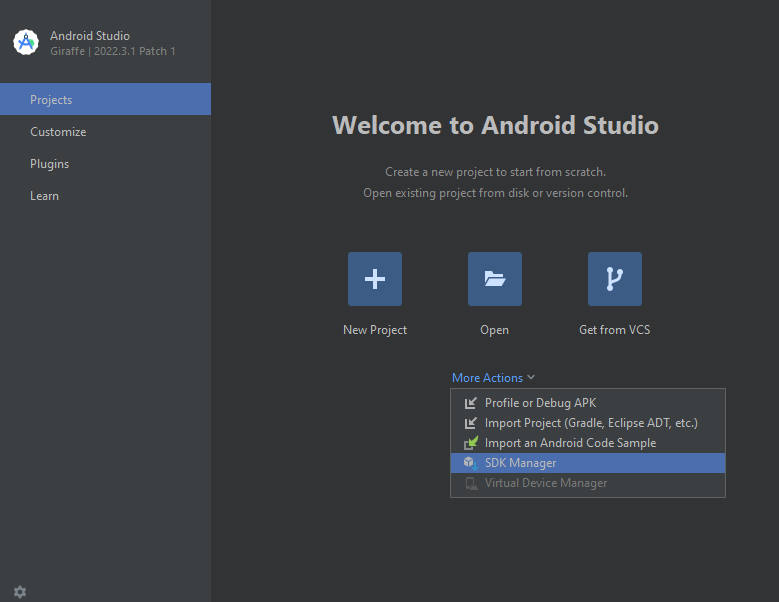
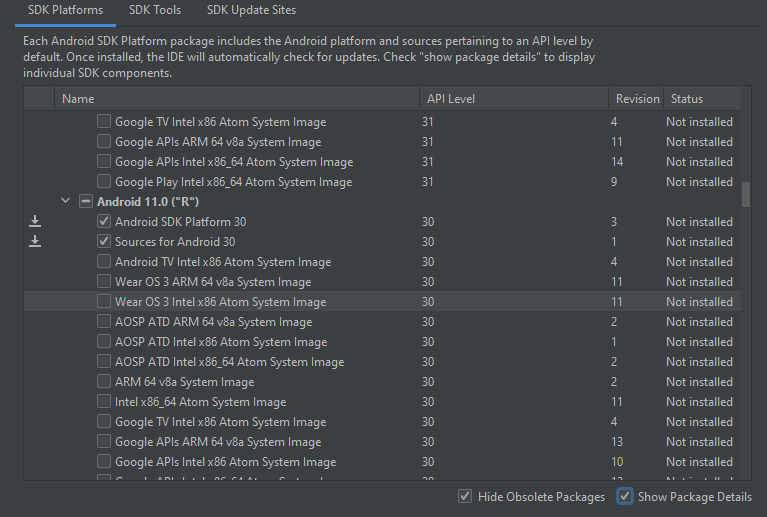
**Preludio**

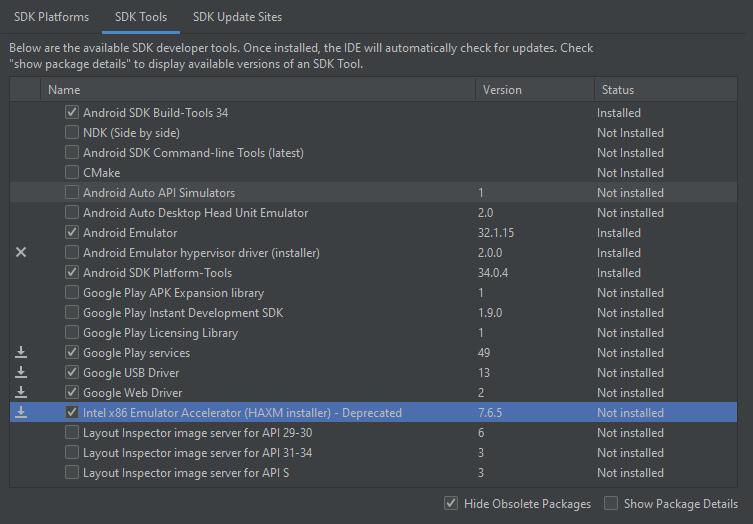
Antes de comenzar, seleccionaremos e instalaremos las extensiones necesarias en Android Studio. Para hacerlo, vamos a la opción "More Actions" (o "Más Acciones" en español) y seleccionamos "SDK Manager".



En primer lugar, instalaremos Android 11.0 ("R"), centrándonos especialmente en "Android SDK Platform 30" y "Sources for Android 30". Esto se debe a que Android 11.0 (API nivel 30) es una de las versiones más importantes.



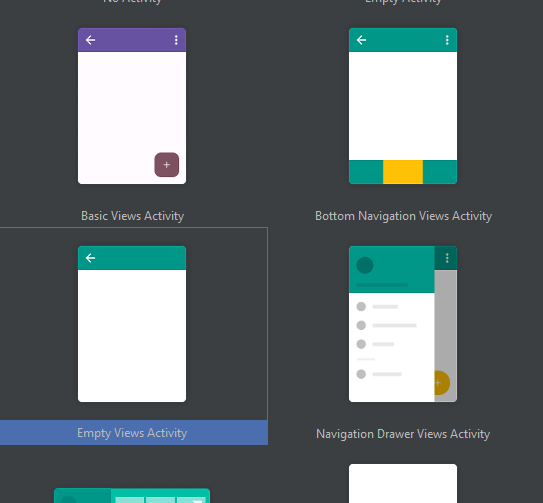
Luego, vamos a "SDK Tools" y desactivamos "Android Emulator hypervisor driver" y activamos "Google Play services", "Google USB Driver", "Google Web Driver" y "Intel x86 Emulator Accelerator...". El resto de opciones pueden mantenerse como están.

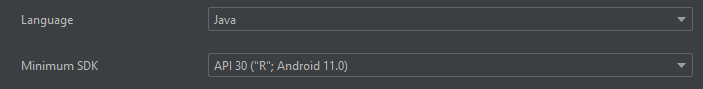


Con estos pasos, estamos listos para crear un nuevo proyecto.

**Primeros pasos**

El primer proyecto que crearemos será un "Empty Views Activity". Seleccionamos Java como lenguaje, API 30 y hacemos clic en "Finish" para iniciar la creación del proyecto.





Una vez dentro del proyecto, veremos su estructura jerárquica.

**Jerarquía de un proyecto Android**

Dentro de la jerarquía de un proyecto Android, a la izquierda, podemos encontrar la aplicación con todas sus carpetas y los archivos de configuración en "Gradle Scripts". Estos archivos están escritos en el lenguaje de construcción Gradle y se utilizan para definir cómo se construirá y compilará el proyecto de Android, similar al Maven de NetBeans. Entre los archivos, el "build.gradle.kts" almacena las dependencias.

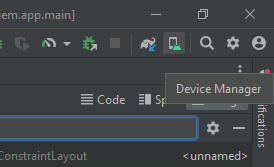
Volviendo a la carpeta de la aplicación, en "src > res" encontramos el archivo "AndroidManifest.xml", que describe la secuencia de métodos y el orden en el que se ejecutarán. El primero de ellos es el método "onCreate", equivalente al "main" de NetBeans. También tenemos una carpeta llamada "test" donde se realizan pruebas, que no abordaremos por el momento. Luego está la carpeta "res", donde se almacenan recursos como imágenes y sonidos. Por último, en "layout", encontramos el archivo "activity\_main.xml", que técnicamente es similar a la vista en NetBeans.

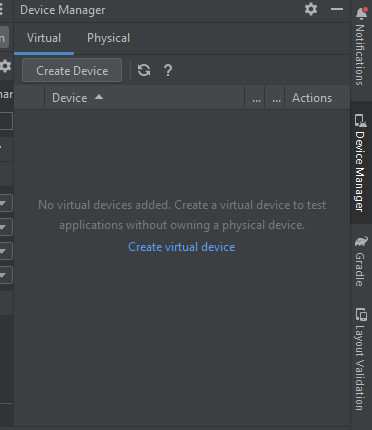
Otro archivo importante es "string.xml", que almacena todo el texto de la aplicación y es útil para realizar traducciones.

**Empecemos con activity\_main.xml**

La vista "activity\_main.xml" tiene, por defecto, "Constraint Layout", pero para evitar problemas futuros, cambiaremos a "Linear Layout". Para hacerlo, hacemos clic derecho y seleccionamos "Convert View". También podemos cambiar la orientación de como se colocaran los objetos en la vista de vertical a horizontal buscando la propiedad "orientation".

Antes de continuar con los componentes, hablaremos sobre la creación de una máquina virtual. Para ello, vamos a "Device Manager", ubicado en la parte superior derecha, y luego a "Create Device". Elegimos el dispositivo, como Pixel 5 o Pixel 6, y seleccionamos la versión con "R" o "S" si es necesario. Para ejecutarlo, hacemos clic en el botón de reproducción. Se recomienda cambiar el "View Mode" a "Window" para una mejor experiencia.





**Componentes (**[**API**](https://developer.android.com/guide/topics/manifest/uses-sdk-element?hl=es-419)**)**

Comencemos a explorar los componentes de Android.

* **View**

Los componentes y elementos en Android Studio se conocen como "View" o "vista".

* **TextView**

El "TextView" es un componente de texto básico que hereda de "View".

* **Button**

El "Button" o botón también hereda de "TextView" y representa un elemento de texto que funciona como botón.

**EditText**

El "EditText" es un campo de texto editable en la aplicación y hereda de "TextView". A veces, se le llama "PlainText".

**Métodos**

* **Compact Activity**: Este método se utiliza varias veces en el desarrollo de aplicaciones Android.
* **FindViewById**: Este método se utiliza para buscar elementos por su ID, que son identificadores comúnmente utilizados. Android genera automáticamente una clase de constantes que almacena estos IDs.
* **Métodos de componentes**: Para seleccionar métodos de un componente específico, debemos importar la clase con "Alt + Intro" y seleccionar el componente deseado. De lo contrario, Android Studio seleccionará automáticamente los métodos de "View".