

# INTRODUCCIÓN A ANDROID



# ¿Qué es Android?

- Sistema Operativo de última generación.
- Está basado en Linux.
- Creado por Google.
- Interfaz basada en tecnología DMI (Direct Manipulation Interface).
- Pensado para dispositivos con pantallas táctiles.
- Programado en C/C++ con licencia opensource.

# Versiones



# ¿Problemas?

- Intentar dar soporte al mayor número posible de APIs.
- Definir el rango de versiones para las que funcionará nuestra App:
  - Especificar la versión mínima:
    - android:minSdkVersion
  - Especificar la versión máxima:
    - android:maxSdkVersion

# Información de Interés

- Android SDK:
  - <http://developer.android.com>
- Canal de YouTube de desarrolladores Android:
  - <https://www.youtube.com/user/androiddevelopers>



*developers*

# Más referencias útiles



- Stackoverflow:
  - [stackoverflow.com](https://stackoverflow.com)



- Github:
  - <https://github.com>

# Programación para Android



- **Android Studio.**
  - **Incorpora un IDE propio.**
  - <https://developer.android.com/sdk/installing/studio.html>
- También se puede programar con:
  - Netbeans:
    - NBPlugin.
  - ADT de Google (Android Developer Tools).
    - Utilizando el IDE de Eclipse.

# Componentes de Android Studio

- Entorno de desarrollo (Android Studio).
- Android SDK Tools:
  - Conjunto de herramientas de desarrollo y depuración de programas.
- Android Platform Tools:
  - Herramientas para compilar y generar paquetes (apk) para el SO Android.
- La imagen del SO Android.

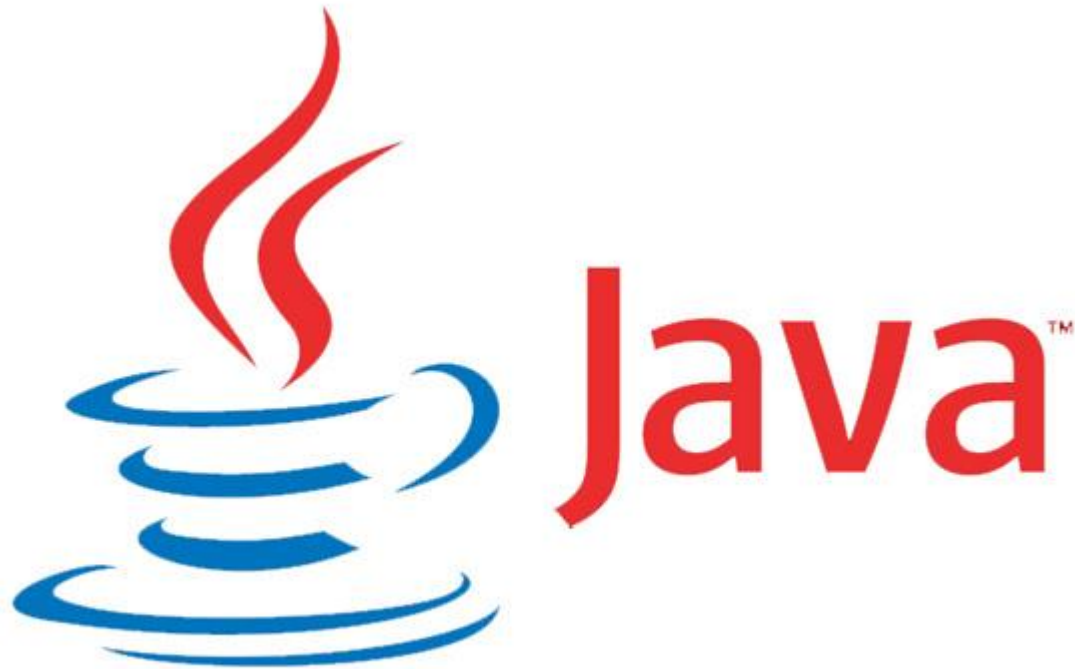


# Cosas a tener en cuenta

- Un dispositivo Android no es un PC:
  - Dispositivo portable y pequeño.
  - Capacidad de procesamiento limitada.
  - Pantallas pequeñas.
  - Teclados pequeños o virtuales.
  - Pantallas LCD “multitouch”.
  - Conexión a redes limitadas y de ancho de banda pequeño (4G, Internet...).

# Prerrequisitos

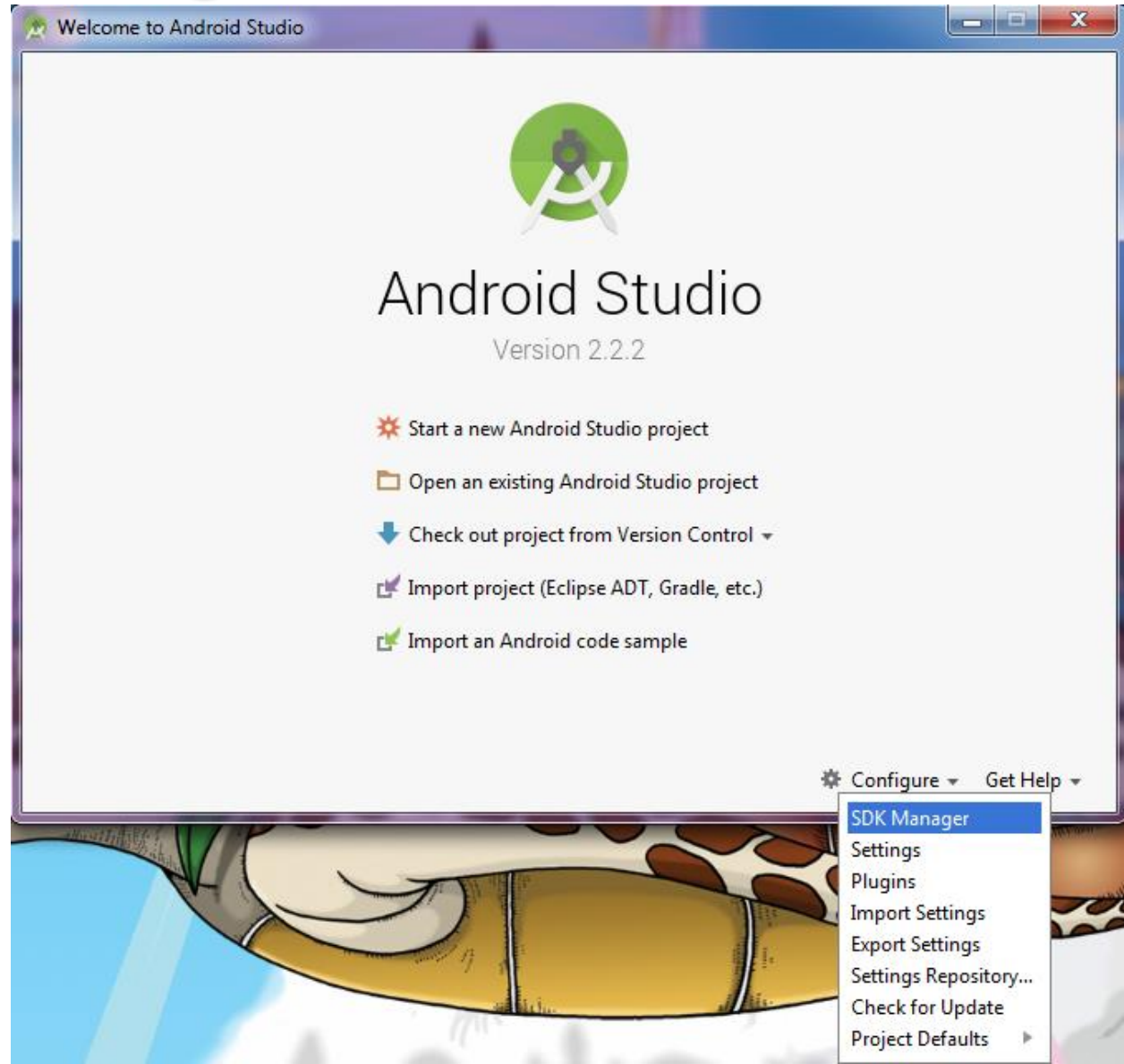
- JDK de Java.
  - <http://oracle.com/technetwork/java/javase/downloads/index.html>
  - O buscando en google Java JDK si el link anterior no funciona.



# Configuración Android Studio

- En la pantalla de bienvenida tenemos la opción “Configure”.
- Si pulsamos en SDK Manager podremos descargar diversos componentes necesarios para compilar, probar y depurar nuestros proyectos. En principio en esta versión de Android Studio se ha descargado e instalado todo lo necesario desde el principio.
- En caso de necesitar instalar componentes utilizaremos esta opción.

# Configuración Android Studio




# Primer Proyecto

- En la pantalla de bienvenida pulsamos en “Start a New Android Studio Project”.
- Ponemos el nombre de la aplicación incluyendo el dominio de la compañía.
- Indicamos la ubicación donde se almacenará en el disco duro.
- Pulsamos “Next” y seleccionamos el tipo de plataforma para nuestra app.
- Escogemos el mínimo SDK con el que funcionará nuestra app.

# Primer Proyecto

Create New Project

 Target Android Devices

**Select the form factors your app will run on**

Different platforms may require separate SDKs

☒ Phone and Tablet

Minimum SDK

Lower API levels target more devices, but have fewer features available.  
By targeting API 16 and later, your app will run on approximately **95,2%** of the devices that are active on the Google Play Store.  
[Help me choose](#)

☐ Wear

Minimum SDK

☐ TV

Minimum SDK

☐ Android Auto

☐ Glass

Minimum SDK

Previous Next Cancel Finish

# Primer Proyecto

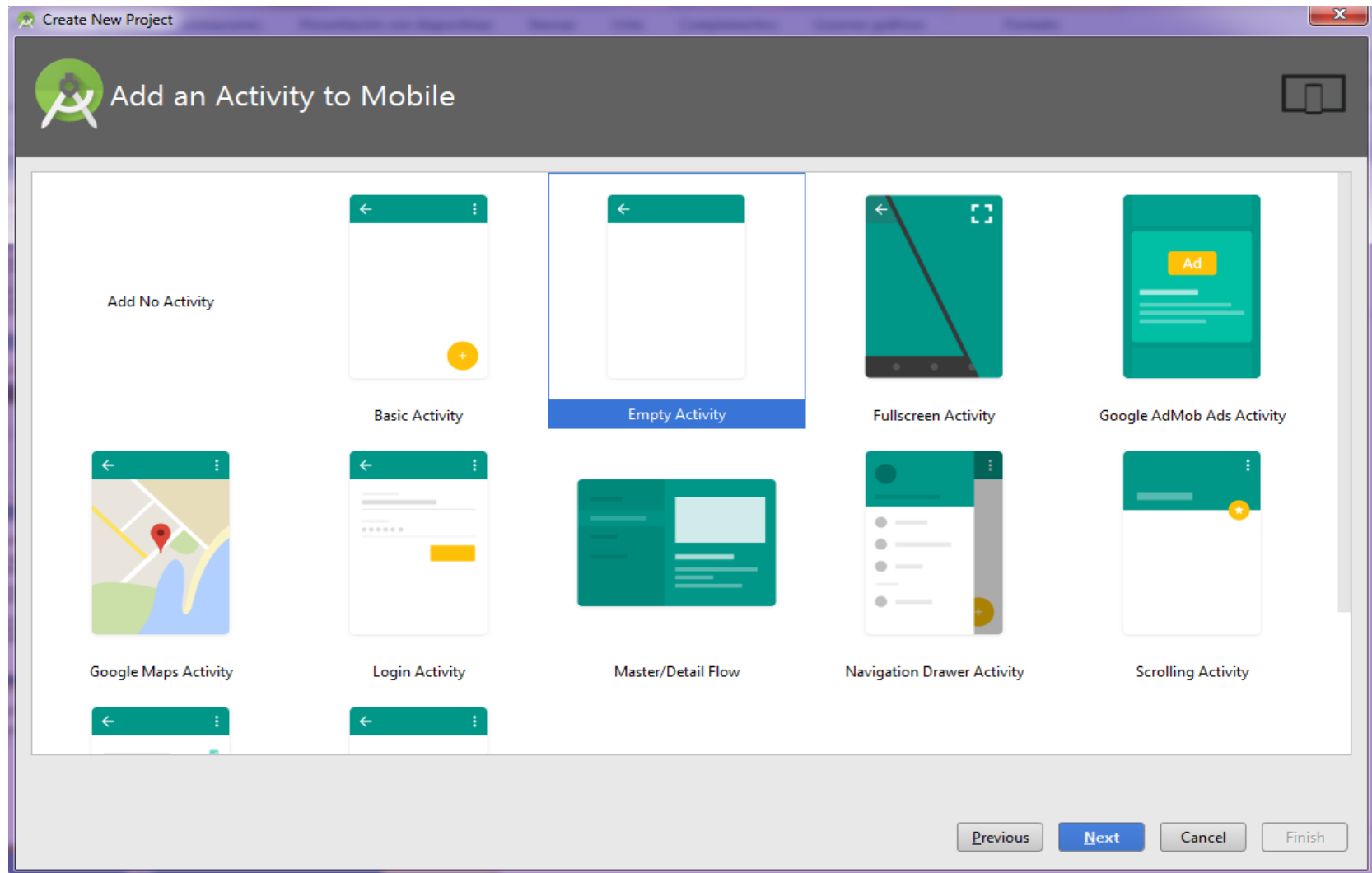
- Al pulsar “Next” aparecerá otra pantalla pidiendo que seleccionemos el tipo de actividad.
- Una actividad es nuestro programa en sí mismo, contiene la interfaz de usuario de nuestra App.
- Una actividad es un programa pequeño y ligero, controlado por Android y sometido a las normas de funcionamiento de Android.

# Primer Proyecto

- Escogemos una actividad en blanco “Empty Activity”.
- Pulsamos “Next”, elegimos el nombre de nuestra actividad y pulsamos “Finish”.





# Primer Proyecto




# Primer Proyecto

Create New Project

 Customize the Activity 

Creates a new empty activity



Empty Activity

Activity Name:

☒ Generate Layout File

Layout Name:

☒ Backwards Compatibility (AppCompat)

The name of the activity class to create

[Previous](#) [Next](#) [Cancel](#) [Finish](#)

# Primer Proyecto

- Una vez creado el proyecto saldrá una ventana con consejos que son importantes leer para familiarizarnos con el uso de la herramienta.



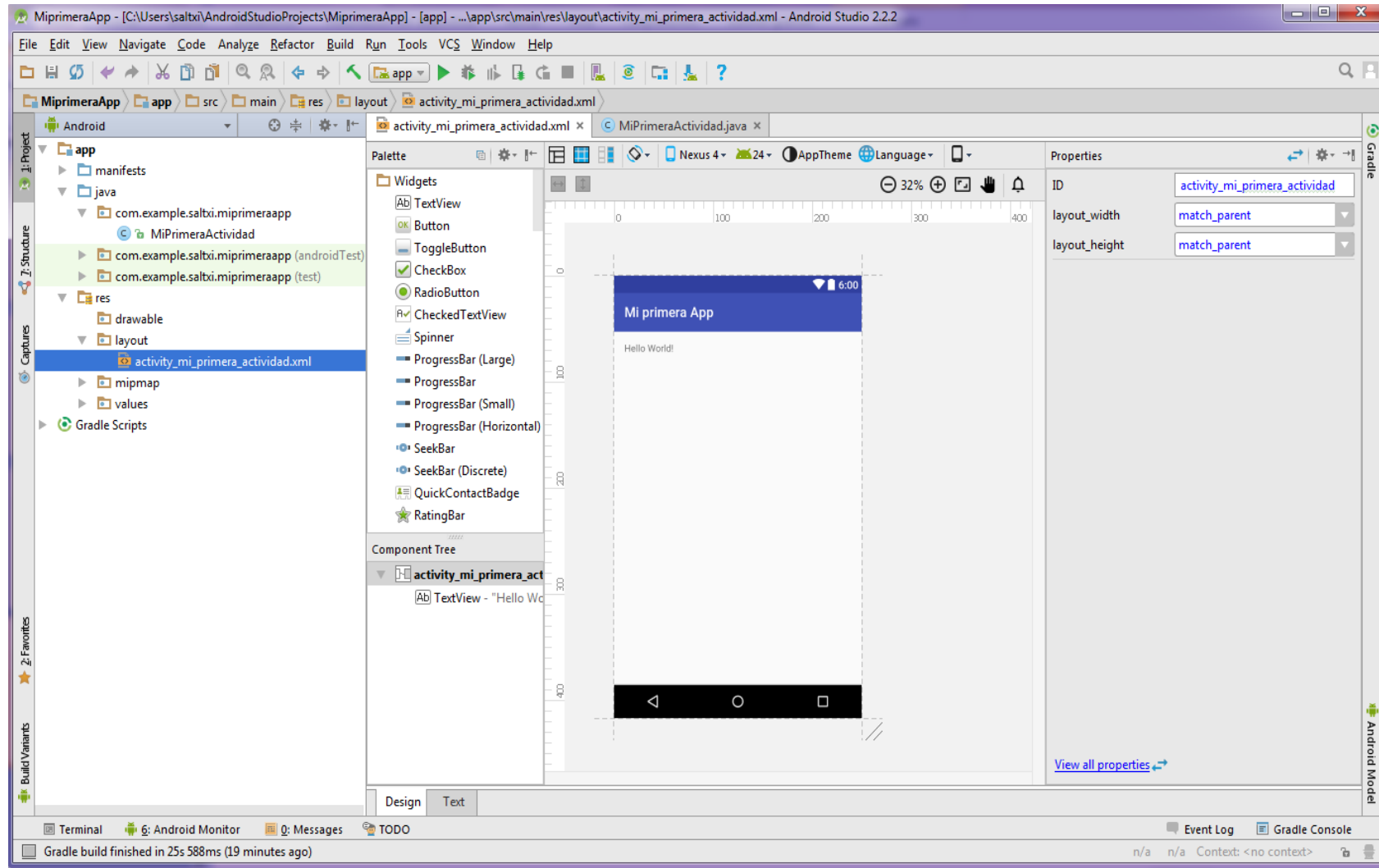
# Primer Proyecto

- Al abrir el proyecto a la izquierda tenemos la pestaña que abre el explorador de proyectos:
  - Aquí aparecen todas las carpetas y ficheros que componen el proyecto.
  - La carpeta java contiene los ficheros que modificaremos para dar vida a la aplicación.
  - La carpeta gradle es el plugin que utiliza android studio para compilar, construir y depurar nuestros programas.
  - El Android Manifest.xml contiene la descripción de nuestra aplicación y qué componentes están incluidos (servicios, actividades, imágenes...).

# Primer Proyecto

- En el explorador de proyectos vamos a la carpeta “res→layout” y hacemos doble click sobre el fichero xml que hay dentro.
- Esto nos abrirá en la parte central una ventana en la que tendremos una visión del diseño de nuestra aplicación y de todos los componentes o widgets que podemos ir insertando para configurar la interfaz gráfica.

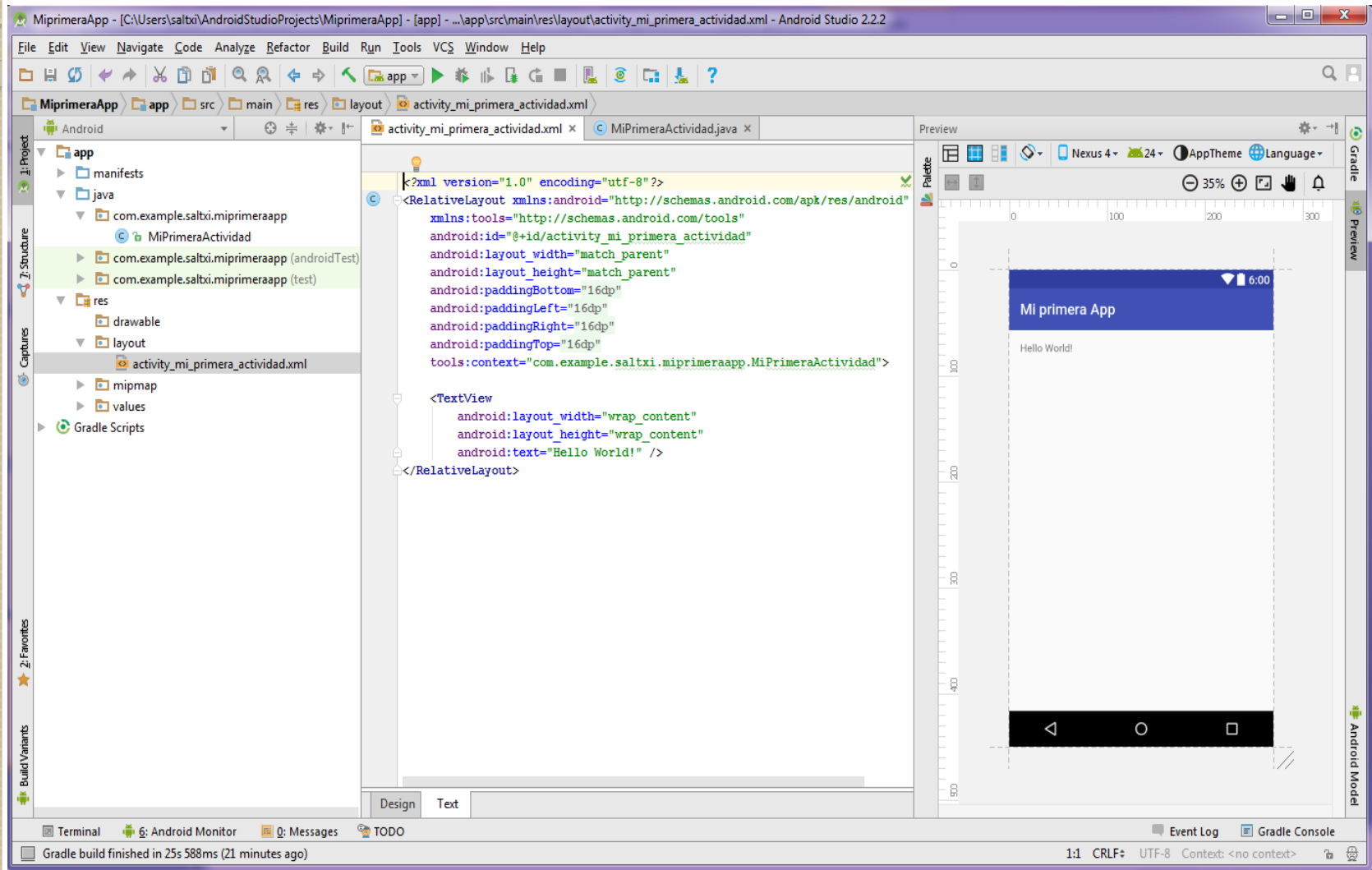
# Primer Proyecto



# Primer Proyecto

- En la parte inferior tenemos dos pestañas “Design” (es en la que estamos por defecto) y “Text”.
- En la pestaña “Text” tenemos el código xml autogenerado que define la interfaz gráfica de la aplicación.

# Primer Proyecto



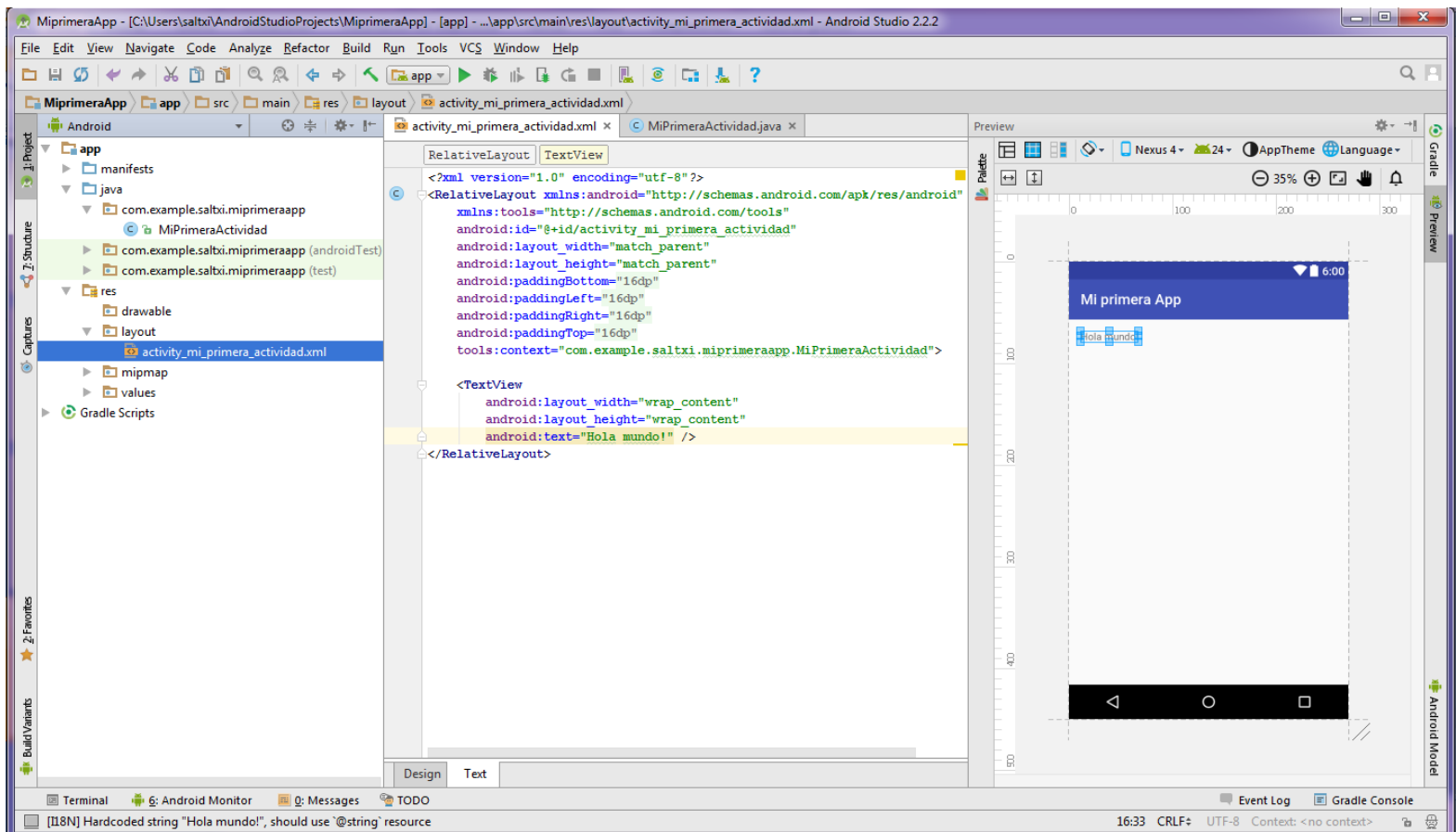


# Primer Proyecto

- En la carpeta java podemos abrir el fichero MiPrimeraActividad.java.
- En la parte superior tendremos dos pestañas, la del fichero xml y la del fichero java.
- En xml se declaran todos los componentes.
- En el fichero java se programan los comportamientos.

# Primer Proyecto

- Al modificar el fichero xml podemos ver a la derecha una vista previa de cómo



# Desarrollo sin Código

- En la vista diseño podemos arrastrar widgets a la pantalla de nuestra aplicación y el código del fichero xml se modifica automáticamente para reflejar los cambios.

# Desarrollo sin Código

- Insertamos un botón debajo del “TextView” y hacemos click sobre él para ver y editar sus propiedades.
- Podemos cambiar el texto del botón y el id que usaremos para referenciarlo desde el código.
- Esto funciona con todos los componentes que agreguemos a nuestra aplicación.

# Desarrollo con Código

- Implementar la interfaz OnClickListener.
- Registrar el objeto que implementa la interfaz mediante el método `setOnClickListener`.
- Programar un método llamado `OnClick` que hace de `Callback`.

# Desarrollo con Código

- Para poder referenciar a las clases de los componentes incluidos en el XML tenemos que importar las clases en nuestro código.
- Imports para el TextView y el Button:
  - `import android.widget.TextView;`
  - `import android.widget.Button;`

# Desarrollo con Código

- Creamos una referencia al objeto de la clase que queremos y llamamos a la función `findViewById(...)`.

*Button miBoton;*

*miBoton = (Button) findViewById(R.id.button);*

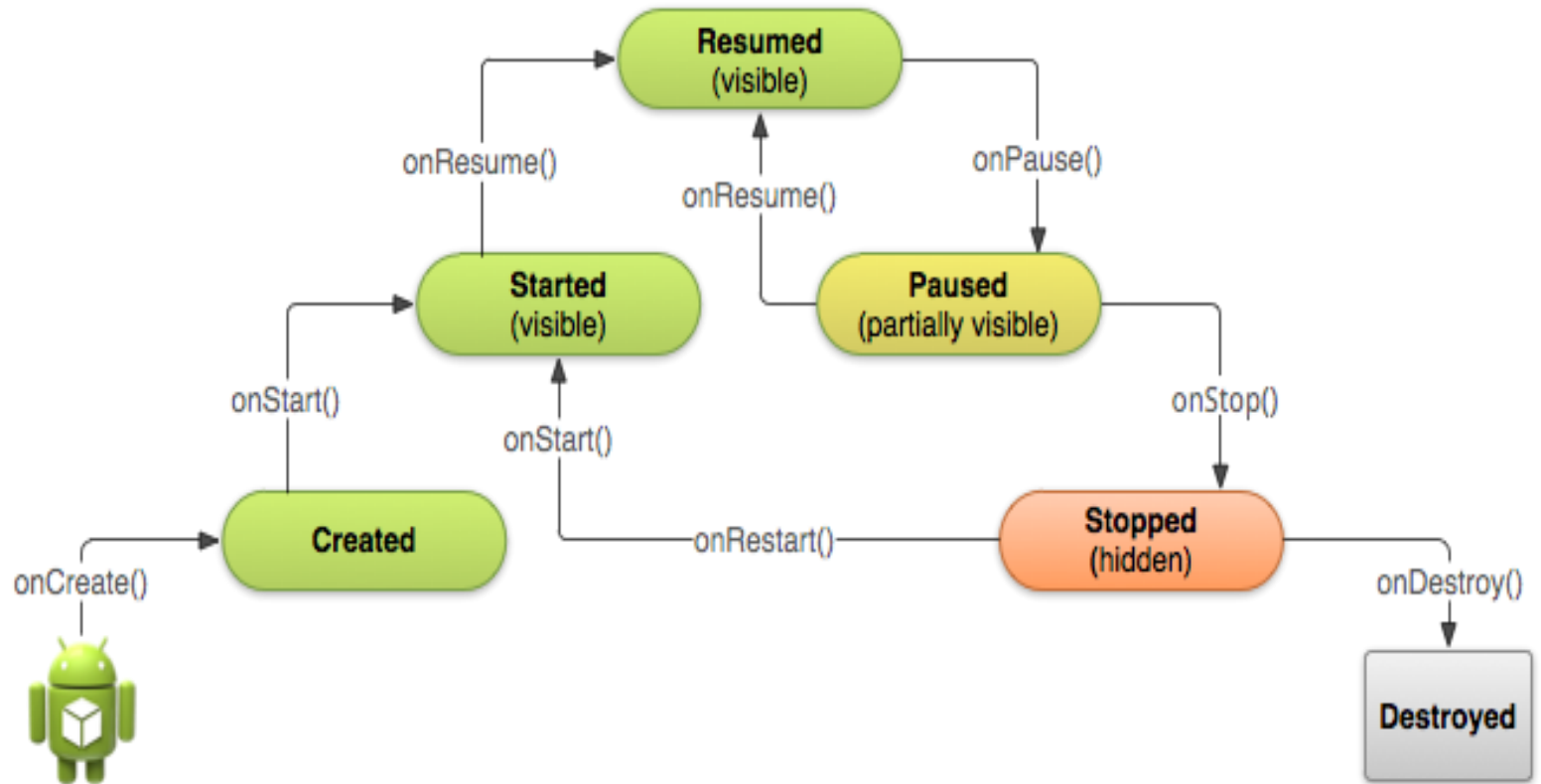
- A partir de aquí podemos acceder a múltiples propiedades y métodos para programar el botón.

# Desarrollo con Código

- Al programar para Android no tenemos función main.
- Cada actividad tiene un ciclo de vida, va sucediendo llamadas a funciones callback según la actividad experimenta interacciones con el usuario.



# Desarrollo con Código



# Desarrollo con Código

- De esta manera aseguramos que nuestra app está adaptada a un dispositivo móvil:
  - Se bloquea o deja de funcionar cuando el usuario recibe una llamada o pasa a otra app.
  - No consume recursos valiosos del sistema cuando el usuario no está usándola activamente.
  - No se pierde el progreso del usuario si abandonan la app y luego vuelven a ella.
  - No se bloquea cuando el usuario cambia la posición de la pantalla...

# Desarrollo con Código

- Al programar no es necesario implementar todas las funciones callback.
- La primera acción que el ciclo de vida ejecuta cuando el SO arranca la app es la función callback “onCreate”.
- Primero añadimos la implementación de la clase View.OnClickListener.
- Después añadimos el código para acceder a los widgets que hemos agregado en el XML.
- A continuación añadimos en el código de la función onCreate el código para poder referenciar a los componentes textView y button, y registramos el listener “OnClick”.

# Desarrollo con Código

- “MiPrimeraActivity.java” quedaría de la siguiente manera:

```
public class MiPrimeraActivity extends ActionBarActivity implements View.OnClickListener{
```

```
    Button miBoton; //Referencias a los widgets añadidos
```

```
    TextView miTexto;
```

```
@Override
```

```
protected void onCreate(Bundle savedInstanceState){
```

```
    super.onCreate(savedInstanceState);
```

```
    setContentView(R.layout.activity_mi_primera);
```

```
        miBoton = (Button) findViewById(R.id.button);
```

```
        miBoton.setOnClickListener(this);
```

```
}
```

```
Public void onClick(View view){
```

```
    //responde al evento Click
```

```
    miTexto = (TextView) findViewById(R.id.textView);
```

```
    miTexto.setText(“pulsado”);
```

```
}}
```

# Desarrollo con Código

- Compilamos la app:
  - Build → Make Project
- Ejecutamos la app:
  - Run → Run 'app'
- Como no estamos ejecutando el emulador tendremos que crear el emulador.

# Desarrollo con Código

- Como no estamos ejecutando el emulador tendremos que crear el emulador.
- Para ello sacamos el Android Virtual Device Manager (AVD Manager):
  - Tools → Android → AVD Manager.
- Podemos crear tantos dispositivos virtuales como queramos, pero de momento vamos a crear uno.
- Por ejemplo elegimos un emulador de Nexus One, con pantalla pequeña, con Nougat.
- La CPU es Intel Atom (x86).
- El emulador es lento y consume muchos recursos.

# Desarrollo con Código

- Compilamos la app:
  - Build → Make Project
- Ejecutamos la app:
  - Run → Run 'app'
- Ahora al ejecutar podemos seleccionar el dispositivo creado.

# Desarrollo con Código

## Cosas a tener en cuenta

- Para conseguir el acceso a un widget y poder operar con él tenemos que hacerlo después de la instrucción `setContentView(R.layout.activity_mi_primera)`.
- Si no se hace así el resultado de la llamada `findViewById()` será nulo y no se podrá acceder al widget.
- Debemos implementar la interfaz para responder mediante callback al evento deseado.
- Debemos registrar la función callback.
- Debemos programar la función en sí.