
	RAMA:	Informática	CICLO:	Desenvolvemiento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

Tema 1

Introducción a la plataforma Android. Android Studio

Índice

1.	Introducción	1
2.	Arquitectura de la plataforma	1
3.	IDE: Android Studio	3
3.1.	Crear un proyecto	6
3.2.	Visión general de Android Studio.....	8
4.	SDK Manager	11
5.	El emulador, dispositivos virtuales y la terminal de control	12
5.1.	Aceleración Hardware del emulador:.....	16
6.	Conexión de dispositivos.....	18
7.	ABD	19

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

1. Introducción

En este tema daremos un vistazo genérico a la plataforma Android para conocer su arquitectura y entender cómo se relacionan las distintas capas del software y como, a su vez, está enlazado con el hardware correspondiente.

También introduciremos las herramientas que se usaran en el curso: tanto el IDE como los emuladores de dispositivos además de otras herramientas afines.

Finalmente veremos cómo probar las aplicaciones que realicemos tanto en un dispositivo real, como puede ser un móvil o una tablet, como en el emulador integrado en Android Studio.

Hay muchos tutoriales, blogs, foros y webs en los que recabar información, pero quizá un buen sitio donde empezar es en la propia web de desarrolladores de Android: <https://developer.android.com/index.html> que entre las siguientes secciones podemos encontrar la guía de Android: <https://developer.android.com/guide>.

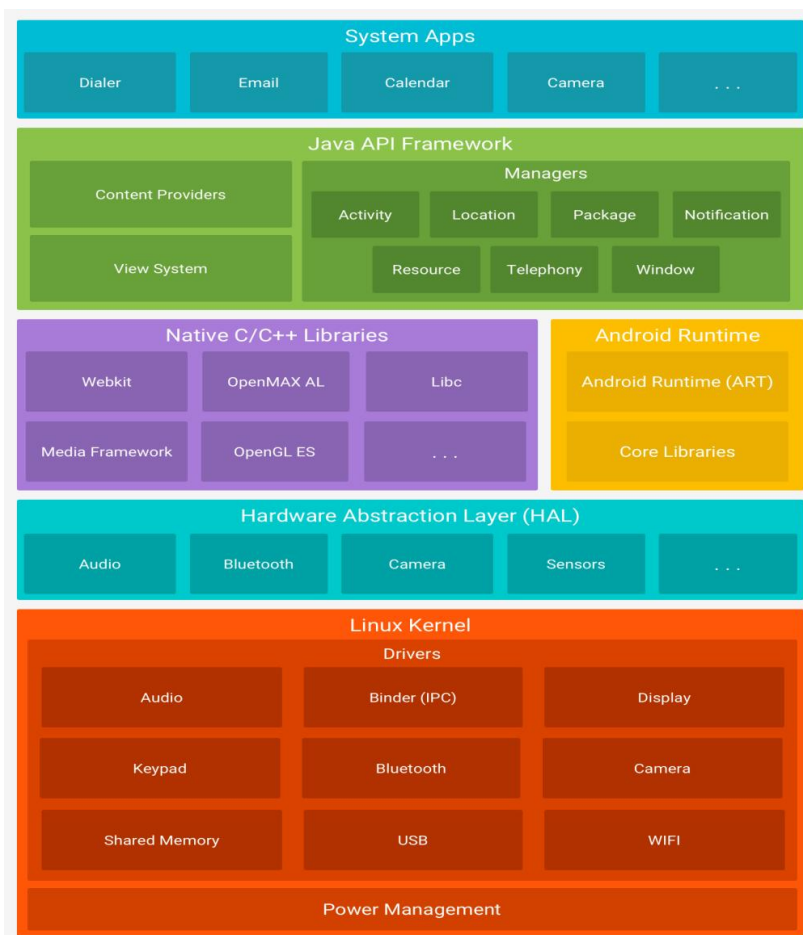
2. Arquitectura de la plataforma¹

Android utiliza una estructura por capas mediante el uso de módulos que se relacionan entre sí y con las capas adyacentes.

Estas capas son:

- Núcleo del sistema operativo (Kernel): Android trabaja con un núcleo Linux con sus módulos típicos de gestión de memoria, CPU, drivers de hardware, pero además, dispone de elementos específicos de Android. Elementos como la gestión de la energía o la interrelación entre procesos son críticos en este sistema.
- Capa de abstracción hardware: proporciona una serie interfaces estándar que permite trabajar con componentes hardware como pueden ser la cámara, los sensores, ...
- Runtime de Android: en esta capa encontramos el JRE (Java Runtime Environment) compuesto por las librerías básicas (Core Libraries) que contienen los paquetes android.* y el entorno de ejecución o máquina virtual según la versión de Android.

Para versiones Android 5.0² (Lollipop, Api level 21) y posteriores se utiliza el entorno de ejecución de aplicaciones ART³ (Android Runtime) mientras que en versiones anteriores se utiliza la máquina virtual DalvikVM⁴.




¹ <https://developer.android.com/guide/platform?hl=es>

² Más información sobre las versiones en: http://es.wikipedia.org/wiki/Anexo:Historial_de_versiones_de_Android

³ [https://es.wikipedia.org/wiki/Android_Runtime_\(ART\)](https://es.wikipedia.org/wiki/Android_Runtime_(ART))

⁴ <https://es.wikipedia.org/wiki/Dalvik>

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

- Bibliotecas nativas: en esta capa se encuentran una serie de librerías programadas que buscan sobre todo optimización de uso de recursos: OpenGL para gráficos, WebKit para renderización Web, SSL para seguridad, ... Utilizadas por los desarrolladores de Android de bajo nivel usando NDK y programando en C/C++.
- Java API Framework: funciones que proporciona la API de *Android* y que se utilizarán, por los desarrolladores de Android, para crear aplicaciones. Veamos algunos:
 - View System: componentes de la interfaz gráfica como pueden ser botones, iconos, listas, ...
 - Package manager: base de datos con información de aplicaciones instaladas en el sistema. De esta forma cuando una aplicación necesita un servicio de otra mediante este módulo obtiene información.
 - Resources manager: gestiona los recursos que no son código como por ejemplo los distintos *Strings* para internacionalización, imágenes, ...
 - Activity manager: una activity es una pantalla de la interfaz de usuario en una aplicación. Esta capa se encarga de la gestión del ciclo de vida de los activities y proporciona el soporte para la navegación entre los distintos activities.
 - Content Provider: permite el intercambio de información entre aplicaciones. Por ejemplo desde cualquier aplicación se puede acceder a los contactos.
 - Notification Manager: centro de notificaciones.
 - Location manager: acceso al GPS o a la localización de la que disponga el dispositivo.
- System Apps: en la capa superior se establecen las distintas aplicaciones. Tanto las que vienen por defecto (aplicación para email, sms, calendarios, contactos, ...) como las que instalemos o programemos nosotros. Esta es la capa que utiliza el usuario.

Estos elementos son, más o menos, comunes en todos los dispositivos Android (salvo que alguno no tenga algún hardware como puede ser la cámara o el giroscopio, lo que limita el uso de ciertas librerías). Sin embargo uno de los grandes problemas a los que se enfrentan los desarrolladores de Android es los que se denomina la fragmentación del mercado: la enorme variedad de dispositivos que existen con características diferentes.

Esta fragmentación es debida a los siguientes factores:


- Tipo de terminal: hay muchas gamas de terminales; medias, bajas, altas,... Todo eso lleva a decidir, al inicio del desarrollo, unos requerimientos de potencia y memoria mínimos a la hora de ejecutar una aplicación.
- Tamaño terminal: La cantidad de tamaños de pantallas y densidad de píxeles de cada pantalla es también amplia. Aquí conviene hacer pruebas con los emuladores y utilizar *layouts* automáticos para distribuir los componentes por la pantalla. También, por supuesto, se pueden plantear requerimientos de tamaño.
- Versión de Android: Otro problema es la fragmentación en las versiones de Android que se ejecuta en el terminal. Este es un problema importante ya que terminales con versiones antiguas que no soportan todos los elementos de las APIs más modernas.

Todo esto lleva al desarrollador a dos puntos importantes en las primeras y últimas fases del ciclo de vida de una aplicación.

- Por un lado en las primeras fases hay que decidir para qué ecosistema se va a diseñar.
- En las últimas, sobre todo en la de prueba y depuración, hay que probar distintos terminales y distintas versiones de Android para confirmar que todo funciona de forma correcta.

Para poder realizar todas estas pruebas no suele ser necesario disponer de una gran cantidad de terminales físicos si no que se usará el emulador que nos proporciona la plataforma Android.

Además no se debe olvidar que cuando hablamos de Android no solamente nos referimos a móviles y tablets. Si no a un sistema operativo que funciona y funcionará cada vez en más dispositivos: TV, coches, relojes, ...

	RAMA:	Informática	CICLO:	Desenvolvemiento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

3. IDE: Android Studio

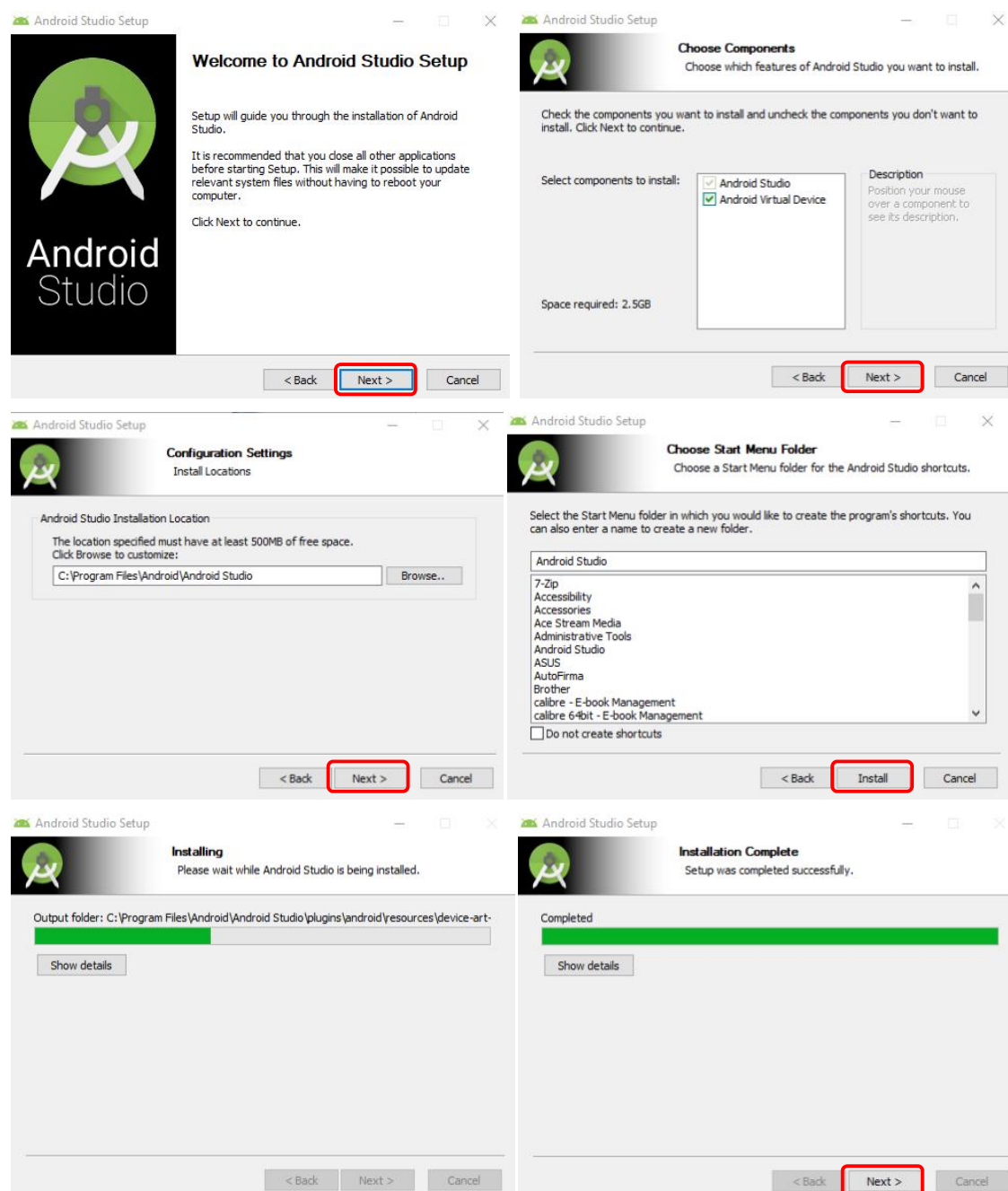
Como IDE de desarrollo se va a utilizar Android Studio. Este IDE es proporcionado por Google y está basado en el IDE IntelliJ IDEA de JetBrains: <https://developer.android.com/studio/index.html>


En este punto veremos una visión general del mismo para poder comenzar a trabajar. A medida que avance el curso iremos viendo los distintos elementos de una forma más detallada.

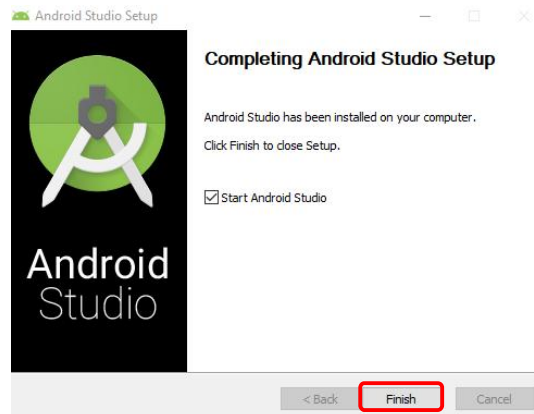
Para su instalación descargamos desde la siguiente página <https://developer.android.com/studio> la versión correspondiente al sistema operativo que estemos usando y procedemos a su instalación.

Windows

Ejecutamos el instalable que nos hemos descargado, por ejemplo el fichero android-studio-2020.3.1.24-windows.exe.



	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					



Linux

En Linux se descomprime el fichero descargado `android-studio-xxxxxx-linux.tar.gz` (por ejemplo `android-studio-2020.3.1.24-linux.tar.gz`). Para descomprimir se puede utilizar el interfaz gráfico pulsado con el botón derecho sobre el fichero y escoger una de las opciones extraer o mediante el comando de consola:

```
tar xvf android-studio-2020.3.1.24-linux.tar.gz
```

Donde la cadena 2020.3.1.24 se corresponde con la versión descargada.

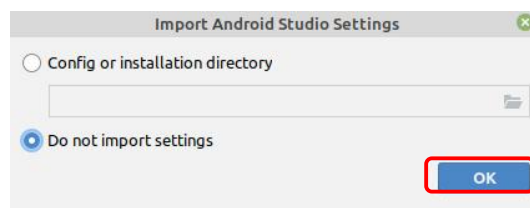
Una vez descomprimido esta carpeta, que solo contendrá el IDE, se puede mover a la ubicación que se desee.

Para arrancar Android Studio se ejecuta el fichero `studio.sh` disponible en el directorio `bin` de la carpeta de `android-studio`. Si se quiere añadir un acceso directo a este fichero en el escritorio se puede ejecutar la siguiente opción de menú: *Tools* → *Create Desktop Entry*.

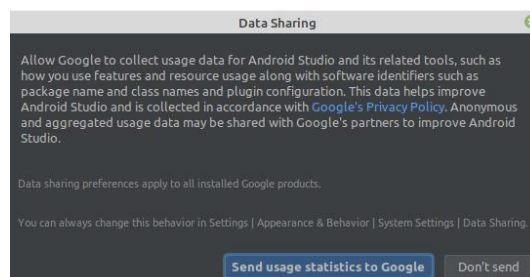
Pasos comunes

Tras la instalación ejecutaremos Android Studio.

La primera vez que se ejecute nos mostrara la siguiente ventana en la que podemos importar la configuración de una versión previa que tengamos instalada o realizar una instalación limpia. Escogeremos esta segunda opción.

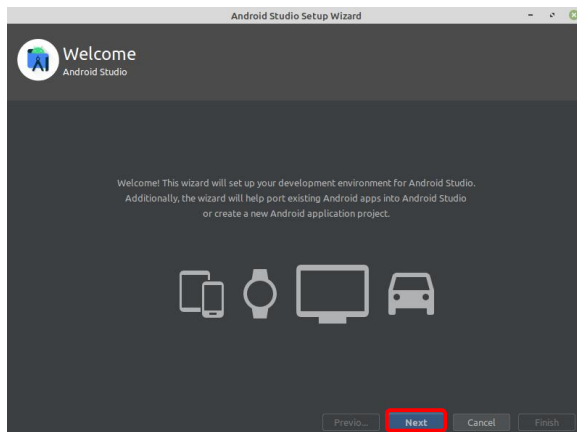


Después de la pantalla de presentación nos pregunta si queremos enviar información sobre la utilización que realicemos en Android Studio. En esta opción puedes elegir la opción que desees.

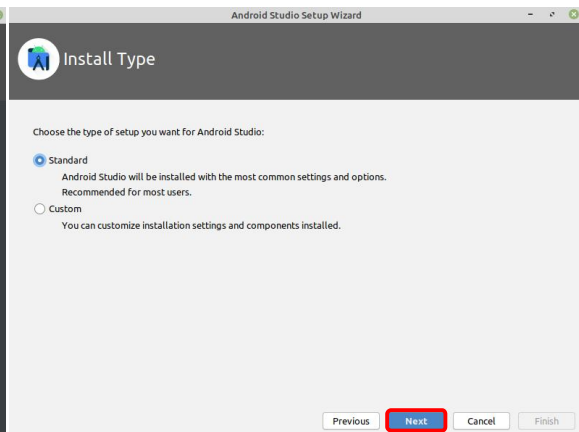


A partir de ahora se comienza a configurar la instalación.

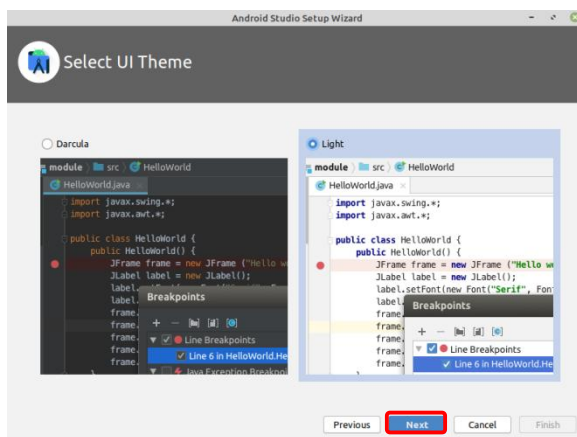
<div>COLEXIO</div> <div>VIVAS</div> <div>S.L.</div>	RAMA:	Informática	CICLO:	Desenvolvemento de Aplicacions Multiplataforma				
	MÓDULO	Programación Multimedia y Dispositivos Móviles					CURSO:	2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022		
	UNIDAD COMPETENCIA							



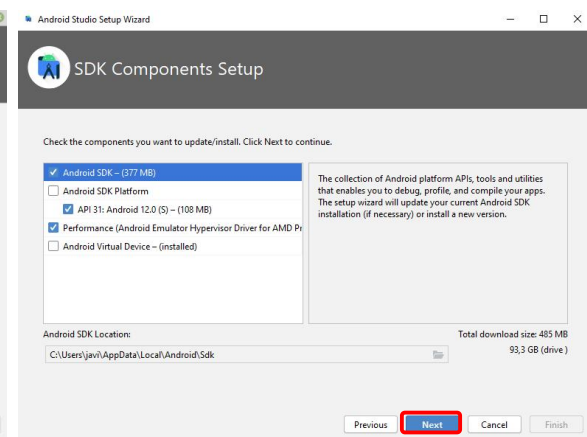
Pantalla de bienvenida



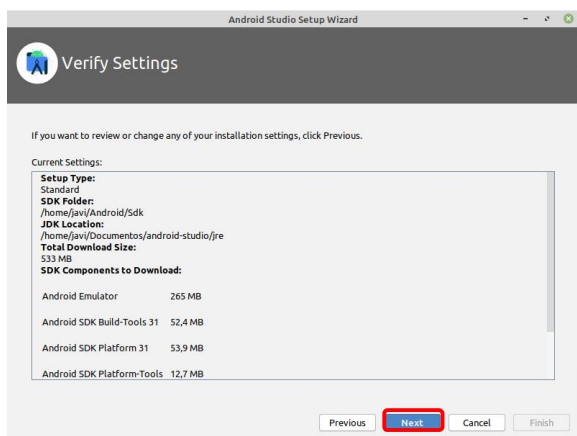
Tipo de instalación. Si se escoge Custom permite escoger la ubicación del JDK.



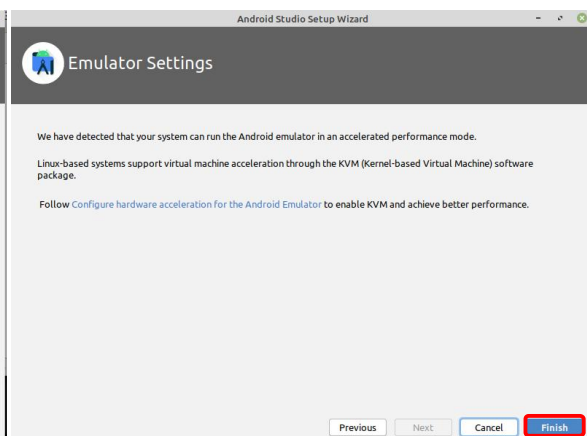
Modo de pantalla



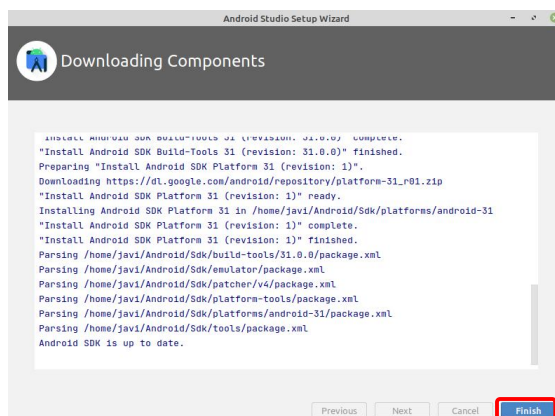
Solo Windows: Componentes SDK



Componentes a instalar



Solo Linux: Aceleración hardware




Finalización de la instalación

En Linux instrucciones de instalación están disponibles en el fichero *Install-Linux-tar.txt* del directorio *android-studio*. Las instrucciones completas se pueden consultar en: <https://developer.android.com/studio/install.html>

En algunos Linuxs de 64bits puede requerirse la instalación de algunas bibliotecas de 32 bits (su ausencia puede producir, por ejemplo en la creación de la tarjeta *sd* en el emulador) mediante el comando:

```
sudo apt-get install lib32z1 lib32ncurses5 lib32bz2-1.0 lib32stdc++6
```

La librería marcada en rojo puede no estar disponible, si fuese el caso habría que eliminarla del comando.

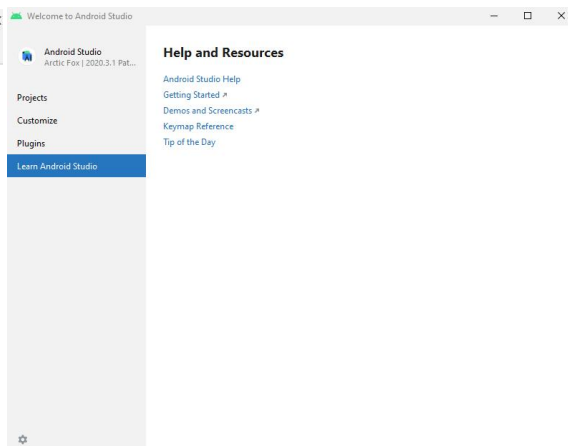
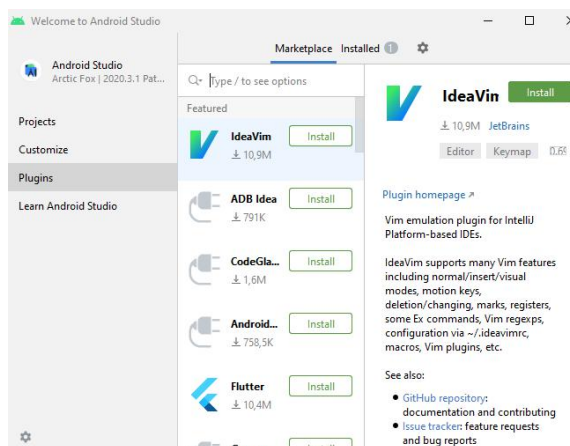
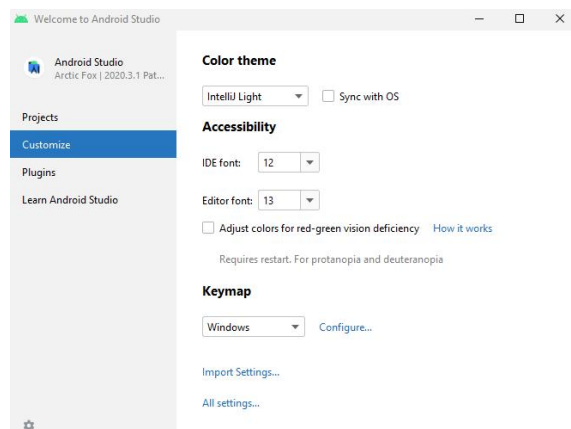
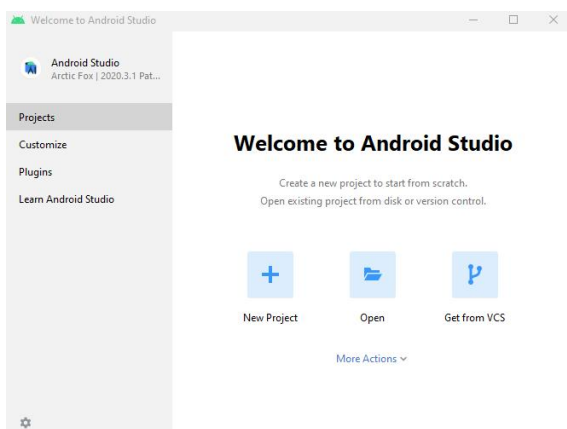
	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

3.1. Crear un proyecto

Al arrancar por primera vez Android Studio no muestra la siguiente ventana de bienvenida. Las próximas ejecuciones mostrarán el último proyecto abierto.

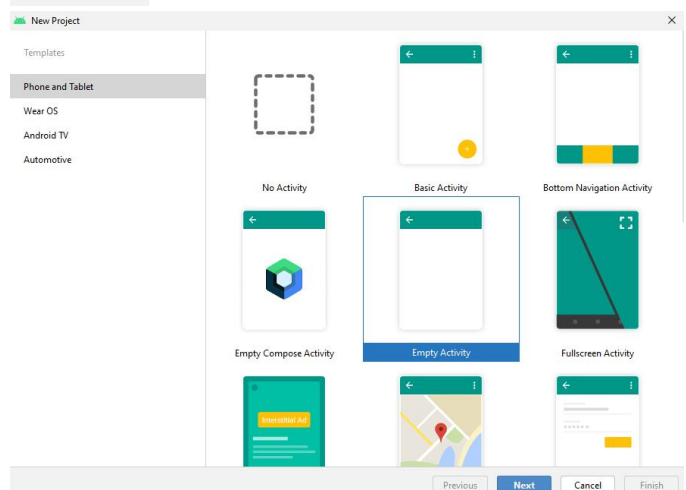
En esta ventana tenemos cuatro opciones en la parte izquierda:

- **Projects:** permite crear un proyecto nuevo. Se puede crear un nuevo proyecto, abrir un proyecto que este en disco o abrir un proyecto desde un sistema de control de versiones: soporta Git, Google Cloud, Mercurial y Subversión.
- **Customize:** permite realizar una personalización básica de Android Studio.
- **Plugins:** permite añadir para ampliar las funcionalidades de Android Studio.
- **Learn Android Studio:** enlaces de ayuda.



Crearemos un proyecto nuevo escogiendo la opción *New Project* de la opción *Projects*. En caso de tener un proyecto ya abierto usaremos la opción *File* → *New* → *New Project* en el menú de Android Studio.

- **Escoger el tipo de proyecto:** permite escoger el tipo de Activity inicial de la aplicación. En nuestro caso usaremos, dentro de la pestaña de *Phone and Tablet*, una actividad vacía (*Empty Activity*).



COLEXIO VIVAS S.L.	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

- **Configurar el proyecto:** Para finalizar la creación del proyecto se deberán rellenar los siguientes apartados:

1. **Name:** nombre de la aplicación. Este es el nombre que tendrá la aplicación tras su instalación (se puede cambiar después).
2. **Package name:** el nombre del paquete en el que se escribirá el código de la aplicación. Dejaremos el valor por defecto.
3. **Save location:** carpeta donde se guardará el proyecto. Dejaremos el valor por defecto.
4. **Language:** lenguaje que se utilizará para desarrollar la aplicación. Escogeremos Java.
5. **Minimum Api Level:** versión más baja de la Api a la que se dará soporte (este valor se puede modificar en cualquier momento durante el desarrollo). Dependiendo de las versiones del SDK que tengamos instaladas podría ser necesario instalar el SDK de una versión escogida de Android (se verá como realizar este paso más adelante).

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.0 Ice Cream Sandwich	15	
4.1 Jelly Bean	16	99,8%
4.2 Jelly Bean	17	99,2%
4.3 Jelly Bean	18	98,4%
4.4 KitKat	19	98,1%
5.0 Lollipop	21	94,1%
5.1 Lollipop	22	92,3%
6.0 Marshmallow	23	84,9%
7.0 Nougat	24	73,7%
7.1 Nougat	25	66,2%
8.0 Oreo	26	60,8%
8.1 Oreo	27	53,5%
9.0 Pie	28	39,5%
10. Android 10	29	8,2%

La versión mínima de la API que escojamos determina las funcionalidades que podemos utilizar en nuestra aplicación y la cantidad de dispositivos en la que se pueda instalar.


Mediante la opción *Help me to choose* se pueden ver qué porcentaje de los dispositivos del mercado soportan cada versión y las opciones relacionadas con la fragmentación de la que se habló anteriormente y que nos pueden ayudar a decidir para qué versión mínima del SDK queremos desarrollar nuestra aplicación. Pulsando en cada versión se puede ver los elementos soportados.

6. **Use legacy android.support libraries⁵:** conjunto de bibliotecas que da soporte a la biblioteca de compatibilidad AndroidX. A partir de Android 9.0 (nivel de API 28) es reemplazada con AndroidX⁶, que es parte de Jetpack. No lo usaremos.

Pulsando en el botón *finish* termina la creación del proyecto pudiendo comenzar la codificación de la aplicación.

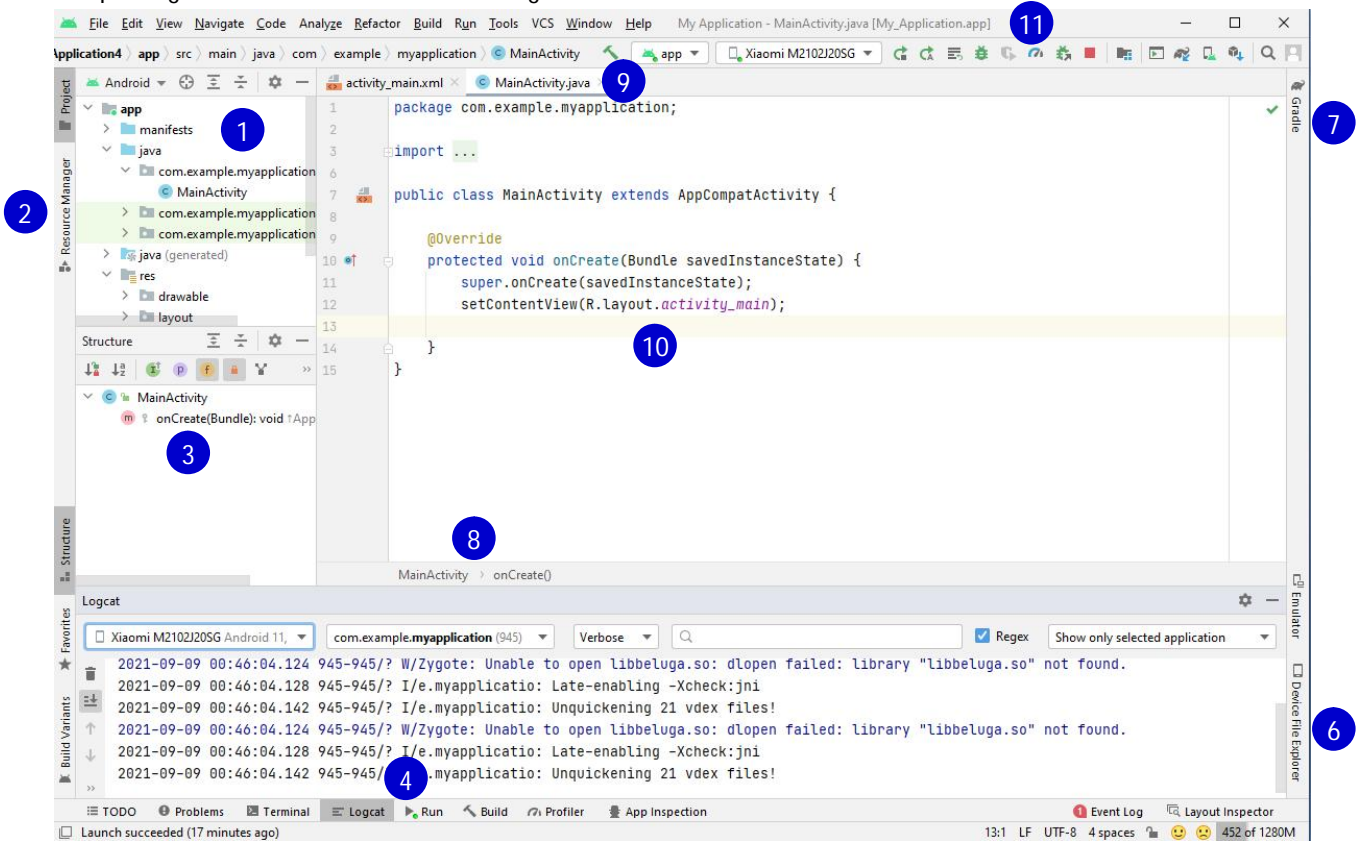
⁵ <https://developer.android.com/topic/libraries/support-library>

⁶ <https://developer.android.com/jetpack/androidx>

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					


3.2. Visión general de Android Studio

El aspecto general de Android Studio es el siguiente.

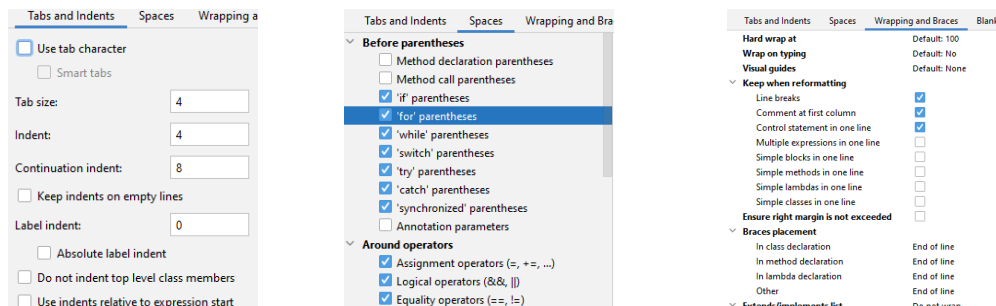


Donde podemos encontrar las siguientes secciones:

1. Estructura del proyecto. Se comentará en el tema siguiente.
2. Mediante *resource manager* se pueden administrar los recursos que añadamos a la aplicación. Estos se irán viendo en temas siguientes.
3. Estructura de cada Activity: pulsando en el botón se mostraran los métodos y atributos de la Activity.
4. Sección informativa: dependiendo de la pestaña muestra la siguientes información
 - o Todo: lista de mensajes *todo* de la aplicación.
 - o Problems: errores presentes en la aplicación
 - o Terminal: ejecuta un terminal del sistema.
 - o Logcat: Información de *log* de la ejecución de la aplicación. En la parte superior existen tres desplegables que permiten escoger de izquierda a derecha: el dispositivo, la aplicación y el nivel de log que se desea mostrar. Existe un campo de texto, marcado con el icono de la lupa, que nos permite filtrar texto. El último desplegable permite indicar de donde procede la información que se muestra.
 - o Run: información de la aplicación actual.
 - o Build: Información sobre la compilación de la aplicación.
 - o Profiler: muestra información del uso de CPU, memoria y uso de la red de nuestra aplicación. Nos permite realizar una traza de los métodos invocados para ver cuellos de botella, consumo excesivo de memoria, ...

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

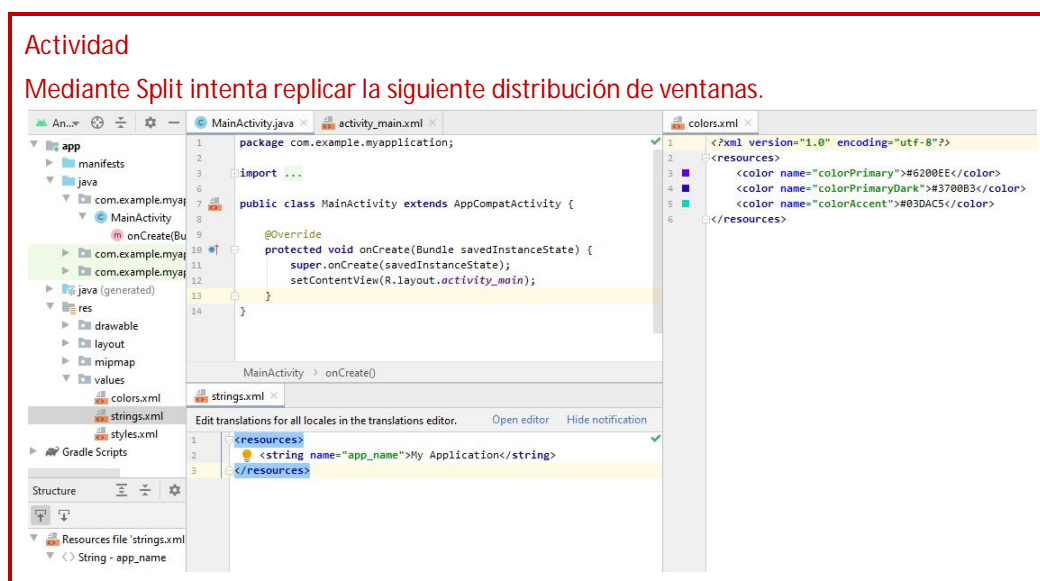
5. Permite, pulsando sobre ellos, cambiar el carácter de fin de línea, la codificación de caracteres del fichero que se está editando y el tamaño de la tabulación. Una configuración más avanzada se puede realizar en la siguiente opción: File -> Settings -> Editor -> Code Style -> Java. En esta opción se pueden modificar los estilos de código que establece, por defecto, Android estudio.



6. Es un gestor de archivos: se muestran el sistema de archivos de los dispositivos conectados al ordenador o del emulador. En la parte superior del mismo se puede cambiar entre dispositivos.
7. *Gradle* es el un sistema de compilación que usa Android Studio. Se pueden ejecutar tareas de forma individual. Normalmente solo usaremos la opción refrescar los proyectos cuando encontremos algún error de dependencias.
8. *Breadcrumb*: indica la clase y el método que estamos editando.
9. Pestañas con los diferentes archivos abiertos. De entre las opciones que aparecen pulsando con el botón derecha encima se pueden destacar las relacionadas con *split* que permite copiar y/o mover pestañas entre zonas mientras que *unsplit* permiten deshacer esa división.


Actividad

Mediante *Split* intenta replicar la siguiente distribución de ventanas.

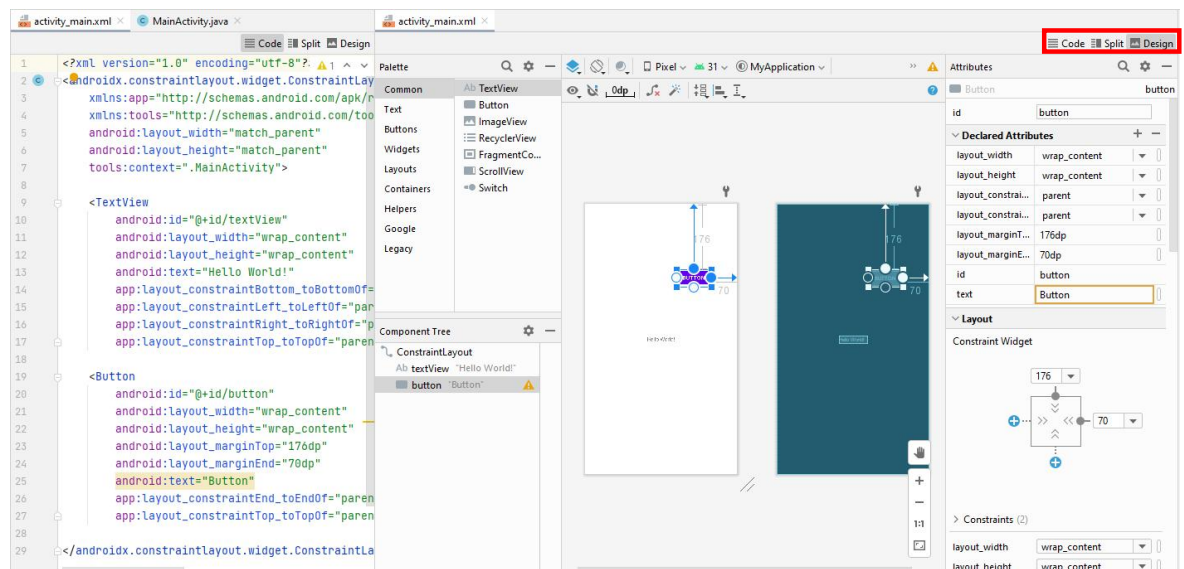


10. Ventana de edición. Dependiendo del tipo de fichero se verán distintos elementos:
- o Fichero java: ventana para editar código (primera imagen de este punto).
 - o Fichero de recursos XML: editor de texto para ficheros XML.



	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

- o Fichero XML con un *layout*: se puede usar tanto el editor XML como la herramienta de diseño de layouts para añadir y editar los elementos de la interfaz (este editor lo que hace internamente es modificar el fichero XML correspondiente). En la zona superior tenemos tres botones que permiten alternar entre: solo código XML, solo el editor de layouts o los dos de manera simultánea.



11. Iconos de acceso rápido.













- o La aplicación no se ha lanzado:




- o La aplicación ya se ha lanzado:



Los principales iconos disponibles son:

-  Ejecuta el proyecto y lo instala en el dispositivo seleccionado. Su atajo de teclado es may+F10.
-  En vez de compilar la aplicación solo aplica los cambios que se han realizado sobre una aplicación en funcionamiento reiniciando la actividad. La aplicación se lanza mucho más rápido pero no está disponible en todos los escenarios. Su atajo de teclado es ctrl+F10.
-  Es igual que el anterior pero si reiniciar la actividad. Su atajo de teclado es ctrl+Alt+F10.
-  Ejecuta la aplicación en modo debug. Su atajo de teclado es May+F9.
-  Run app with coverage: se activa cuando se pasan los test junit.
-  Android profiler: lanza la ventana de análisis de recursos consumidos por la aplicación.
-  Detiene la aplicación que actualmente está en ejecución.
-  Project Structure: permite modificar la configuración del proyecto. Es la misma opción que *File* → *Project Structure*. Entre otras opciones en *Modules* → *Properties* se puede cambiar el SDK con el que se compila la aplicación y en *Modules* → *Default config* la versión mínima de SDK soportado.
-  Sincroniza el proyecto con los ficheros de Gradle.
-  AVD Manager: Ejecuta el gestor de máquinas virtuales. Se verá en un punto siguiente.
-  SDK Manager: Ejecuta el gestor de versiones de SDK instaladas. Se verá en un punto siguiente.
-  Permite buscar, distintos elementos, dentro del proyecto.

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

Los atajos de teclado más habituales se muestran en el siguiente enlace:

<https://developer.android.com/studio/intro/keyboard-shortcuts?hl=es-419>

En ellos se pueden destacar:

Tipo	Descripción	Atajo
Completar código de manera básica	Muestra sugerencias básicas para, entre otros, variables, tipos, métodos y expresiones.	Control + Barra espaciadora
Completar código de manera inteligente	Muestra opciones relevantes en función del contexto. La función "Completar de manera inteligente" reconoce el tipo y los flujos de datos previstos.	Control + Mayús + Barra espaciadora
Completar enunciados	Completa la instrucción actual agregando entre otros elementos que faltan, como paréntesis, corchetes, llaves, ...	Control + Mayús + Intro
Acciones de intención	Realizar correcciones rápidas	Alt + Intro.
Buscar una clase	Busca una clase en el código y navega hasta ella	Control + N
Buscar referencia	Busca en el código todas las referencias a una clase, método, campo, parámetro o instrucción actual.	Alt + F7
Reformar código	Aplica un formato de forma automática al código	Cotrol + Alt + L
Indentar código	Identa de forma automática todo el código.	Cotrol + Alt + i

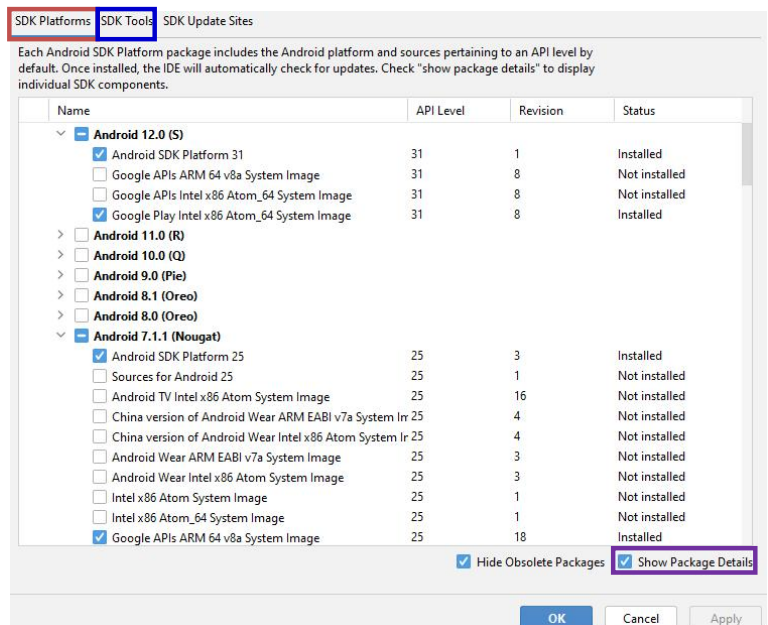
4. SDK Manager


Permite añadir soporte a versiones de Android que actualmente no se tienen instaladas.



En la imagen siguiente se puede ver que están marcadas, en azul, las versiones instaladas. Para programar una aplicación para una API específica esta debe estar instalada.

- La pestaña **SDK Platforms**: permite instalar otras versiones del SDK Android. Si una vez creada una aplicación queremos cambiar la versión mínima que soporta solo se podrán escoger entre las versiones que estén instaladas. Si la versión deseada no aparece habrá que instalarla previamente desde este gestor. Mediante la opción **Show Package Details** permite seleccionar elementos individuales de cada paquete.
- La pestaña **SDK Tools**: permite instalar una serie de herramientas como pueden ser las herramientas SDK y el emulador (instalados por defecto), el driver USB de Google o habilitar el soporte de aceleración hardware (ver sección dedicada al emulador para un descripción de cómo activar la aceleración hardware del emulador).



	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

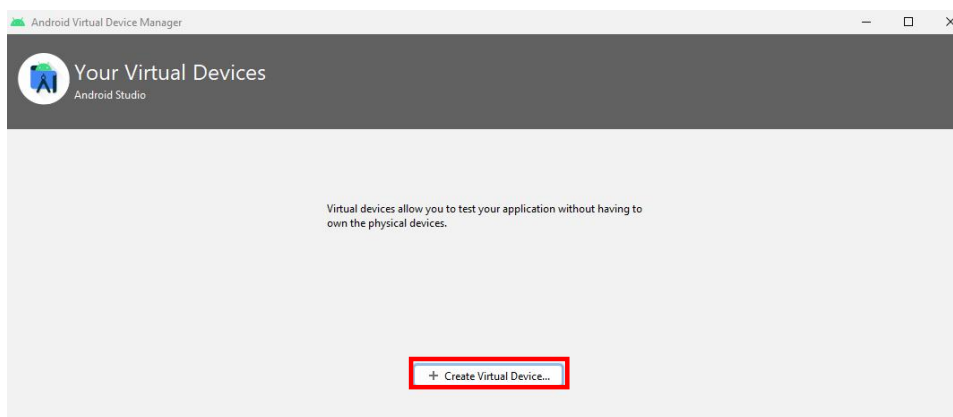
5. El emulador, dispositivos virtuales y la terminal de control

Antes de poder ejecutar una aplicación, tenemos que tener algún dispositivo en la cual ejecutarla. Antes de usar un dispositivo físico vamos a ver como se crea y configura un emulador de Android para poder hacer las pruebas necesarias.

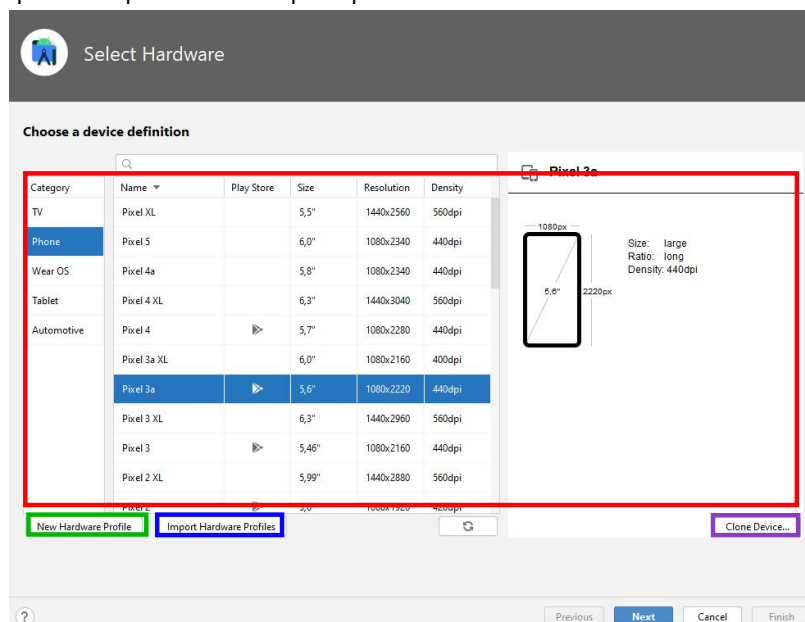
En primer lugar ejecuta el Android Virtual Device Manager (AVD Manager).



En esta ventana aparecen, en caso de haberlos, los dispositivos virtuales creados. Para crear un dispositivo virtual (AVDs) nuevo se usa el botón *Create Virtual Device*⁷ que ofrece tres formas de crear el dispositivo:




- A partir de definiciones que vienen por defecto (Device Definitions) en Android Studio para los distintos tipos de dispositivo. En un principio solo usaremos Phone o tablet.

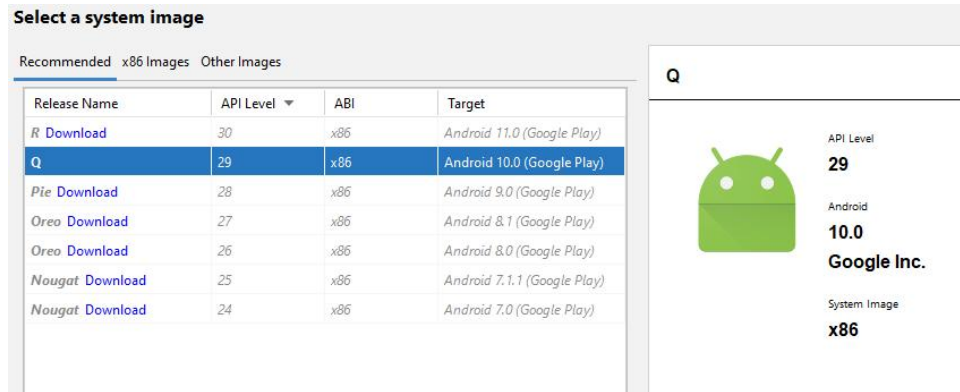


- **Creando una definición a medida:** Mediante el botón *New Hardware Profile*, definir un nuevo dispositivo escogiendo las características del mismo. Algunas de las características son: memoria, tamaño de pantalla, resolución, cámara, ...
- **Importando una definición de disco.**
- **Clonado una definición existente** y modificándola.

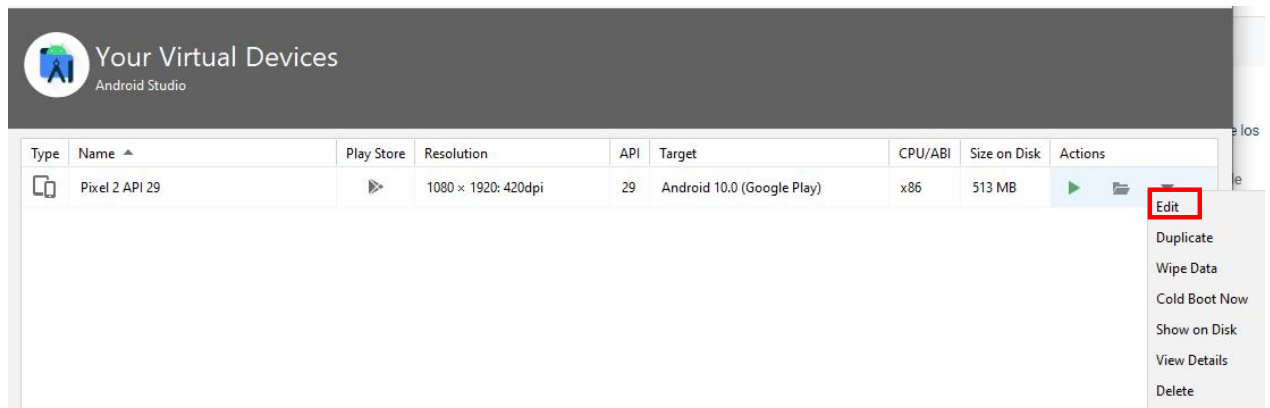
⁷ Guía de creación de dispositivos virtuales: <https://developer.android.com/studio/run/managing-avds.html>

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					


Cuando creemos un dispositivo deberemos escoger la versión de la Api de Android va a tener el dispositivo virtual instalado. En caso de no tener esa versión descargada en la propia ventana de selección del la imagen del sistema nos ofrece la opción de descargarla e instalarla de forma directa.




Una vez creada se puede editar sus características: orientación por defecto, versión de API, si tiene aceleración grafica por hardware o software, tipo de cámaras, memoria interna y tamaño de tarjeta SD usada, ...



Además de editarlo la otras opciones relevantes que tenemos son:

- Arrancar el emulador: mediante el icono  de todos modos cuando se instala una aplicación en un dispositivo virtual este se arranca de forma automática.
- Duplicate: crear una copia exacta.
- Wipe data: borrar la información que pudiese tener el dispositivo virtual.
- Delete: borrar el dispositivo virtual.

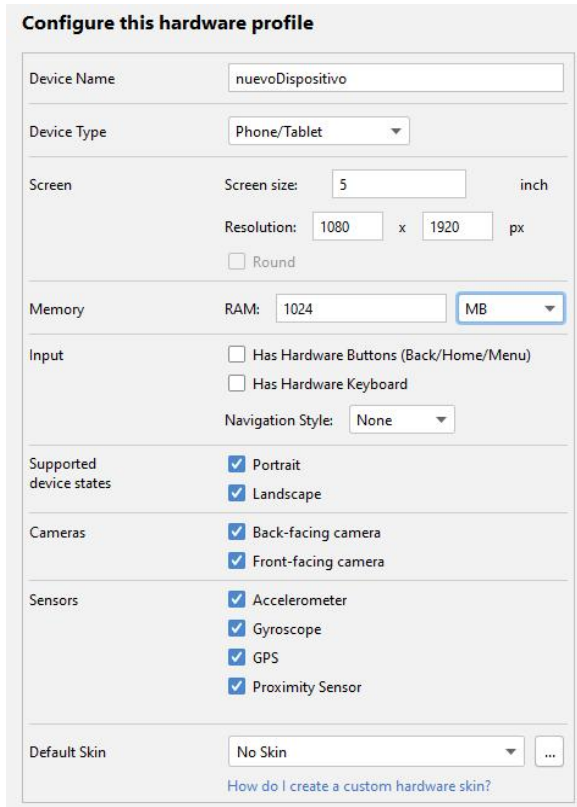
Siempre que se pueda suele ser mejor probar las aplicaciones en terminales reales, sobre todo por disponer de todos los elementos hardware que el emulador no dispone o por velocidad ya que el emulador es más lento.

	RAMA:	Informática	CICLO:	Desenvolvimento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

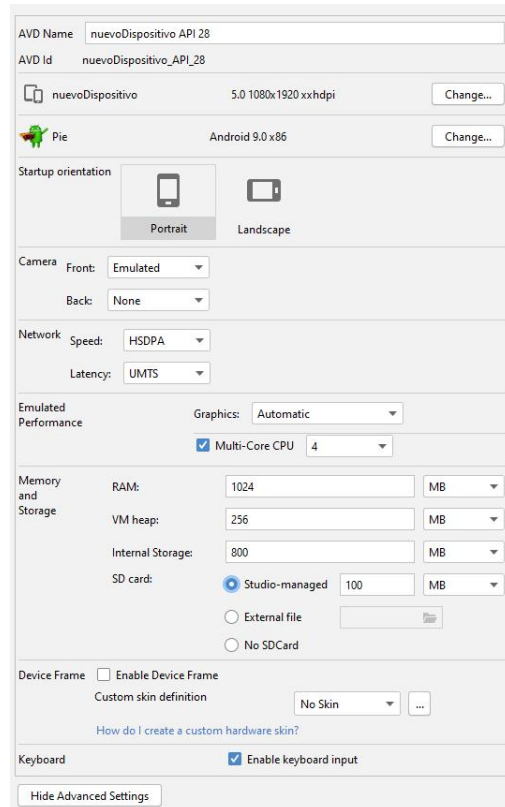
Actividad

Crearemos un nuevo dispositivo virtual llamado *nuevoDispositivo*. Para ello rellena los campos tal y como se ve en las siguientes imágenes.

Pantalla de creación



Pantalla de modificación, avanzada, de configuración



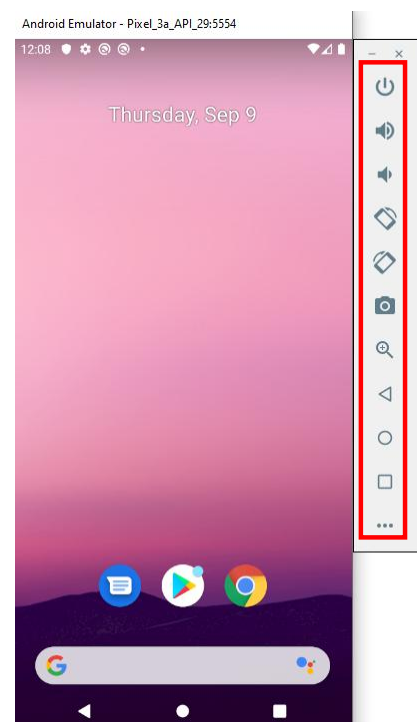
Una vez creado el dispositivo virtual procedemos a arrancarlo y ejecutar el proyecto seleccionando en este dispositivo virtual.


El dispositivo virtual dispone de una serie de iconos, a la derecha del mismo, que permite realizar una serie de tareas básicas sobre el mismo: subir y bajar volumen, rotar, capturar la pantalla y realizar zoom. También dispone de los tres botones típicos de un dispositivo Android: inicio, cancelar y atrás.

Además se puede interaccionar con el dispositivo mediante atajos de teclado. Una lista se puede consultar en:

<https://developer.android.com/studio/run/emulator?hl=es#tasks>

Como ya se ha comentado el emulador es más lento que un dispositivo físico pero sin embargo tiene como ventaja el permitir probar, mediante distintos dispositivos virtuales, muchas configuraciones diferentes. Además permite simular distintas situaciones para comprobar cómo se comporta la aplicación. Por ejemplo se puede simular la recepción de SMSs, la localización GPS, realización de llamadas, movimiento e inclinación del dispositivo, ...



	RAMA:	Informática	CICLO:	Desenvolvemiento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

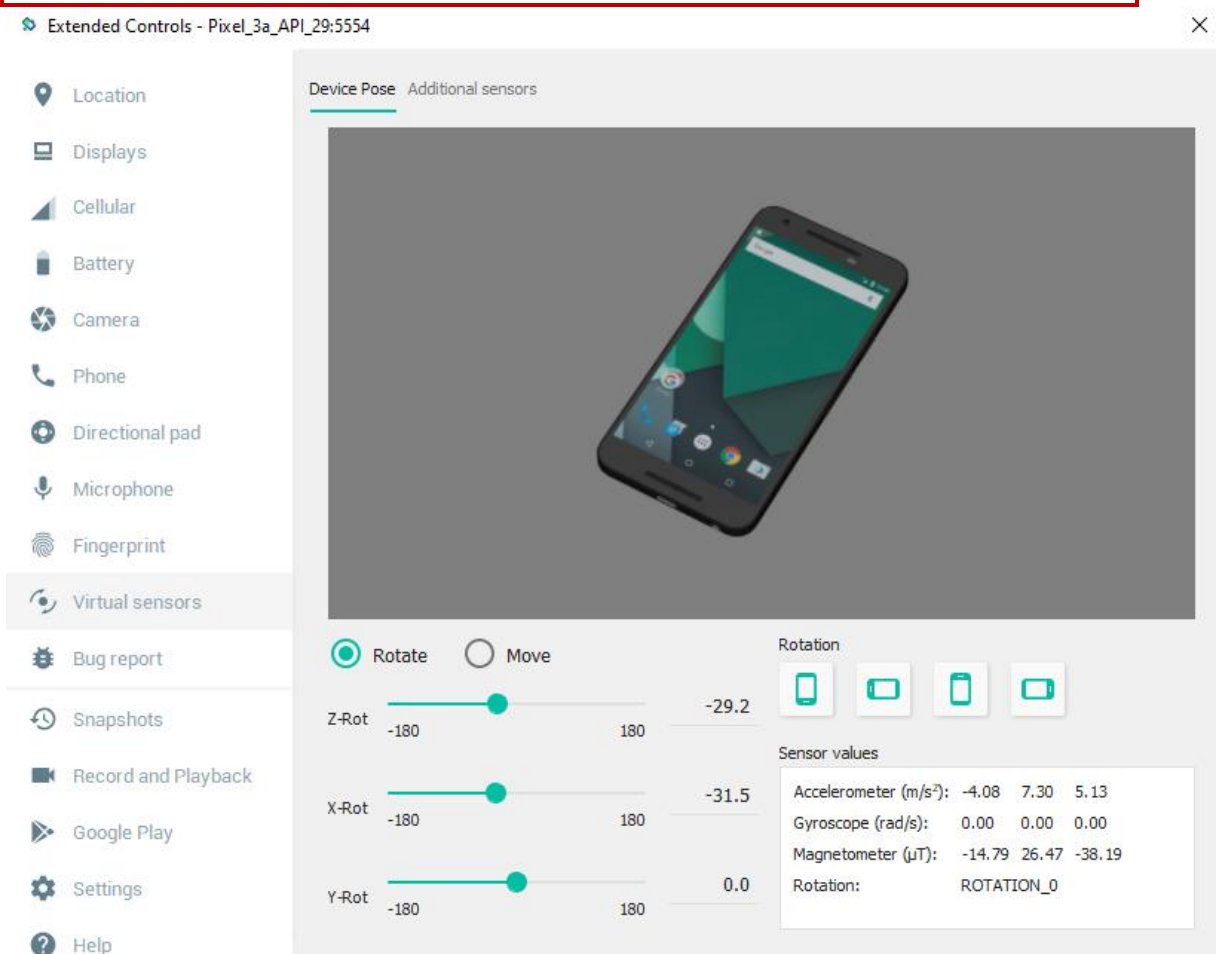
Tenemos dos opciones para realizar estas pruebas:

- Mediante la interfaz grafica (pulsando en los tres puntos en la parte inferior de la barra de iconos):

Se puede observar en la imagen siguiente que se pueden probar los siguientes elementos: localización del dispositivo, tipo de señal y velocidad, estado de batería, cámara, llamadas y sms, grabación de sonidos, movimientos del dispositivo, ...

Actividad

Navega y prueba las distintas opciones que nos ofrece el emulador.



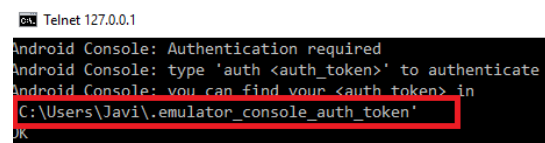
- Mediante comandos: Para ello usamos una conexión telnet con dicho emulador. Observa en la barra de título del emulador el número que tiene al final del mismo, seguramente sea 5554 o similar, indica el puerto al que hay que conectarse. Por tanto podemos ejecutar desde un terminal:


```
telnet localhost 5554
```

Una vez realizada la conexión mediante el comando help se pueden para ver los distintos comandos disponibles.

Para ejecutar la mayoría de los comando se requiere realizar una autenticación contra el dispositivo virtual. Este paso se realiza mediante un token de autenticación que se crea en el fichero .emulator_console_auth_token en nuestro directorio home (marcado en la imagen). El comando para realizar la autenticación es:

```
auth wnbddadraULNpGX (aquí hay que poner el valor del token que tengas en el fichero)
```



	RAMA:	Informática	CICLO:	Desenvolvemeto de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

Actividad

Prueba los comandos siguientes observando cómo cambian en la barra de notificaciones del emulador los iconos correspondientes a la batería y el tipo de conexión a la red.

network speed edge
network speed full
power capacity 15
power status not-charging

Prueba ahora los siguientes comandos:

sms send 555555555 "Compra galletas"
gsm call 555555555

Prueba otros comandos de las páginas que se muestran a continuación.

Para ver todos los comandos disponibles podemos consultar las páginas:

<https://developer.android.com/studio/run/emulator-console.html>

En la siguiente página se pueden ver las funciones no compatibles con el emulador:

<https://developer.android.com/studio/run/emulator?hl=es#limitations>

Para un uso más avanzado del emulador se pueden consultar los siguientes enlaces:

<https://developer.android.com/studio/run/emulator-commandline.html>

<http://developer.android.com/intl/es/tools/devices/emulator.html>

5.1. Aceleración Hardware del emulador⁸:

Para poder utilizar la aceleración hardware del emulador se deberán cumplir los siguientes requisitos:

- Activar en la bios las extensiones Intel Virtualization Technology (VT, VT-x, vmx) y Execute Disable (XD) Bit en procesadores Intel y las extensiones de la virtualización AMD (AMD-V, SVM) en procesadores AMD.
- Procesador de 64 bits.
- Usar, como mínimo, una versión 17 de las SDK tools.
- Usar una imagen de sistema basada en x86 para el emulador.

Para habilitar la aceleración gráfica se deberán realizar los siguientes pasos:

- Linux

Se necesita tener instalado el paquete KVM (Kernel Virtual Machine).

En caso de que no esté instalado se puede instalar mediante el comando:


```
sudo apt-get install qemu-kvm libvirt-bin ubuntu-vm-builder bridge-utils ia32-libs-multiarch
```

- Windows

Dependiendo de los casos mostrados en la siguiente tabla usaremos una de las dos siguientes opciones: Intel HAXM o WHPX.

Criterio	Opción
Procesador Intel y no se necesita utilizar Hyper-V al mismo tiempo que el emulador Android.	Intel HAXM
Procesador Intel y se necesita utilizar Hyper-V al mismo tiempo que el emulador Android.	WHPX
Procesador AMD	WHPX

⁸ <https://developer.android.com/studio/run/emulator-acceleration?hl=es#dependencies>

	RAMA:	Informática	CICLO:	Desenvolvemiento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

o Intel HAXM.

- ❖ Disponer de un Windows 7, 8 o 10 y deshabilitar Hyper-V en el panel de control de Windows.
- ❖ Instalar Intel HAXM desde la *pestaña SDK Update Sites* en el SDK manager.

SDK Platforms	SDK Tools	SDK Update Sites
These are the sites checked for Android SDK Updates Tools. When unchecked, the Android Studio SDK Manager will not check the site for updates. Adding additional add-on updates sites can add new add-ons or extra SD packages.		
Enabled	Name	URL
<input checked="" type="checkbox"/>	Android Automotive System Images	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Android Repository	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Android System Images	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Android TV System Images	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Android Wear System Images	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Android Wear for China System Images	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Glass Development Kit, Google Inc.	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Google API add-on System Images	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Google API with Playstore System Images	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Google Inc.	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Intel HAXM	https://dl.google.com/android/reposit
<input checked="" type="checkbox"/>	Offline Repo	file://C:/Programas/Desarrollo/Android/

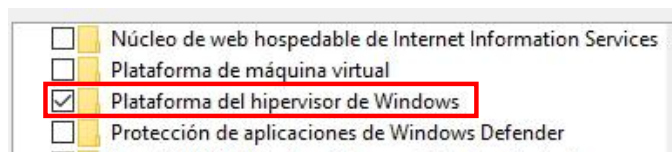
o WHPX


Para instalar Windows Hypervisor Platform (WHPX) se deberán cumplir los siguientes requisitos:

- ❖ Android Studio 3.2 Beta 1 o superior.
- ❖ Android Emulator version 27.3.8 o superior.
- ❖ Windows 10 con la actualización de abril de 2018 o superior.

Los pasos para la instalación son:

- ❖ Acceder programas y características del panel de control.
- ❖ Pulsar, a la izquierda, en Activar o desactivar las características de Windows.
- ❖ Marcar la opción: plataforma de hipervisor de Windows.



	RAMA:	Informática	CICLO:	Desenvolvemiento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

6. Conexión de dispositivos

El emulador realmente se convierte en una buena herramienta pero no para trabajo continuo debido a que es más lento que un dispositivo real. Es preferible en la medida de lo posible probar la aplicación en un dispositivo real y dejar el emulador para otras pruebas de funcionamiento.

Para poder usar un dispositivo hay que realizar varias acciones en el dispositivo móvil (la localización de las opciones puede variar entre versiones de Android y marcas).

1. Activar opciones de desarrollador (o de desarrollo): A partir de la versión 4.2 de Android vienen, por defecto, ocultas. Para activarlas iremos a Ajustes → Acerca del teléfono (o Sobre el teléfono) → y pulsamos siete veces sobre el número de compilación. En móviles de Xiaomi hay que pulsar sobre Versión de MIUI.
2. En Ajustes → Ajustes adicionales → Opciones de desarrollador tenemos que activar las siguientes opciones:
 - ❖ Depuración USB: permite la ejecución de aplicaciones controlándolas desde el ordenador de desarrollo.
 - ❖ Instalar vía USB
 - ❖ En algunos casos puede ser necesario activar la opción: Depuración USB (Ajustes de seguridad).
3. En versiones antiguas de Android en Ajustes → Seguridad → Activar *Fuentes desconocidas* (u *orígenes desconocidos*) para permitir la instalación de aplicaciones que no provengan de Google Play.

Una vez se termina de trabajar con Android Studio se recomienda desactivar estas opciones.

Actividad

Consulta todas las opciones disponibles en la opciones de desarrollo y prueba algunas.


En caso que el sistema operativo no detecte el dispositivo se pueden probar las siguientes acciones:

Windows

- Instalar, desde el Device Manager, el driver USB de Google.

SDK Platforms SDK Tools SDK Update Sites		
Below are the available SDK developer tools. Once installed, the IDE will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.		
Name	Version	Status
<input checked="" type="checkbox"/> Android SDK Build-Tools 31		Installed
<input type="checkbox"/> NDK (Side by side)		Not Installed
<input type="checkbox"/> Android SDK Command-line Tools (latest)		Not Installed
<input type="checkbox"/> CMake		Not Installed
<input type="checkbox"/> Android Auto API Simulators	1	Not installed
<input type="checkbox"/> Android Auto Desktop Head Unit Emulator	1.1	Not installed
<input checked="" type="checkbox"/> Android Emulator	30.8.4	Installed
<input checked="" type="checkbox"/> Android Emulator Hypervisor Driver for AMD Processors (installer)	1.7.0	Installed
<input checked="" type="checkbox"/> Android SDK Platform-Tools	31.0.3	Installed
<input type="checkbox"/> Google Play APK Expansion library	1	Not installed
<input type="checkbox"/> Google Play Instant Development SDK	1.9.0	Not installed
<input type="checkbox"/> Google Play Licensing Library	1	Not installed
<input type="checkbox"/> Google Play services	49	Not installed
<input checked="" type="checkbox"/> Google USB Driver	13	Installed
<input type="checkbox"/> Google Web Driver	2	Not installed

- Instalar un driver USB que de soporte al dispositivo. Puedes ver cómo hacerlo en <https://developer.android.com/studio/run/oem-usb?hl=es> o busca en otras webs.

	RAMA:	Informática	CICLO:	Desenvolvemiento de Aplicacions Multiplataforma		
	MÓDULO	Programación Multimedia y Dispositivos Móviles				CURSO: 2º
	PROTOCOLO:	Apuntes clases	AVAL:	1	DATA:	2021/2022
	UNIDAD COMPETENCIA					

Linux

Si Linux no detecta el dispositivo podemos seguir las siguientes instrucciones (para más información se puede consulta el siguiente enlace <https://developer.android.com/studio/run/device.html?hl=es>).

- Mediante el comando `lsusb` vemos el listado de dispositivos USB conectados:

```
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 002: ID 203a:fff9 Bus 001 Device 003: ID 04e8:685e Samsung Electronics Co GT-I9100
```

- En el listado anterior hay que buscar el que corresponde al móvil y quedarse con los dos números resaltados: el primero es el identificador del fabricante y el segundo el del modelo de dispositivo.
- En Linux existe un gestor de dispositivos (el daemon `udev` desde el kernel 2.6) que se encarga de controlar los ficheros de dispositivos del directorio `/dev` (y por tanto los dispositivos conectados al sistema).

Tenemos que informar a dicho servicio de qué es lo que estamos conectando. Para ello se establece lo que se denomina una regla `udev`. Creamos entonces un archivo con el nombre `50-android.rules` en el directorio `/etc/udev/rules.d/`

Es imprescindible que la extensión sea `rules`, el nombre no importa tanto, simplemente el número establece el orden de aplicación de reglas dentro de dicho directorio.

- Escribe la siguiente regla:

```
SUBSYSTEM=="usb", ATTRS{idVendor}=="04e8", ATTRS{idProduct}=="685e", MODE="0666", GROUP="plugdev"
```

El `idVendor` es el fabricante y el `idProduct` el modelo según la numeración de `lsusb`.

La asignación `MODE` especifica permisos de lectura y escritura, y `GROUP` define el grupo Unix al que corresponde el nodo del dispositivo.

- Ejecutamos el siguiente comando:

```
chmod a+r /etc/udev/rules.d/51-android.rules
```

- Cargamos las reglas en el servicio:

```
udevadm control --reload-rules
```

- Conectamos el dispositivo (si lo teníamos conectado desconectamos y volvemos a conectar).

7. ABD

Como indica Google, `adb`, es un comando de consola versátil que permite comunicarse con el dispositivo o emulador Android. Puede usarse para cosas tan variadas como ver dispositivos conectados o para instalar apps. El comando se encuentra en el directorio:

```
<path_instalacion_ADT>/sdk/platform-tools/
```

Por ejemplo, en Windows, para conectarse con el Shell de `abd` podemos usar el comando:

```
C:\Users\usuario\AppData\Local\Android\sdk\platform-tools\adb.exe shell
```

Podemos obtener más información sobre `abd` en:

<http://developer.android.com/intl/es/tools/help/adb.html>