá en varias subcarpetas dependiendo de la densidad de pantalla:
      - /mipmap-mdpi
      - /mipmap-hdpi
      - /mipmap-xhdpi
      - …

Android Studio

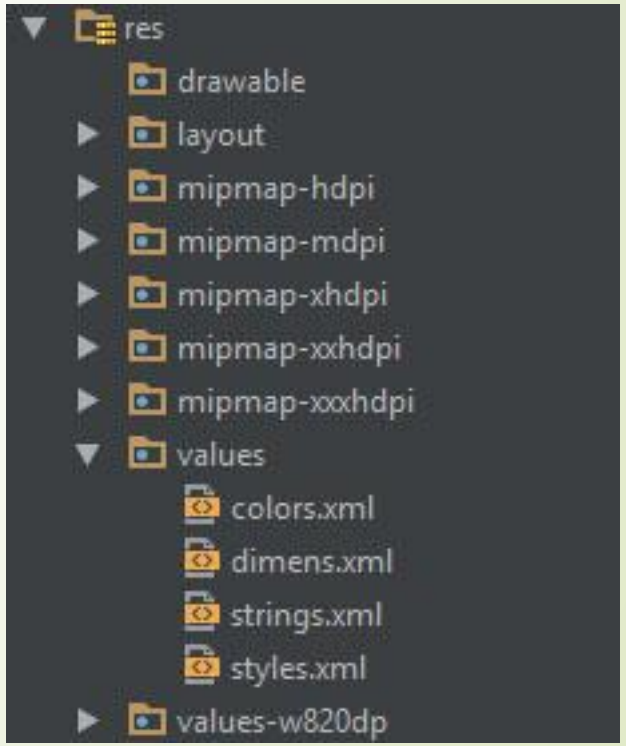
- Contenido principal:
  - Carpeta /app/src/main/res/:
    - /res/layout/: Contiene los ficheros de definición XML de las diferentes pantallas de la interfaz gráfica. Para definir distintos layouts dependiendo de la orientación se pueda dividir también en subcarpetas:
      - /layout (vertical)
      - /layout-land (horizontal)
    - /res/anim/: Contiene la definición de las animaciones utilizadas por la aplicación.
    - /res/animator/: Contiene la definición de las animaciones utilizadas por la aplicación.
    - /res/color/: Contiene ficheros XML de definición de listas de colores según estado.
    - /res/menu/: Contiene la definición de las menús de la aplicación.

Android Studio

- Contenido principal:
  - Carpeta /app/src/main/res/:
    - /res/xml/: Contiene otros ficheros XML de datos utilizados por la aplicación.
    - /res/raw/: Contiene recursos adicionales, normalmente en formato distinto a XML, que no se incluyan en el resto de carpetas de recursos.
    - /res/values/: Contiene otros ficheros XML de recursos de la aplicación, como por ejemplo cadenas de texto (string.xml), estilos (styles.xml), colores (colors.xml), arrays de valores (arrays.xml), tamaños (dimens.xml), etc.

Android Studio

- Contenido principal:
  - Carpeta /app/src/main/res/:
  - No todas estas carpetas tiene que aparecer en cada proyecto, tan sólo las que se necesiten



Android Studio

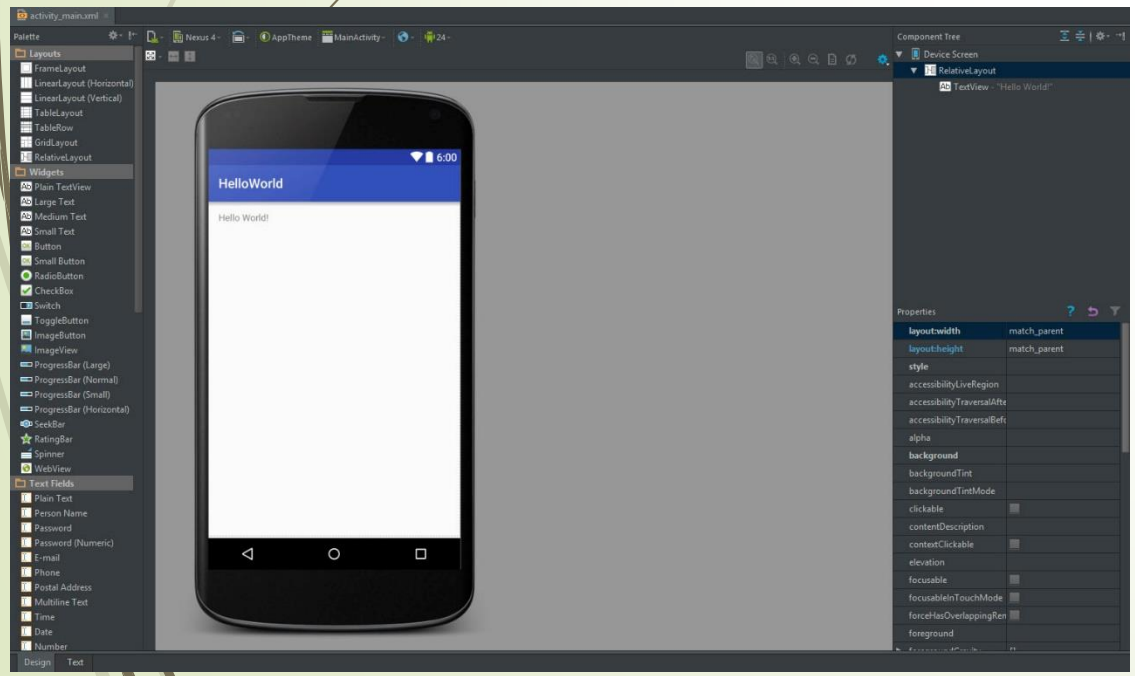
- Contenido principal:
  - Carpeta /app/src/main/res/:
  - Existen algunas carpetas en cuyo nombre se incluye un sufijo adicional, “values-w820dp”. Estos, y otros sufijos, se emplean para definir recursos independientes para determinados dispositivos según sus características. De esta forma, los recursos incluidos en la carpeta “values-w82dp” se aplicarían sólo a pantallas con más de 820dp de ancho, o los incluidos en una carpeta llamada “values-v11” se aplicarían tan sólo a dispositivos cuya versión de Android sea la 3.0 (API 11) o superior. Existen más sufijos además de “-w” y “-v” la lista completa en https://developer.android.com/guide/topics/resources/providing-resources.html

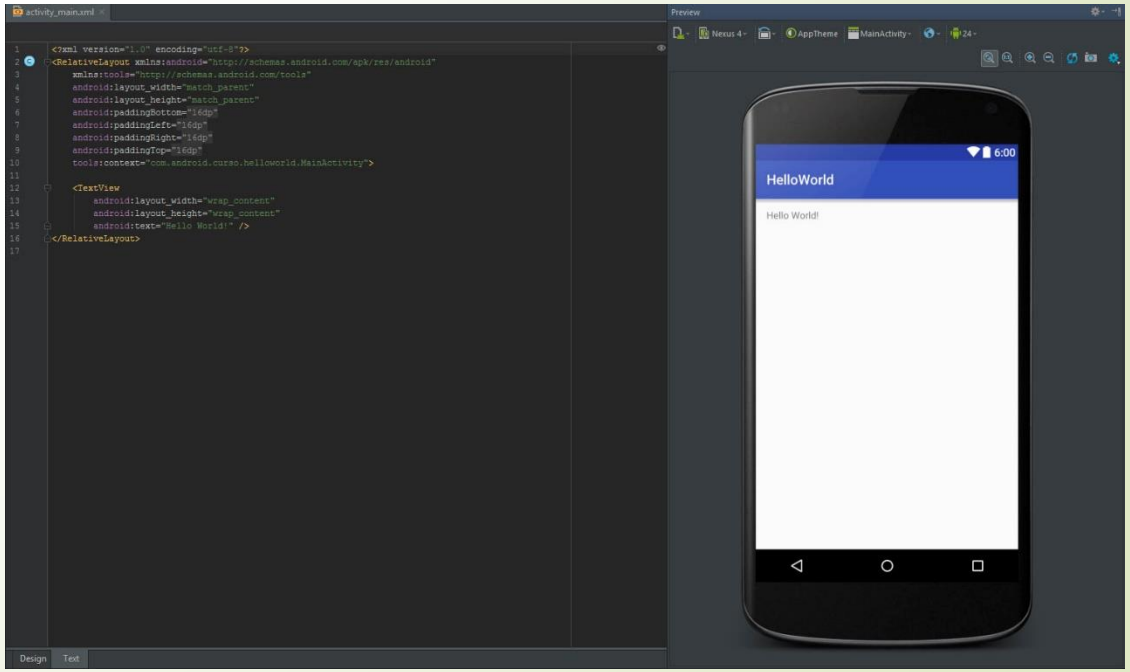
Android Studio

- Contenido principal:
  - Carpeta /app/src/main/res/:
    - Layouts: Los layouts contiene la definición de la interfaz gráfica de las pantallas de la aplicación. Si abrimos cualquier layout nos parecerá el editor gráfico. En las pestañas inferiores podremos cambiar entre el editor gráfico (tipo arrastrar-y-soltar) y el editor XML.

Android Studio

- Contenido principal:
  - Carpeta /app/src/main/res/:
    - Layouts:





Android Studio

- Contenido principal:
  - Fichero /app/src/main/AndroidManifest.xml
    - Contiene la definición en XML de muchos de los aspectos principales de la aplicación, como por ejemplo su identificación (nombre, icono,…), sus componentes (pantallas, servicios,…), o los permisos necesarios para su ejecución.
  - Fichero /app/build.gradle
    - Contiene información necesaria para la compilación del proyecto, (versión del SDK de Android, versión mínima de Android que soportará la aplicación, librerías externas, etc). En un proyecto pueden existir varios ficheros build.gradle, para definir determinados parámetros a distintos niveles. En nuestro proyecto podemos ver que existe un fichero build.gradke a nivel de proyecto, y otro a nivel de módulo dentro de la carpeta /app. El primero definirá parámetros globales a todos los módulos del proyecto, y el segundo sólo tendrá efecto para cada módulo en particular.

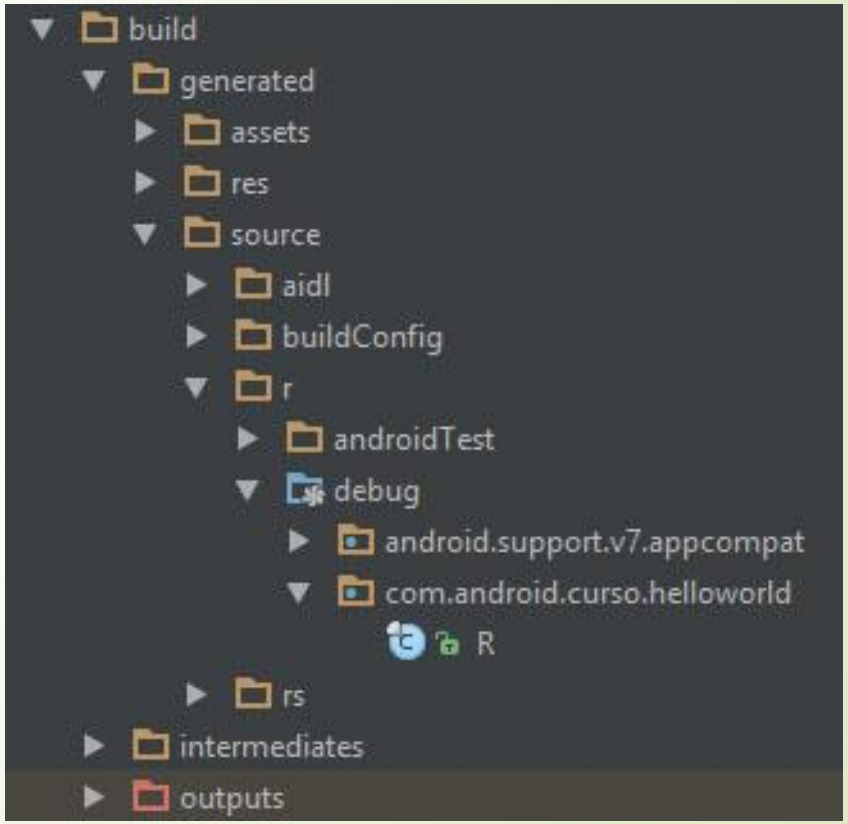
Android Studio

- Contenido principal:
  - Carpeta /app/libs:

    Puede contener las librerías java externas (ficheros .jar) que utilice nuestra aplicación. Normalmente no incluiremos directamente aquí ninguna librería, sino que haremos referencia a ellas en el fichero build.gradle, de forma que entren en el proceso de compilación de nuestra aplicación.
  - Carpeta /app/build/:
    - Contiene una serie de elementos de código generados automáticamente al compilar el proyecto. Cada vez que compilamos nuestro proyecto, la maquinaria de compilación de Android genera por nosotros una serie de ficheros fuente java dirigidos, entre otras muchas cosas, al control de los recursos de la aplicación. Importante: dado que estos ficheros se generan automáticamente tras cada compilación del proyecto es importante que no se modifiquen manualmente bajo ninguna circunstancia.

Android Studio

- Contenido principal:
  - Carpeta /app/build/: Destacar el fichero R.java. Esta clase contendrá en todo momento una serie de constantes con los identificadores (ID) de todos los recursos de la aplicación en la carpeta /app/src/main/res/, de forma que podamos acceder fácilmente a estos recursos desde nuestro código java a través de dicho dato



Android Studio

- Contenido principal:
  - Carpeta /app/src/main/assets/:
    - Contiene el resto de ficheros auxiliares necesarios para que la aplicación funcione, como los ficheros de configuración, de datos, etc. La diferencia entre los recursos incluidos en la carpeta /res/raw/ y los incluidos en la carpeta /assets/ en que para los primeros se generará un ID en la clase R y se deberá acceder a ellos usando un método de esta clase. Sin embargo para los segundos no se generará un ID y se puede acceder a ellos por su ruta como a cualquier otro fichero del sistema.
  - Carpeta /app/src/androidText/:
    - Contiene los test de nuestra aplicación

Gradle

- ¿Qué es Gradle?
  - Es una herramienta para automatizar la construcción de nuestros proyectos, por ejemplo las tareas de compilación, testing, empaquetado y despliegue de los mismos. Es muy flexible para la configuración, pero además ya tiene preparadas tareas para la mayoría de proyectos por defecto.

Gradle

- Añadir librerías:
  - Dependencias: Para añadir una librería por dependencias lo único que debemos hacer es ir al fichero build.gradle de nuestro módulo y en el apartado de dependencias añadir la libraría.
    - Ejemplo: Librería de soporte de Android.

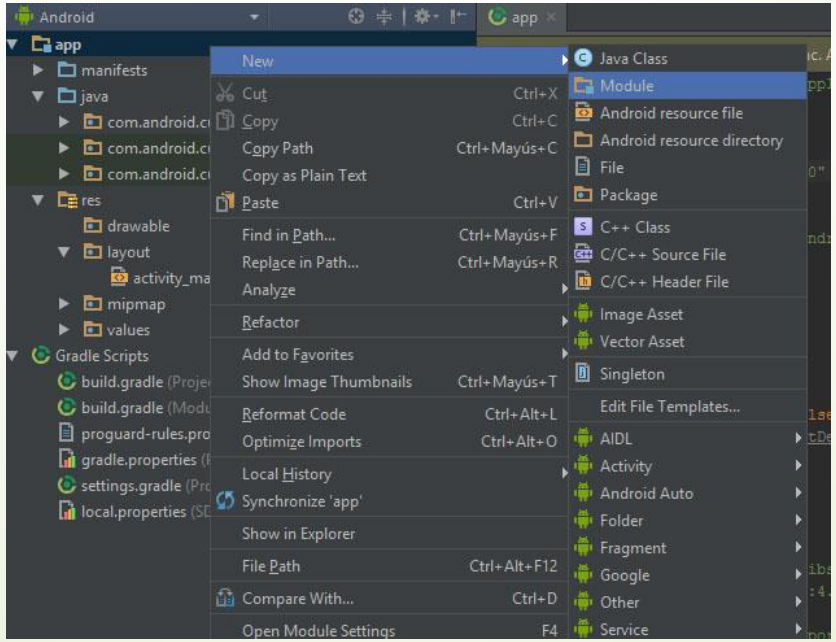
```
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'

    compile 'com.android.support:appcompat-v7:24.0.0'

}
```

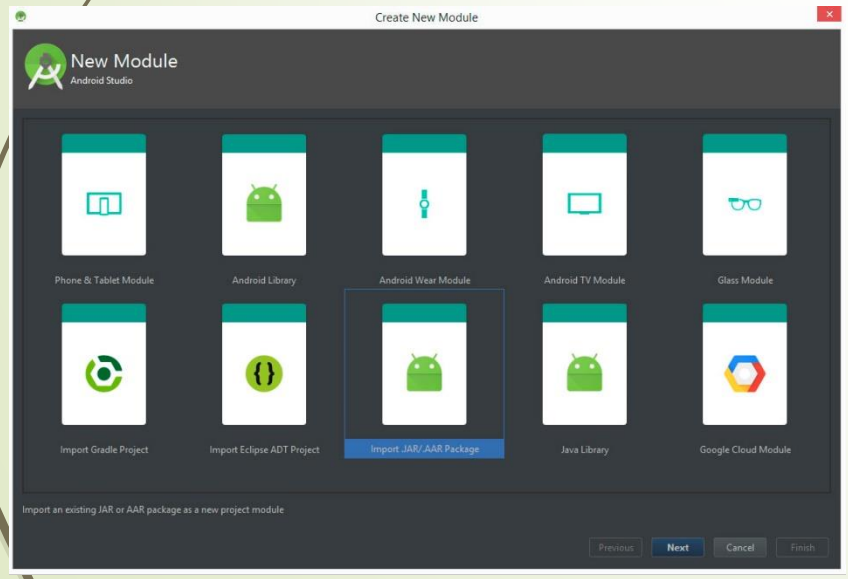
Gradle

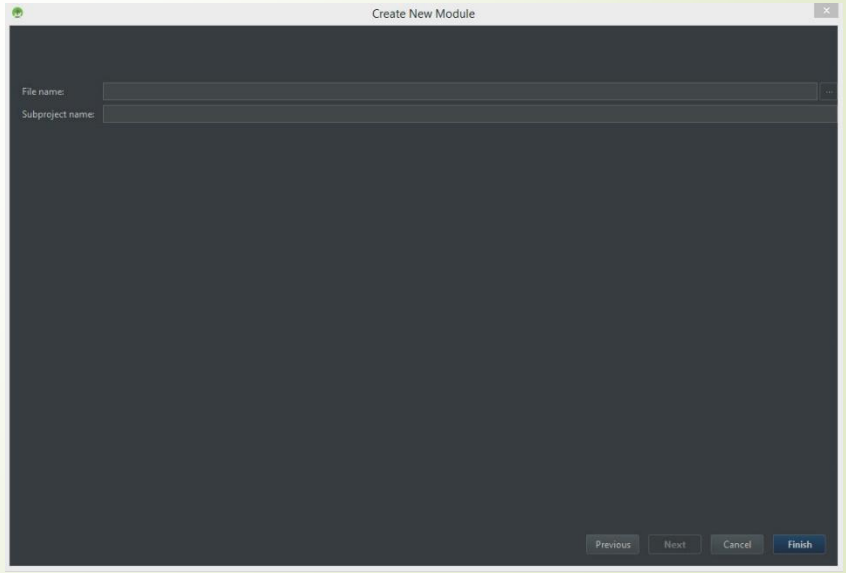
- Añadir librerías:
  - Librerías externas:
    - Descargamos el fichero .jar de la librería.
    - Sobre nuestro proyecto pulsamos el botón secundario y seleccionamos NEW/MODULE



Gradle

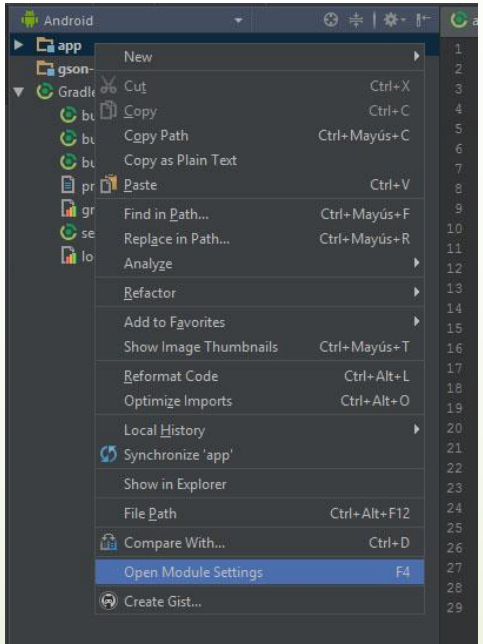
- Añadir librerías:
  - Librerías externas:
    - Nos aparecerá una nueva ventana con varias opciones. Seleccionamos importar paquete JAR o AAR.
    - Al pulsar next nos llevara a otra ventana para seleccionar el fichero.





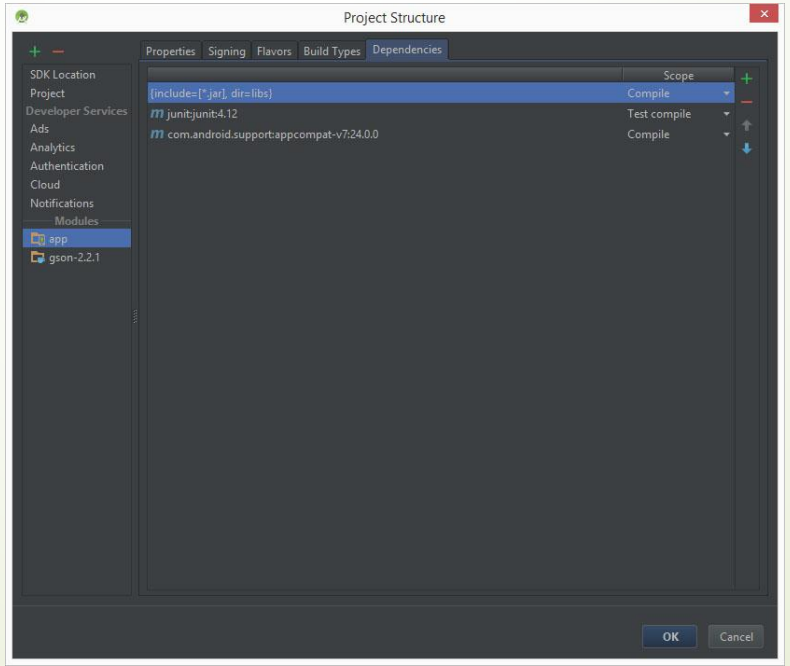
Gradle

- Añadir librerías:
  - Librerías externas:
    - Al finalizar tendremos que esperar a que el proyecto vuelva a construirse.
    - Cuando termina, veremos un nuevo modulo para nuestra librería, volvemos a pulsar el botón secundario sobre el proyecto y seleccionamos ‘Open Module Settings’.



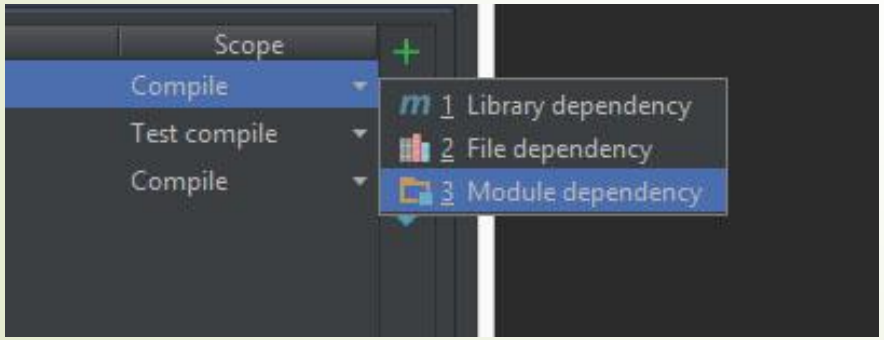
Gradle

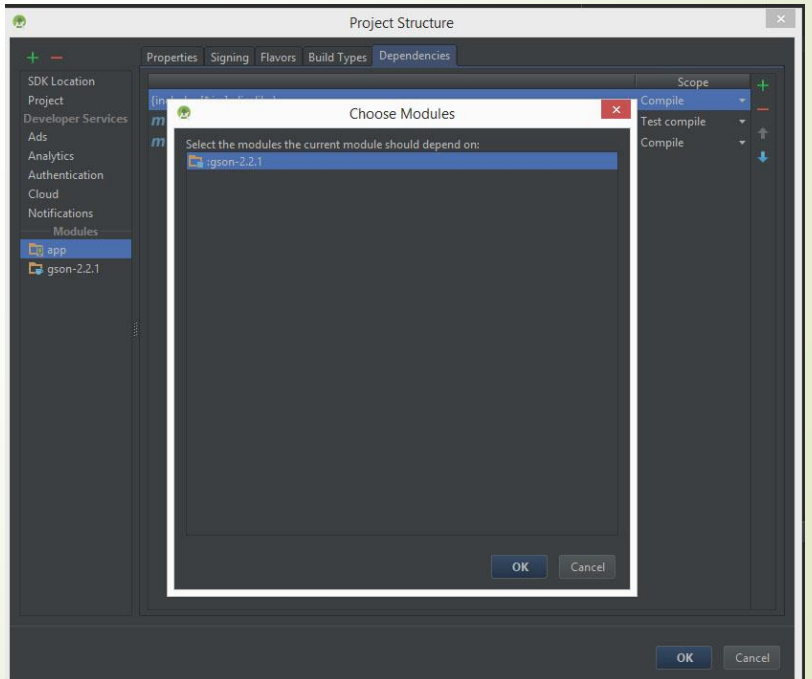
- Añadir librerías:
  - Librerías externas:
    - Se abrirá una nueva ventana y seleccionamos la pestaña ‘Dependencies’.
    - Aquí veremos todas las dependencias de nuestro proyecto.



Gradle

- Añadir librerías:
  - Librerías externas:
    - Pulsaremos sobre el botón de añadir a la derecha de la ventana y seleccionamos ‘Module dependecy’.
    - Nos aparecerá una ventana nueva para seleccionar el modulo que queremos añadir. Seleccionamos el modulo añadido anteriormente.





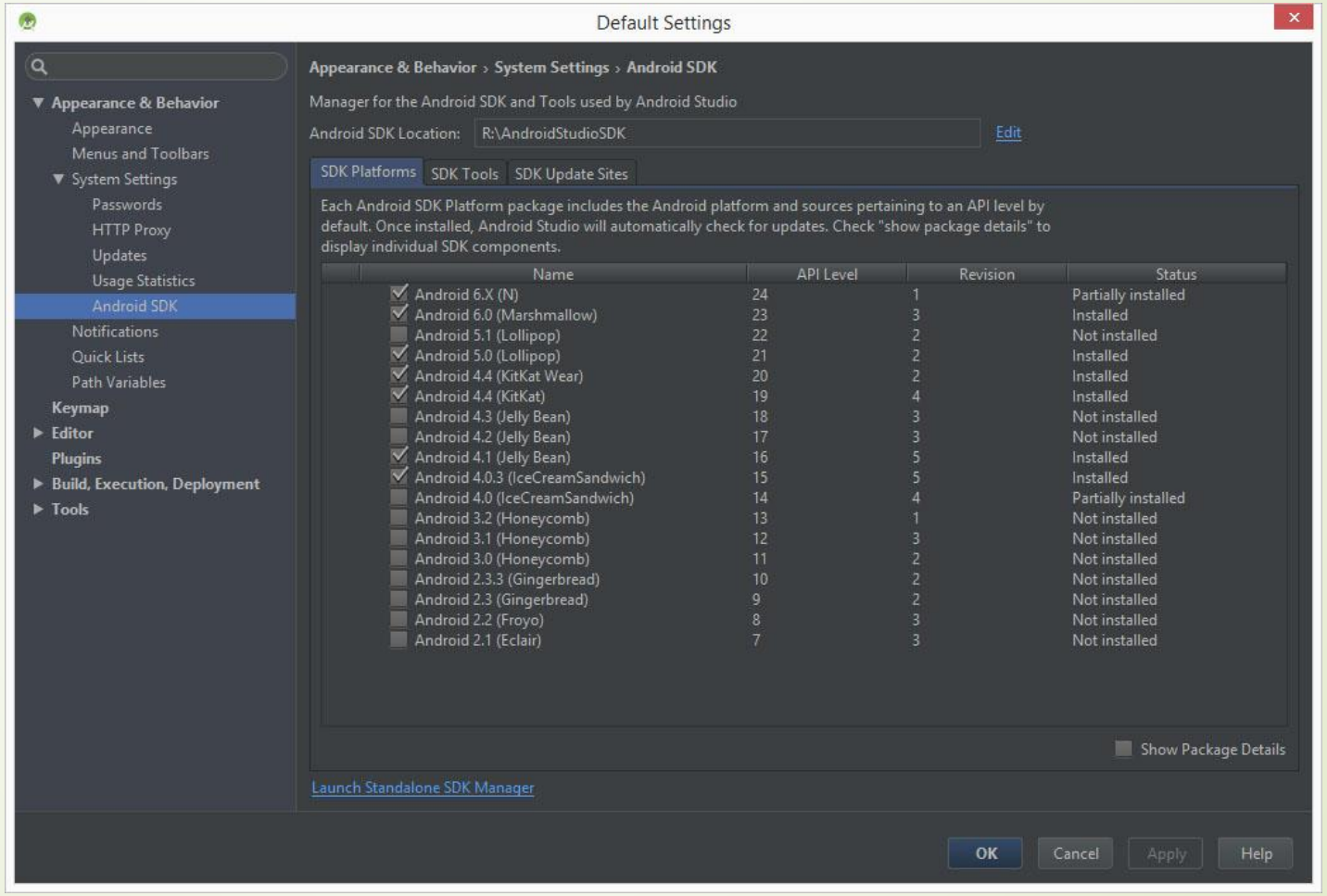
Gradle

- Añadir librerías:
  - Librerías externas:
    - Pulsaremos Ok para terminar y Android Studio volverá a construir el proyecto. Cuando termina ya podremos utilizar la Librería añadida.

SDK Android

- SDK (Software Development Kit): Es el kit de desarrollo de software. Contiene todas las herramientas necesarias para desarrollar y ejecutar un emulador del sistema Android.
- El SDK Manager nos permitirá tener actualizadas estas herramientas.

SDK Android

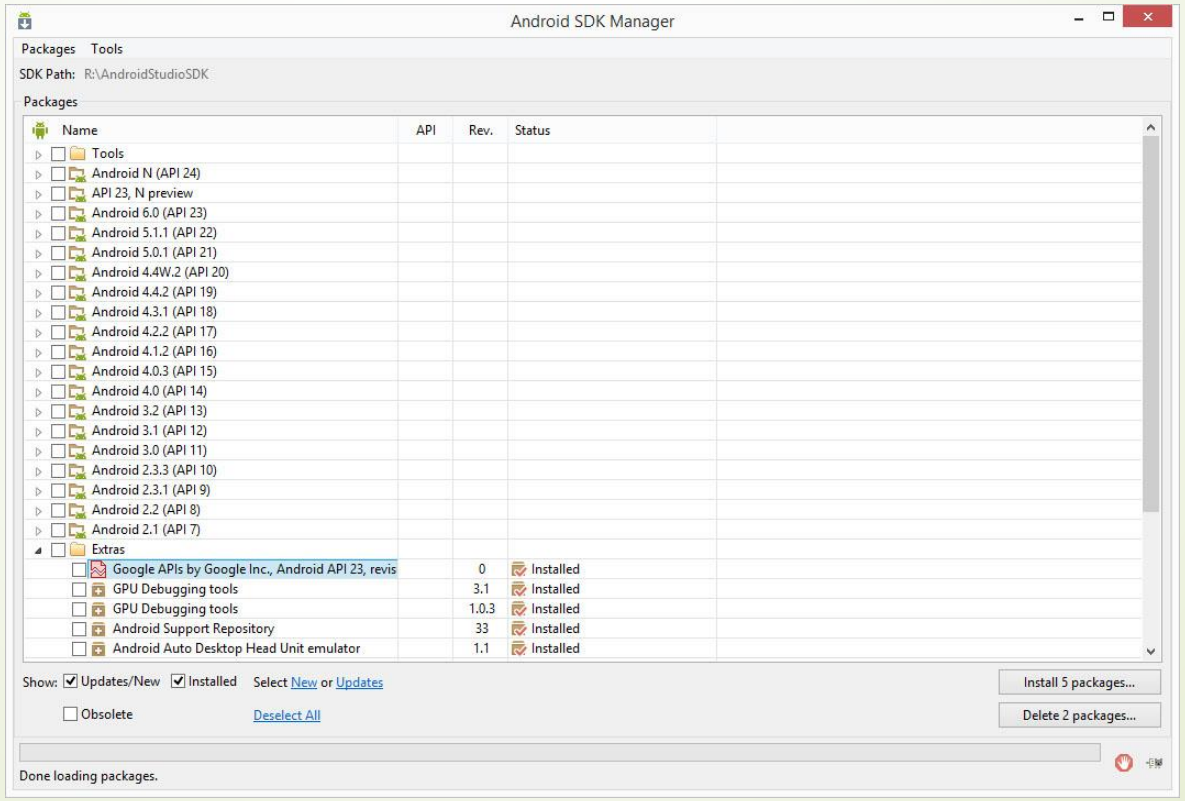

Default Settings
Appearance & Behavior
Appearance
Menus and Toolbars
System Settings
Passwords
HTTP Proxy
Updates
Usage Statistics
Android SDK
Notifications
Quick Lists
Path Variables
Keymap
Editor
Plugins
Build, Execution, Deployment
Tools
Appearance & Behavior › System Settings › Android SDK
Manager for the Android SDK and Tools used by Android Studio
Android SDK Location: R:\AndroidStudioSDK
Edit
SDK Platforms
SDK Tools
SDK Update Sites
Each Android SDK Platform package includes the Android platform and sources pertaining to an API level by default. Once installed, Android Studio will automatically check for updates. Check "show package details" to display individual SDK components.
Name
API Level
Revision
Status
Android 6.X (N) 24 1 Partially installed
Android 6.0 (Marshmallow) 23 3 Installed
Android 5.1 (Lollipop) 22 2 Not installed
Android 5.0 (Lollipop) 21 2 Installed
Android 4.4 (KitKat Wear) 20 2 Installed
Android 4.4 (KitKat) 19 4 Installed
Android 4.3 (Jelly Bean) 18 3 Not installed
Android 4.2 (Jelly Bean) 17 3 Not installed
Android 4.1 (Jelly Bean) 16 5 Installed
Android 4.0.3 (IceCreamSandwich) 15 5 Installed
Android 4.0 (IceCreamSandwich) 14 4 Partially installed
Android 3.2 (Honeycomb) 13 1 Not installed
Android 3.1 (Honeycomb) 12 3 Not installed
Android 3.0 (Honeycomb) 11 2 Not installed
Android 2.3.3 (Gingerbread) 10 2 Not installed
Android 2.3 (Gingerbread) 9 2 Not installed
Android 2.2 (Froyo) 8 3 Not installed
Android 2.1 (Eclair) 7 3 Not installed
Show Package Details
Launch Standalone SDK Manager
OK
Cancel
Apply
Help

SDK Android

- Se nos mostrara una ventana en la que podremos ver lo que tenemos instalado y los que no.
  - SDK Platforms
  - SDK Tools
  - SDK Update Sites
- Si pulsamos sobre Launch Standalone SDK Manager

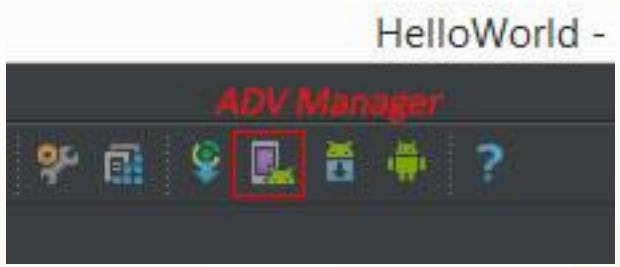
SDK Android

- Esta nueva vista veremos todos los elementos que podemos instalar y los que ya tenemos instalados de una forma más detallada.



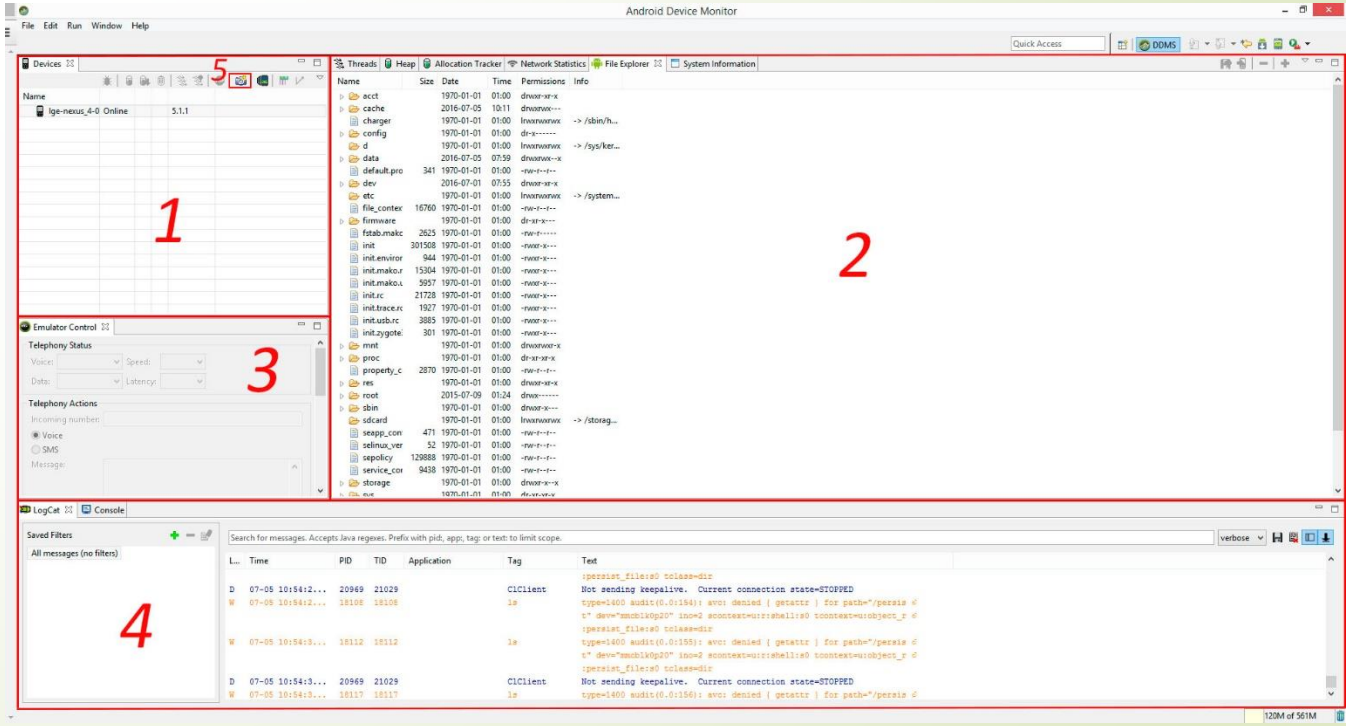
Android device monitor

- En Android device monitor tendremos acceso al DDMS (Dalvic Debug Monitor Service) una utilidad de depuración.
- Para acceder pulsaremos sobre el icono:



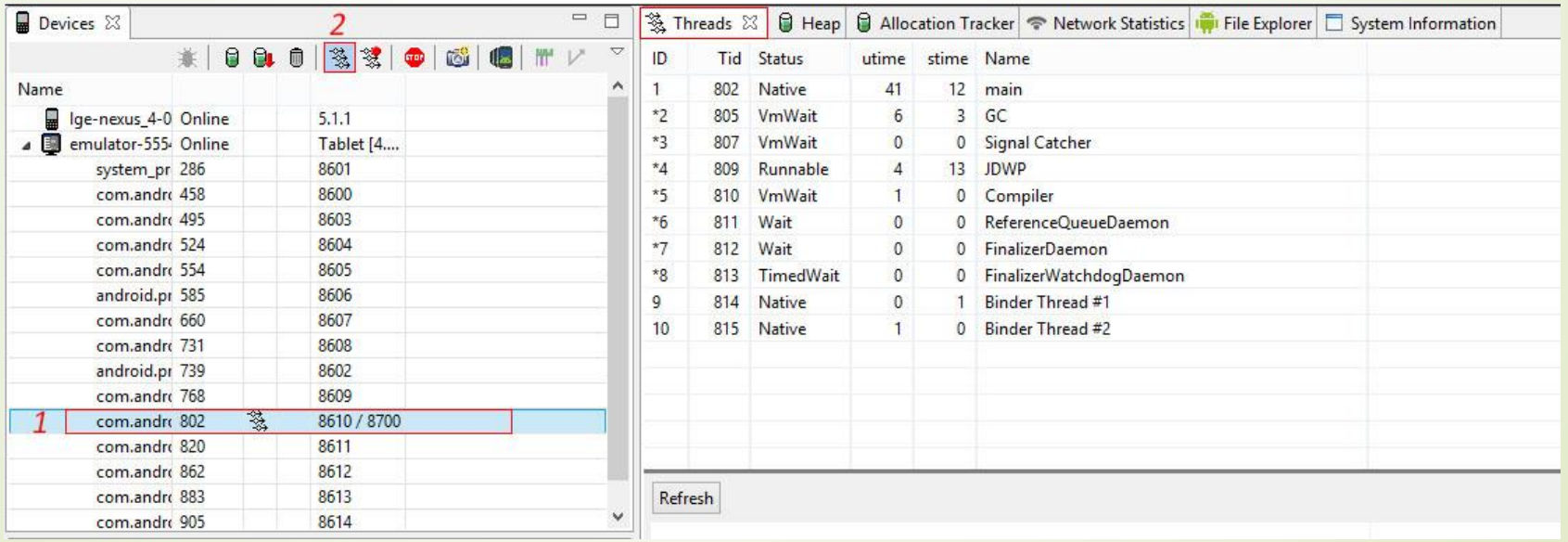
Android device monitor

1. Manejo de Tareas: veremos los emuladores y teléfonos que tengamos conectados y sus instancias.
2. Manejo de archivos
3. Interacción con el emulador
4. Sistemas de Log
5. Capturas de Pantalla



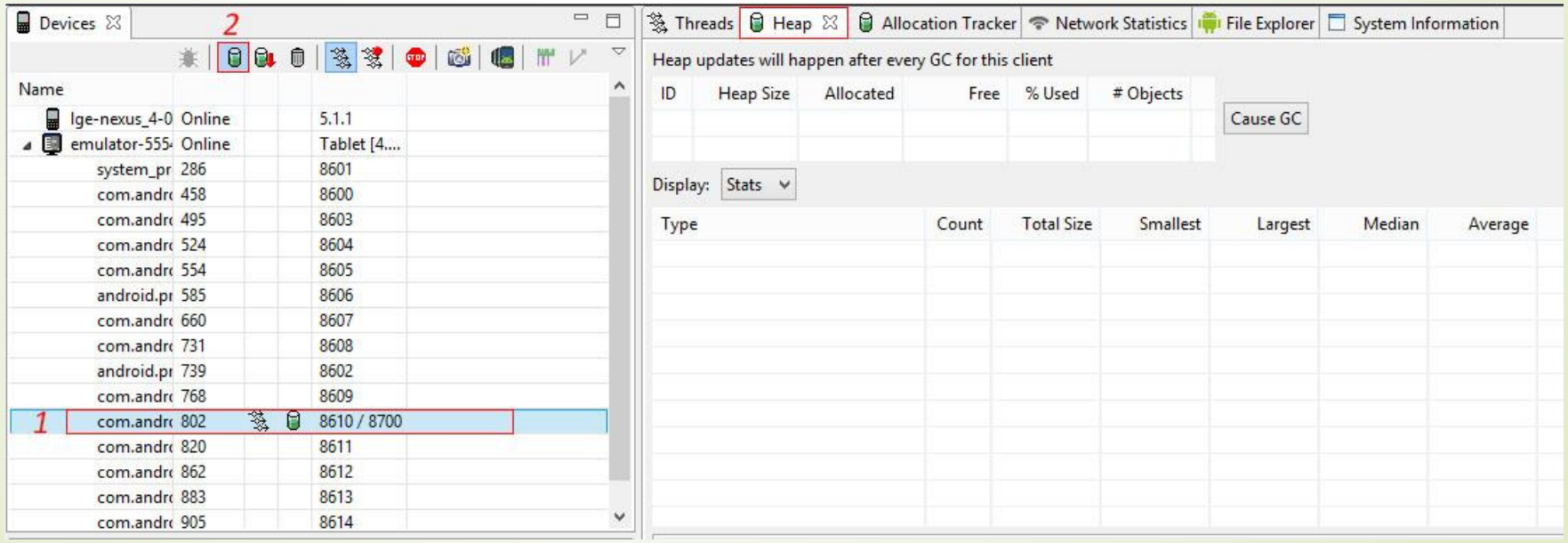
Android device monitor

- Pestañas:
  - Threads: Desde esta pestaña podemos ver los procesos e hilos de ejecución de las instancias individuales del dispositivo o emulador. Para ello seleccionamos el proceso que queramos inspeccionar (1) y pulsamos el botón ‘Update Threats’ (2).



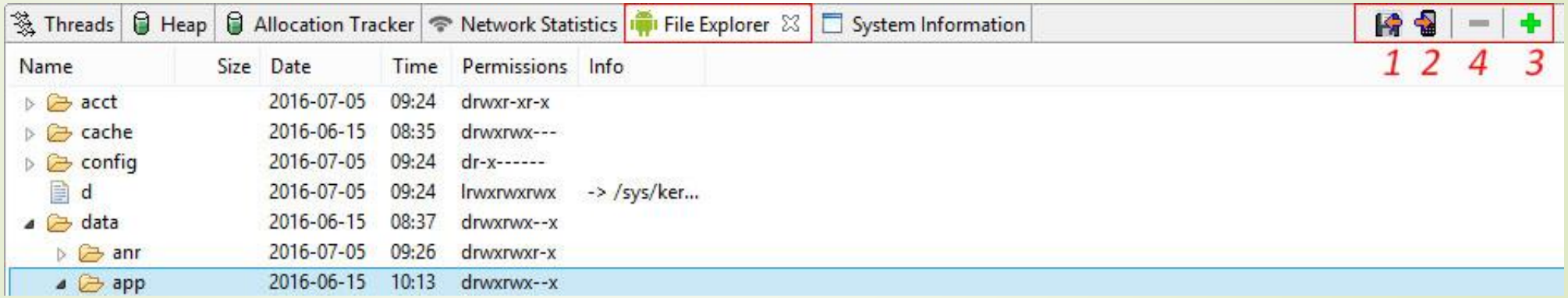
Android device monitor

- Pestañas:
  - Heap: Sirve para ver la pila y las actualizaciones que se vayan haciendo en ella. Si lanzamos un proceso de recolección de basura podemos ver como cambia. Para ello con el proceso seleccionado (1), pulsamos el botón ‘Update head’ (2).



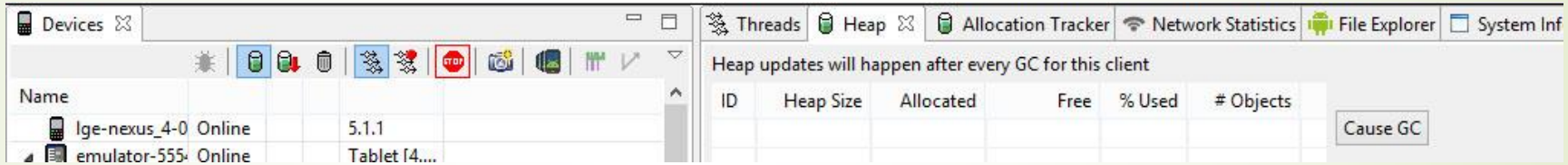
Android device monitor

- Pestañas:
  - File Explorer: Además de navegar por los archivos de nuestro emulador o dispositivo conectado, nos permite transferir archivos entre el dispositivo y nuestra máquina de desarrollo. Para recibir archivos desde el dispositivo usaremos el botón 'Pull a file from the device' (1) y para mandarlos el botón 'Push a file onto the device' (2). Al pulsarlos se nos abrirá el explorador de archivos. También podemos añadir nuevas carpetas con el botón 'New Folder' (3) o borrar archivos con el botón "Delete the selection" (4) o pulsando la tecla [suprimir] en el teclado. En el caso del borrado NO nos pedirá confirmación.

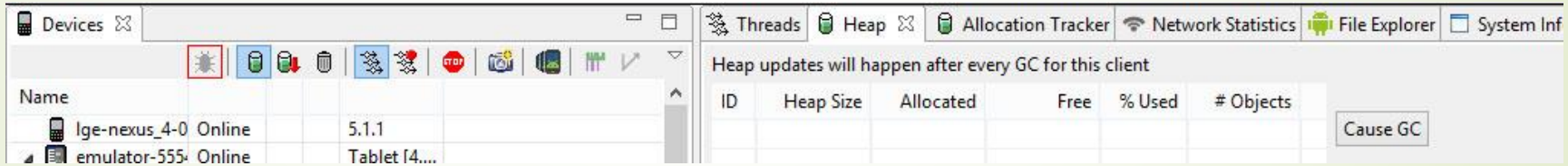


Android device monitor

- Detener un proceso: Pulsar el botón “Stop Process”.



- Depurar un proceso: Pulsar el botón “Debug”.

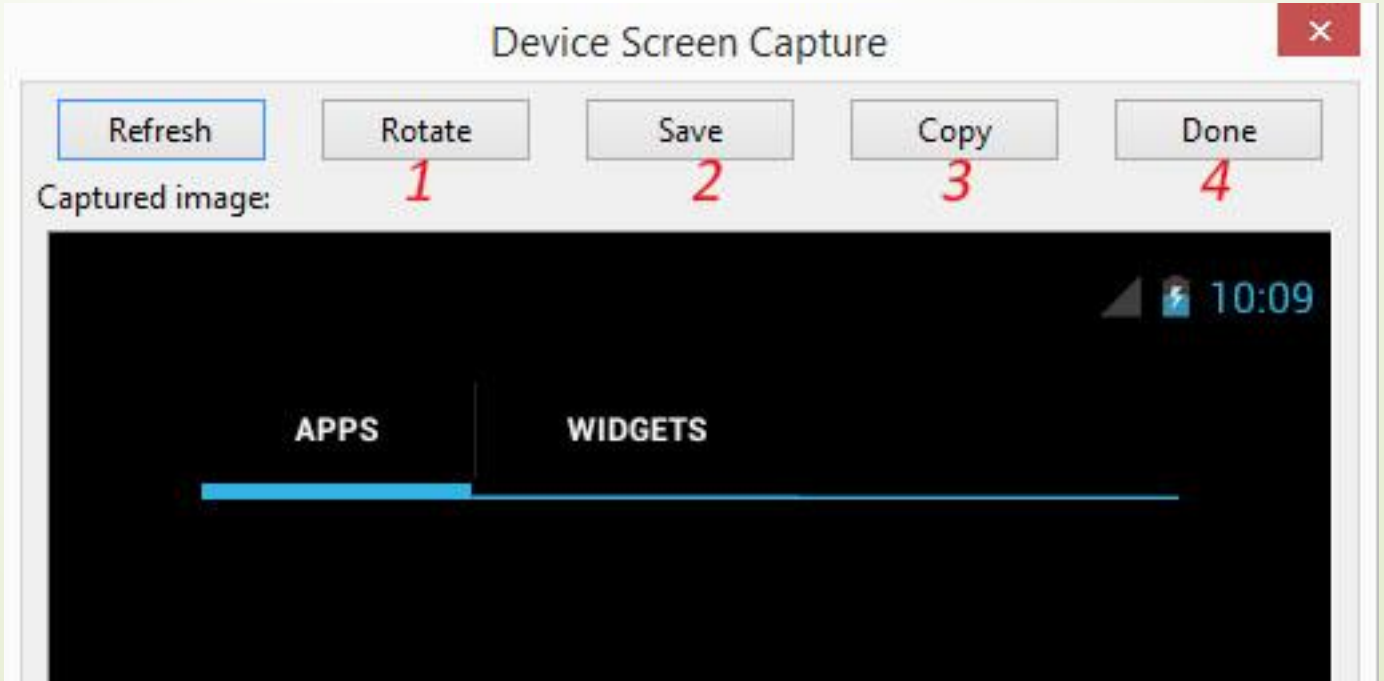


Android device monitor (ya no es compatible, Actualmente debe hacerse desde el emulador)

- Emulator Control:
  - Nos permite interactuar con los emuladores enviando distintos tipos de eventos como llamadas, mensajes SMS o coordenadas de localización. Sólo funciona para emuladores, para teléfonos de hacerse de forma real con su coste incluido.
    - Simular llamada de voz:
      - Escogemos el emulador (1)
      - Introducimos el número de teléfono desde el que llamamos (2)
      - Nos aseguramos de tener marcado ‘voice’ (3)
      - Pulsamos ‘Call’
    - Simular SMS:
      - Igual que la llamada pero en lugar de seleccionar ‘voice’ seleccionamos ‘SMS’.
      - Pulsamos ‘Send’

Android device monitor

- Screen Capture:
  - Nos permite capturar un pantallazo de nuestro emulador o dispositivo conectado. Para ello una vez estemos sobre la pantalla que queremos capturar pulsaremos sobre el icono de ‘Screen capture’ y podremos Rotar la imagen (1), guardarla (2), hacer un copy para pegarla en un documento (3) y salir con el botón ‘done’ (4).



Android device monitor

- Consola de Logs:
  - Es una salida de diagnóstico de funcionamiento de las aplicaciones. Permite filtrar los logs según su importancia y crear filtros de logs personalizados usando etiquetas.