

INTRODUCCIÓN



paco@portadaalta.es

Índice

- **Entorno de desarrollo**
- **Sistema de control de versiones: Git**
- **Estructura de un proyecto Android**
- **Componentes de una aplicación Android**
- **Java para desarrollo Android**
- **Ejemplos:**
 - ✓ **Mi primera aplicación**
 - ✓ **Divisas**
 - ✓ **2 actividades**
 - ✓ **Contador de cafés**

Entorno de desarrollo

Vamos a describir los pasos básicos para disponer en nuestro PC del entorno y las herramientas necesarias para comenzar a programar aplicaciones para Android.

Paso 1. Descarga e instalación de Java.

Si aún no dispones de ninguna versión del JDK (Java Development Kit) en tu equipo, puedes instalarlo de varias formas:

- Descargar la última versión desde la web de [Oracle](#) e instalarla.
- Instalar JDK desde la consola:

```
sudo apt install default-jdk
```

- Instalar openjdk desde el gestor de paquetes

Entorno de desarrollo

Gestor de paquetes Synaptic

Archivo Editar Paquete Configuración Ayuda

Recargar Aplicar Propiedades Filtro rápido Buscar

jdk

Todo	E	Paquete	Versión instalada	Última versión	Descripción
Administración del sistema	<input type="checkbox"/>	gcj-native-helper		2:1.7-51	Herramientas de ayuda estándar
Administración del sistema	<input type="checkbox"/>	openjdk-6-jdk		6b36-1.13.8-0ubuntu1	OpenJDK Development Kit (JDK 6)
Administración del sistema	<input checked="" type="checkbox"/>	openjdk-7-jdk	7u79-2.5.6-0ubuntu1	7u79-2.5.6-0ubuntu1	Conjunto de desarrollo OpenJDK 7
Administración del sistema	<input type="checkbox"/>	default-jdk		2:1.7-51	Kit de desarrollo de Java estándar
Autoría de TeX	<input type="checkbox"/>	openjdk-7-doc		7u79-2.5.6-0ubuntu1	OpenJDK Development Kit (JDK 7)
Autoría de TeX (multiverse)	<input type="checkbox"/>	libpostgresql-jdbc-java		9.2-1002-1	Java database (JDBC) driver for PostgreSQL
Autoría de TeX (universe)	<input type="checkbox"/>	openjdk-6-doc		6b36-1.13.8-0ubuntu1	OpenJDK Development Kit (JDK 6)

Entorno de desarrollo

¿Cómo instalar Java con Apt-Get en Ubuntu 16.04?



Entorno de desarrollo

Paso 2. Descargar e instalar Android Studio.

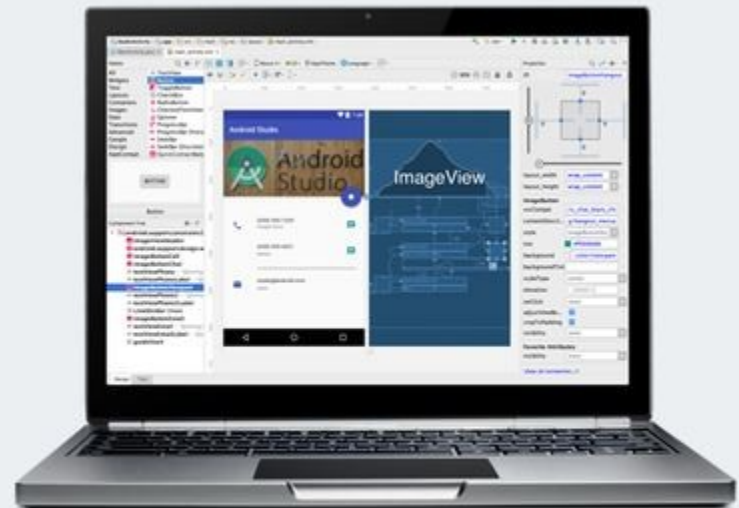
Android Studio

The Official IDE for Android

Android Studio provides the fastest tools for building apps on every type of Android device.

World-class code editing, debugging, performance tooling, a flexible build system, and an instant build/deploy system all allow you to focus on building unique and high quality apps.

DOWNLOAD ANDROID STUDIO
2.3.3 FOR LINUX (468 MB)

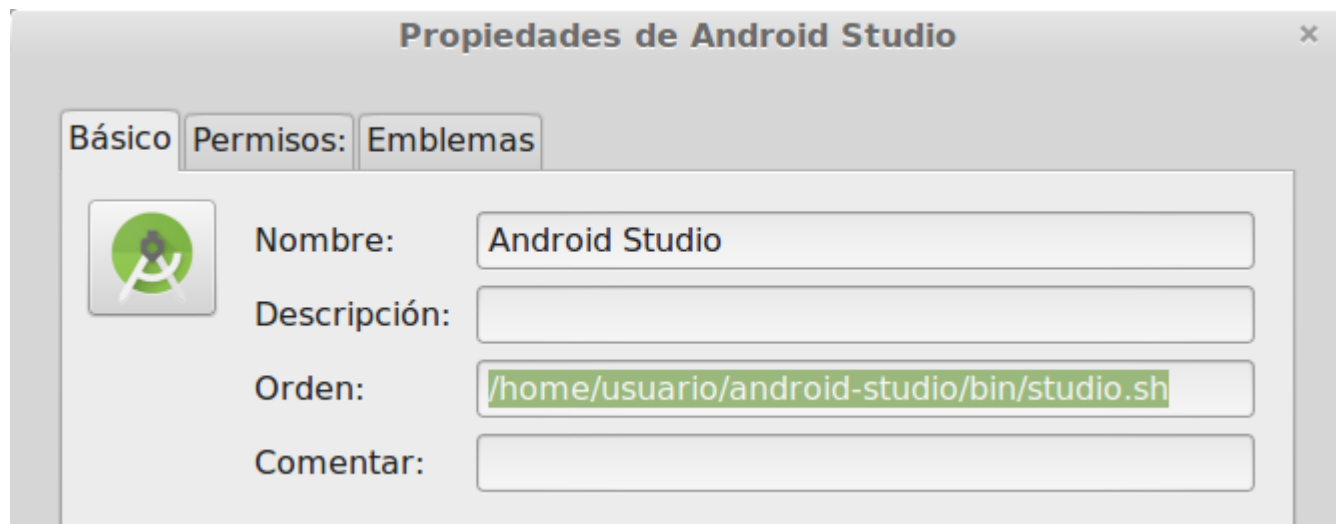


Entorno de desarrollo

Una vez descargado, se descomprime el fichero zip y se ejecuta en consola el programa **studio.sh**:


```
paco@Latitude ~$ cd android-studio/bin  
paco@Latitude ~/android-studio/bin $ ./studio.sh
```

Además, se puede crear un acceso directo en el escritorio al programa **studio.sh**:



Entorno de desarrollo

Android Studio Setup Wizard

 SDK Components Setup

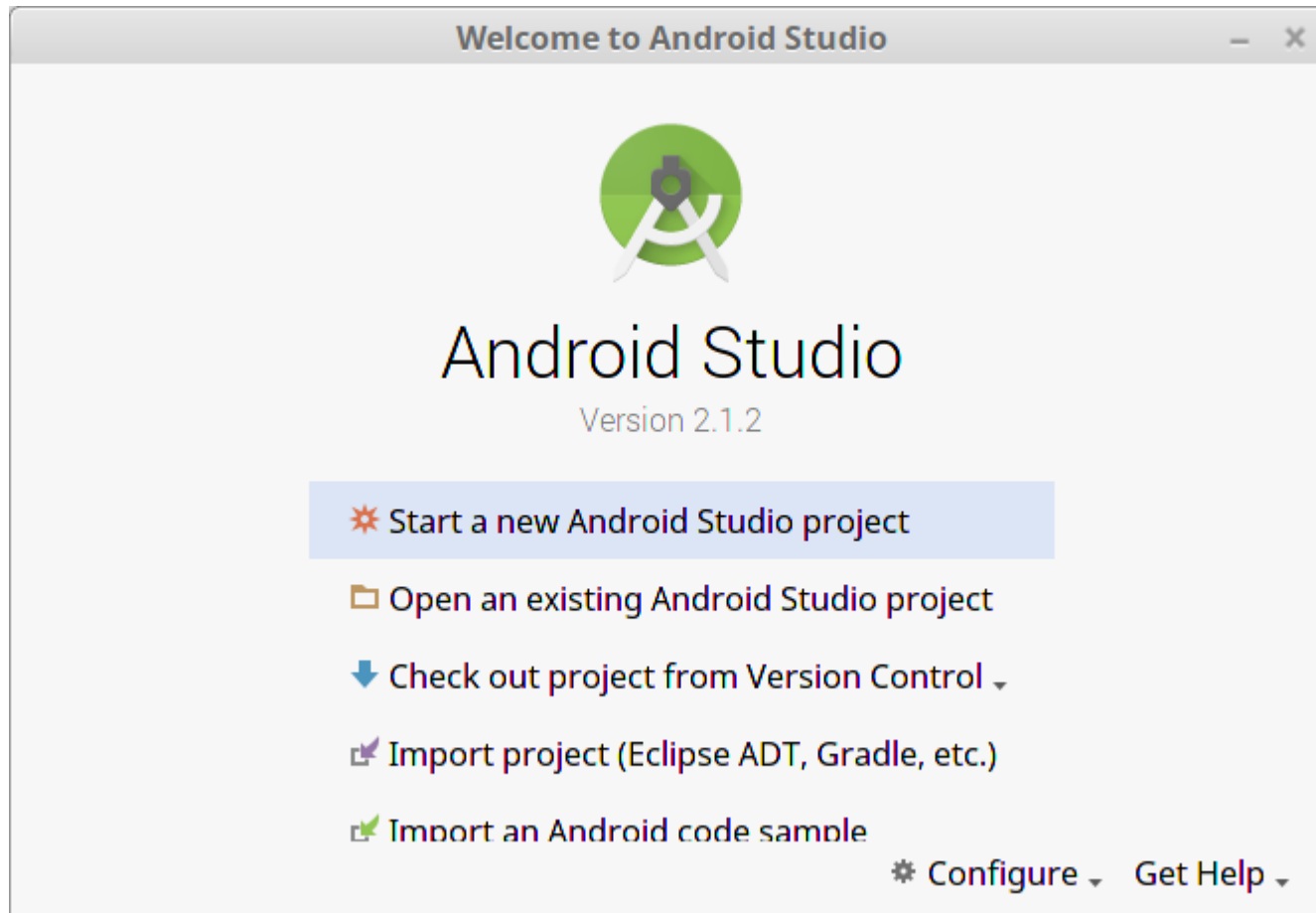
Check the components you want to update/install. Click Next to continue.

<input checked="" type="checkbox"/> Android SDK - (265 MB)	<p>A preconfigured and optimized Android Virtual Device for app testing on the emulator. (Recommended)</p>
<input checked="" type="checkbox"/> Android SDK Platform	
<input checked="" type="checkbox"/> API 24: Android 6.X (N) - (78,8 MB)	
<input checked="" type="checkbox"/> Android Virtual Device - (1 GB)	

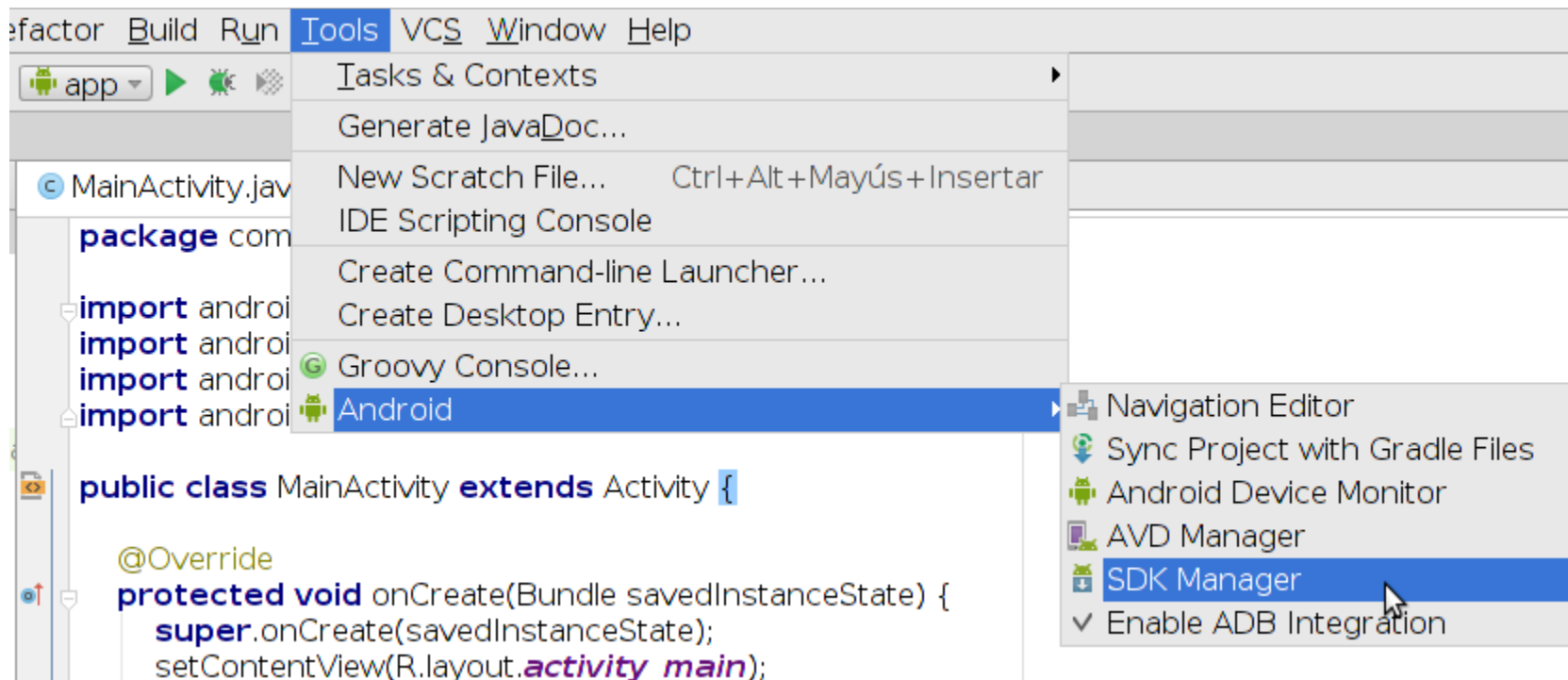
Android SDK Location:

Total disk space required: 1,34 GB
Available disk space: 166 GB

Entorno de desarrollo



Entorno de desarrollo



Entorno de desarrollo

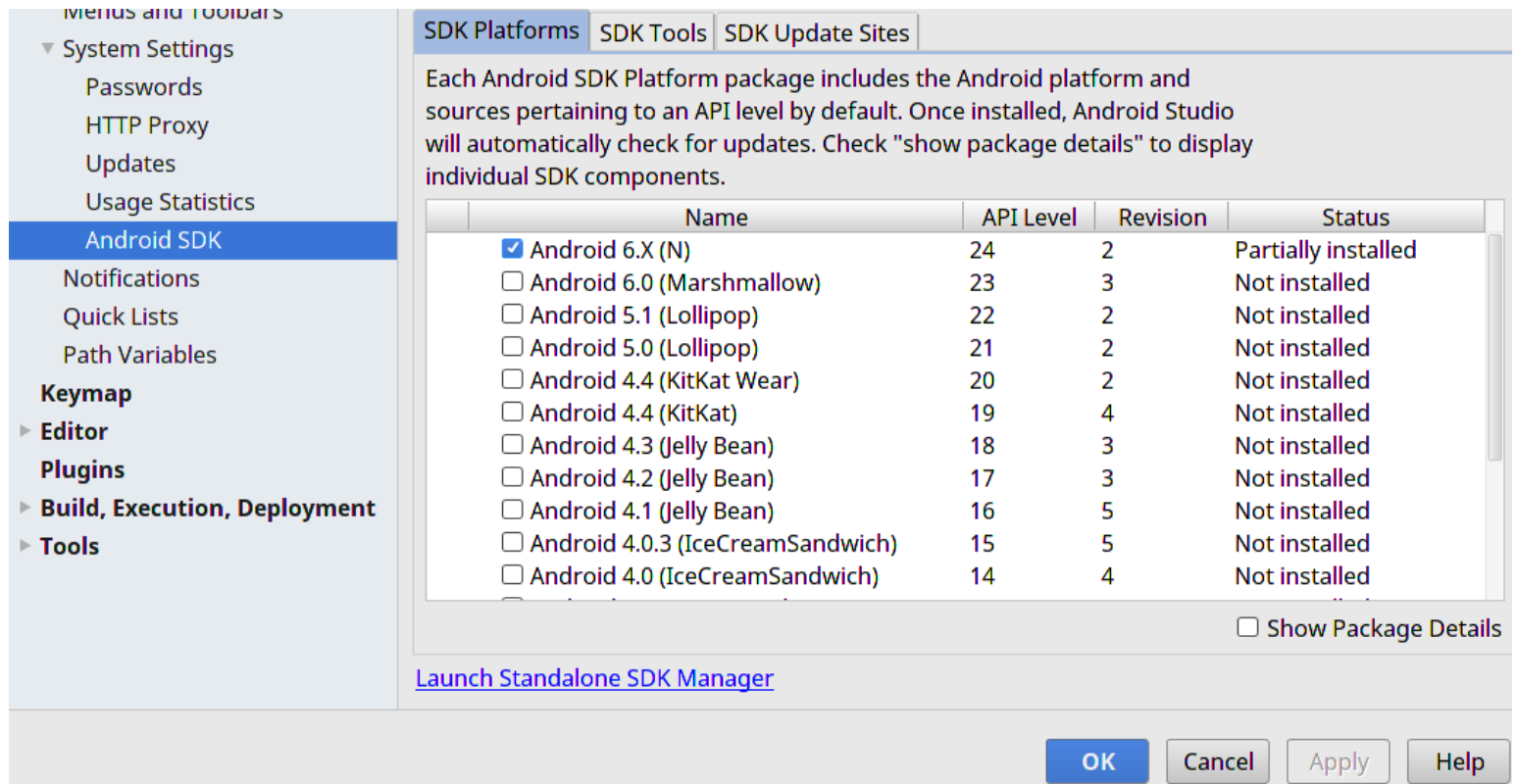
Paso 3. Configurar el SDK.

Understanding Android API Levels

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
2.3 Gingerbread	10	
4.0 Ice Cream Sandwich	15	97,4%
4.1 Jelly Bean	16	95,2%
4.2 Jelly Bean	17	87,4%
4.3 Jelly Bean	18	76,9%
4.4 KitKat	19	73,9%
5.0 Lollipop	21	40,5%
5.1 Lollipop	22	24,1%
6.0 Marshmallow	23	4,7%























Entorno de desarrollo

Instalar Android 5.1 (API 22) y Android 4.1 (API 16):



Entorno de desarrollo

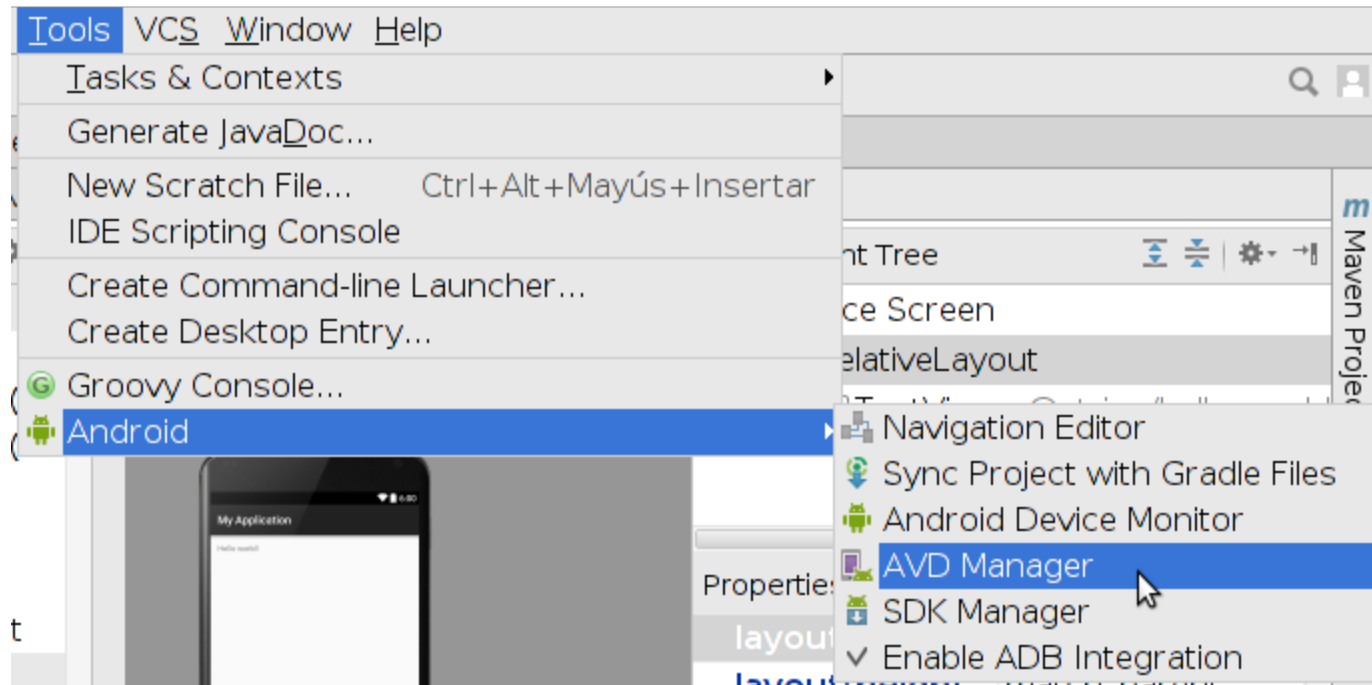
Quitar API 24 e
instalar API 22 y API 16

Packages			
 Name	API	Rev.	Status
▶ <input type="checkbox"/>  Android 7.0 (API 24)			
▶ <input type="checkbox"/>  Android 6.0 (API 23)			
▼ <input type="checkbox"/>  Android 5.1.1 (API 22)			
<input checked="" type="checkbox"/>  SDK Platform	22	2	 Installed
<input type="checkbox"/>  Android TV ARM EABI v7a System Image	22	1	<input type="checkbox"/> Not installed
<input type="checkbox"/>  Android TV Intel x86 Atom System Image	22	3	<input type="checkbox"/> Not installed
<input type="checkbox"/>  Android Wear ARM EABI v7a System Image	22	7	<input type="checkbox"/> Not installed
<input type="checkbox"/>  Android Wear Intel x86 Atom System Image	22	7	<input type="checkbox"/> Not installed
<input type="checkbox"/>  ARM EABI v7a System Image	22	1	<input type="checkbox"/> Not installed
<input checked="" type="checkbox"/>  Intel x86 Atom_64 System Image	22	5	 Installed
<input type="checkbox"/>  Intel x86 Atom System Image	22	5	<input type="checkbox"/> Not installed
<input type="checkbox"/>  Google APIs ARM EABI v7a System Image	22	9	<input type="checkbox"/> Not installed
<input type="checkbox"/>  Google APIs Intel x86 Atom_64 System Image	22	9	<input type="checkbox"/> Not installed
<input type="checkbox"/>  Google APIs Intel x86 Atom System Image	22	9	<input type="checkbox"/> Not installed
<input type="checkbox"/>  Google APIs	22	1	 Installed
<input checked="" type="checkbox"/>  Google APIs Intel x86 Atom_64 System Image	22	9	 Installed
<input type="checkbox"/>  Sources for Android SDK	22	1	<input type="checkbox"/> Not installed

Entorno de desarrollo

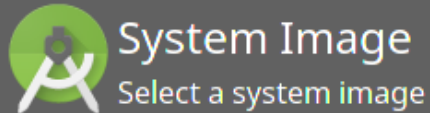
Paso 4. Configurar el AVD.

Instalar Nexus 4 con API 22



Entorno de desarrollo

Virtual Device Configuration



Recommended x86 Images Other Images

Release Name	API Level ▾	ABI	Target
<i>Marshmallow</i> Download	23	x86_64	<i>Android 6.0 (with Google APIs)</i>
<i>Marshmallow</i> Download	23	x86	<i>Android 6.0 (with Google APIs)</i>
Lollipop	22	x86_64	Android 5.1 (with Google APIs)
<i>Lollipop</i> Download	22	x86	<i>Android 5.1 (with Google APIs)</i>
<i>KitKat</i> Download	19	x86	<i>Android 4.4 (with Google APIs)</i>

Lollipop



API Level

22

Android





5.1

Google Inc.

System Image

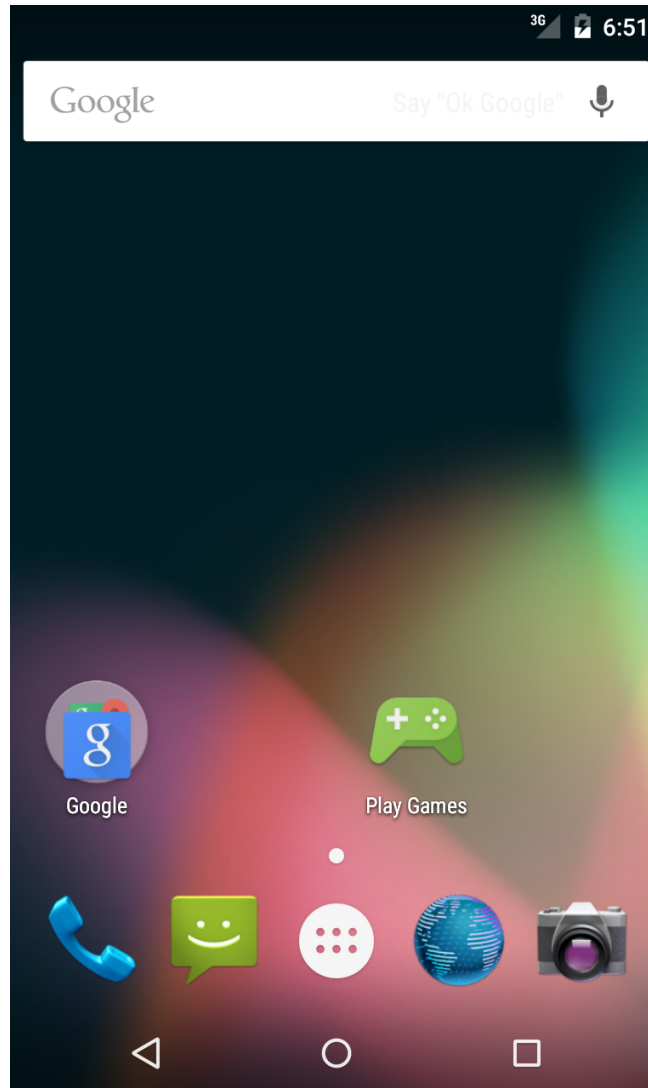
x86_64

Entorno de desarrollo

AVD Name	<input type="text" value="Nexus 4 API 22"/>		
 Nexus 4	4,7" 768x1280 xhdpi	<button>Change...</button>	
 Lollipop	Android 5.1 x86_64	<button>Change...</button>	
Startup size and orientation	Scale:	<input type="text" value="Auto"/>	
	Orientation:	<div><div> Portrait</div><div> Landscape</div></div>	
Emulated Performance	Graphics:	<input type="text" value="Software - GLES 2.0"/>	
Device Frame	<input type="checkbox"/> Enable Device Frame		

Si hay algún error al lanzar el emulador, se puede configurar Graphics en Software.

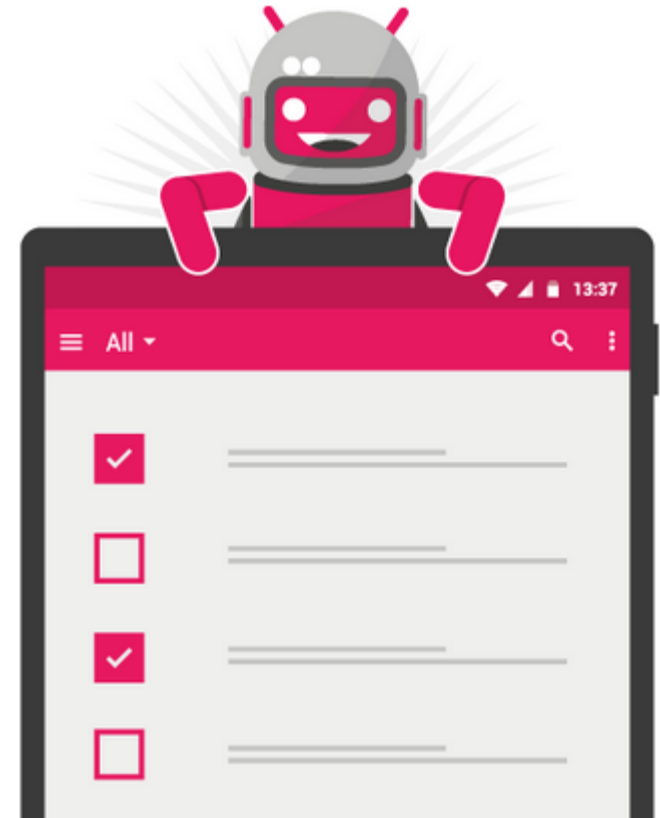
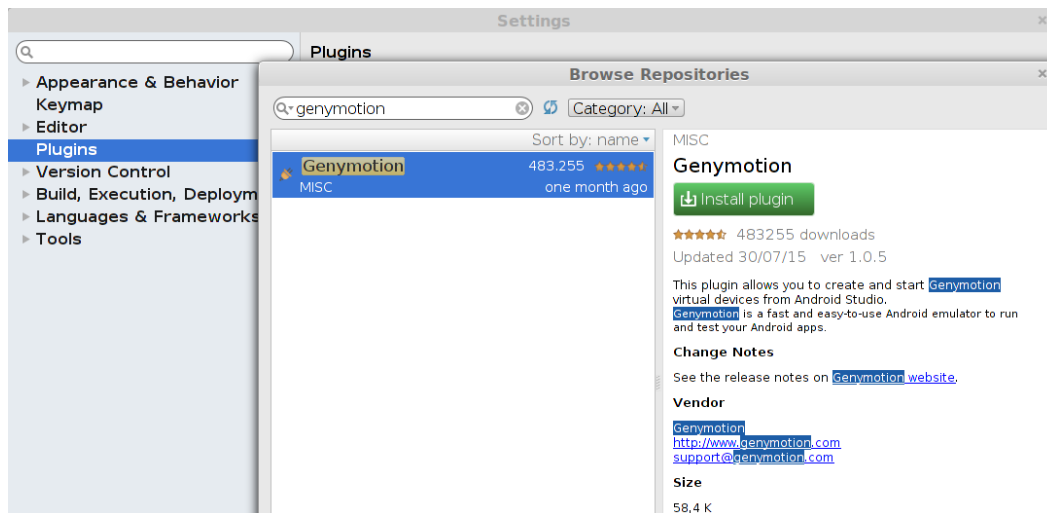
Entorno de desarrollo



Entorno de desarrollo

Otro emulador: Genymotion

Genymotion plugin for Android Studio



Entorno de desarrollo

Kobiton



Test on real iOS and Android devices, because life doesn't happen in a simulator



Plug in and manage internal devices anytime from anywhere



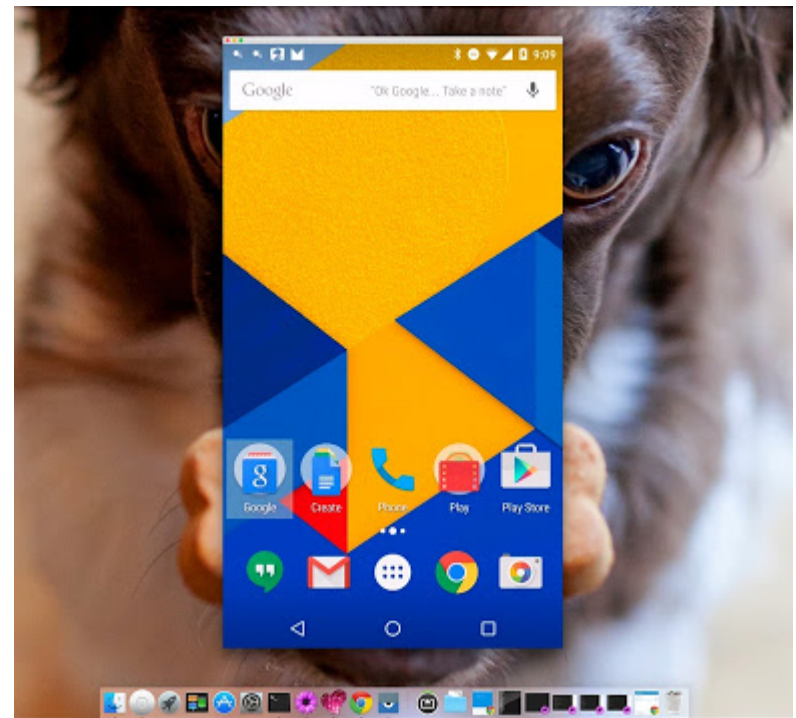
Boost collaboration and accelerate problem solving by sharing results with your team



Save time and money by managing internal and external devices in one place

Entorno de desarrollo

Vysor: A window to your Android
(Vysor puts your Android on your desktop)
Extensión para Chrome



Configurar el móvil para depurar con Android Studio

Entorno de desarrollo

Screen Cast - View mobile on PC

The easist way to show your mobile screen on your PC, Mac, Tablet or Smart TV !

SCREEN CAST



7 Ways To Mirror Your Android Screen to Any Computer

Entorno de desarrollo

Depuración:

developer.android.com/studio/debug/index.html

Android Studio includes a debugger that allows you to debug apps running on the Android Emulator or a connected Android device.

With the Android Studio debugger, you can:

- Select a device to debug your app on.
- Set breakpoints in your code.
- Examine variables and evaluate expressions at runtime.
- Capture screenshots and videos of your app.

Entorno de desarrollo

The screenshot displays the Android Studio IDE interface. The top toolbar shows various development tools. The left sidebar contains the Project view, showing the file structure of the 'beep_beep_beep' project, including folders like .idea, assets, bin, gen, res, and src, and files like AndroidManifest.xml, beep_beep.zip, beep_beep_beep.iml, proguard.cfg, and project.properties. The main editor window shows the Java code for 'Main.java', which extends 'Activity' and implements 'AnimationListener'. The code includes methods for 'onCreate' and 'onStop', both annotated with '@Override'. The 'onCreate' method sets the content view to 'R.layout.main', loads an animation, and starts it. The 'onStop' method clears the animation and sets the listener to null. The bottom panel shows the Debug console, which is currently empty. The Variables tab shows the state of the application, with 'this' pointing to 'com.authorwjf.beep.Main@830015947216', 'savedInstanceState' as null, and 'mAnim' as null. The bottom status bar indicates the breakpoint reached at 'com.authorwjf.beep.Main.onCreate(Main.java:18)'.

```
public class Main extends Activity implements AnimationListener {  
  
    private int state_machine = 0;  
    private Animation mAnim = null;  
  
    @Override  
    public void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.main);  
        mAnim = AnimationUtils.loadAnimation(this, R.anim.xform_left_to_right_begin);  
        mAnim.setAnimationListener(this);  
        ImageView img = (ImageView)findViewById(R.id.blip);  
        img.clearAnimation();  
        img.setAnimation(mAnim);  
        img.startAnimation(mAnim);  
    }  
  
    @Override  
    protected void onStop() {  
        super.onStop();  
        try {  
            ImageView img = (ImageView)findViewById(R.id.blip);  
            img.clearAnimation();  
            mAnim.setAnimationListener(null);  
        }  
    }  
}
```

Debug: beep_beep_beep

Debugger Console Logcat

Frames Threads

Variables

Watches

No watches

6: Android 4: Run 5: Debug TODO

Breakpoint reached at com.authorwjf.beep.Main.onCreate(Main.java:18)

18:1 LF UTF-8 178M of 711M

Sistema de control de versiones: Git

¿Qué es git?

Instalación:

```
$ sudo apt-get install git
```

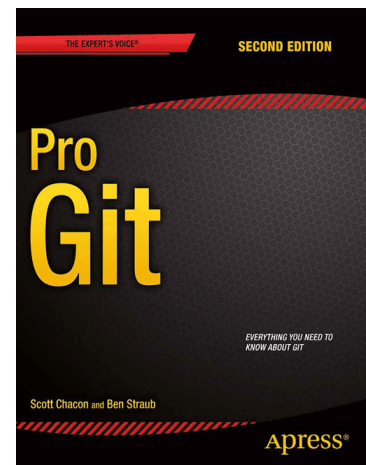
Configuración:

```
$ git config --global user.name "John Doe"
```

```
$ git config --global user.email johndoe@example.com
```

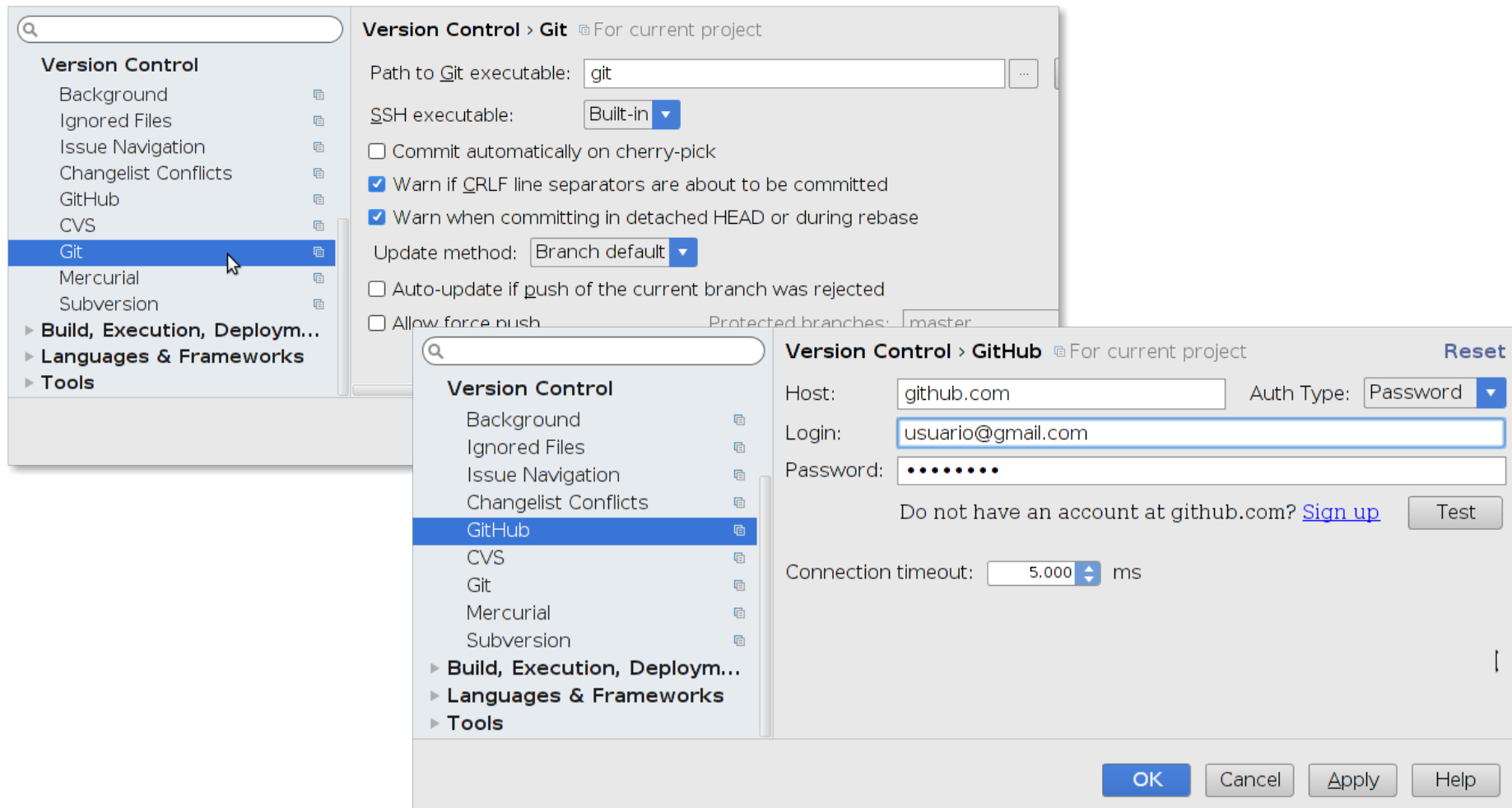
Resumen de git:

[Git Cheat Sheet](#)



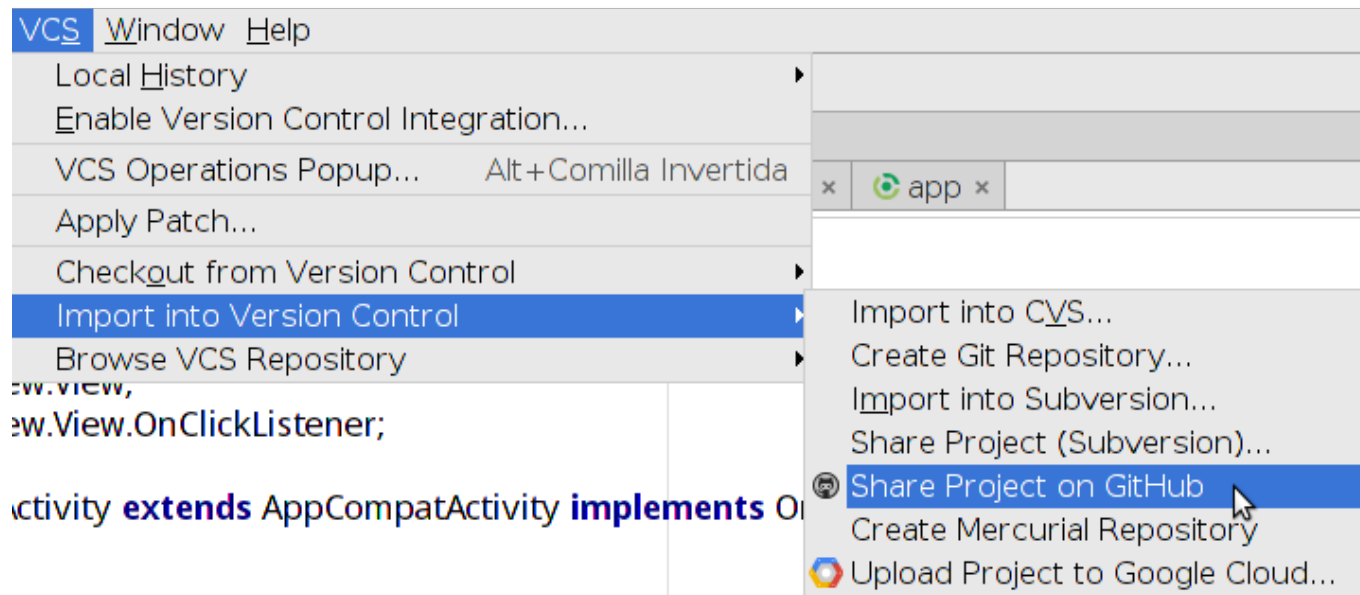
Sistema de control de versiones: Git

Configuración en Android Studio (File / Settings/ Version Control):



Sistema de control de versiones: Git

Subir un proyecto a GitHub.com:

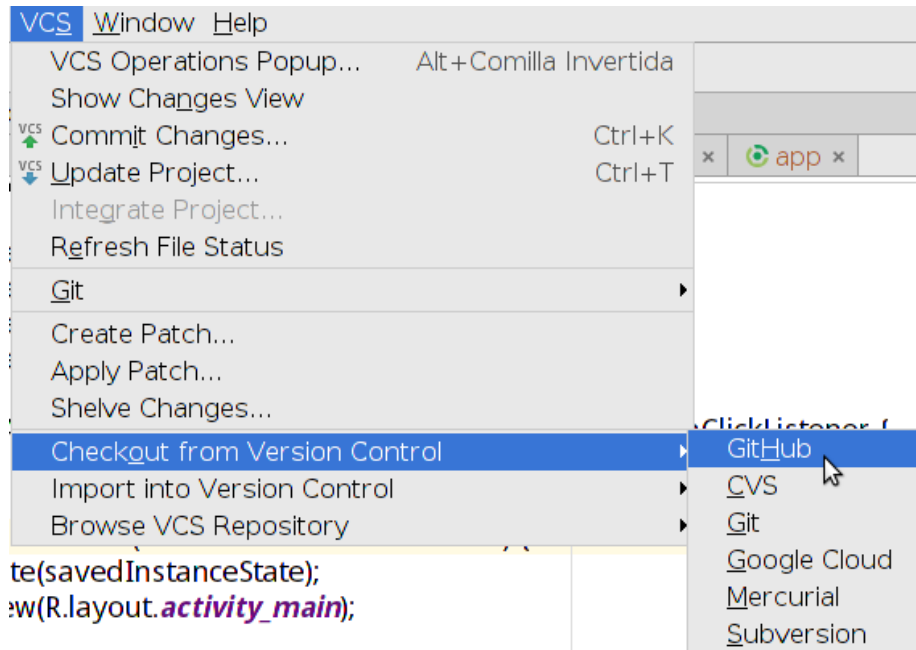


New repository name: ☐ Private

Description:

Sistema de control de versiones: Git

Clonar un proyecto desde GitHub.com:



Git Repository URL: Test

Parent Directory: ...

Directory Name:

Clone Cancel Help

Sistema de control de versiones: Git

[Bitbucket](#) es un servicio de alojamiento basado en la web para proyectos que utilizan los sistemas de control de versiones Mercurial y Git. Fue lanzado en el año 2008 por la empresa Atlassian Software.

Bitbucket ofrece cuentas gratuitas y comerciales. Las gratuitas cuentan con número ilimitado de repositorios privados y cinco usuarios.

Free for small teams. Priced to scale.

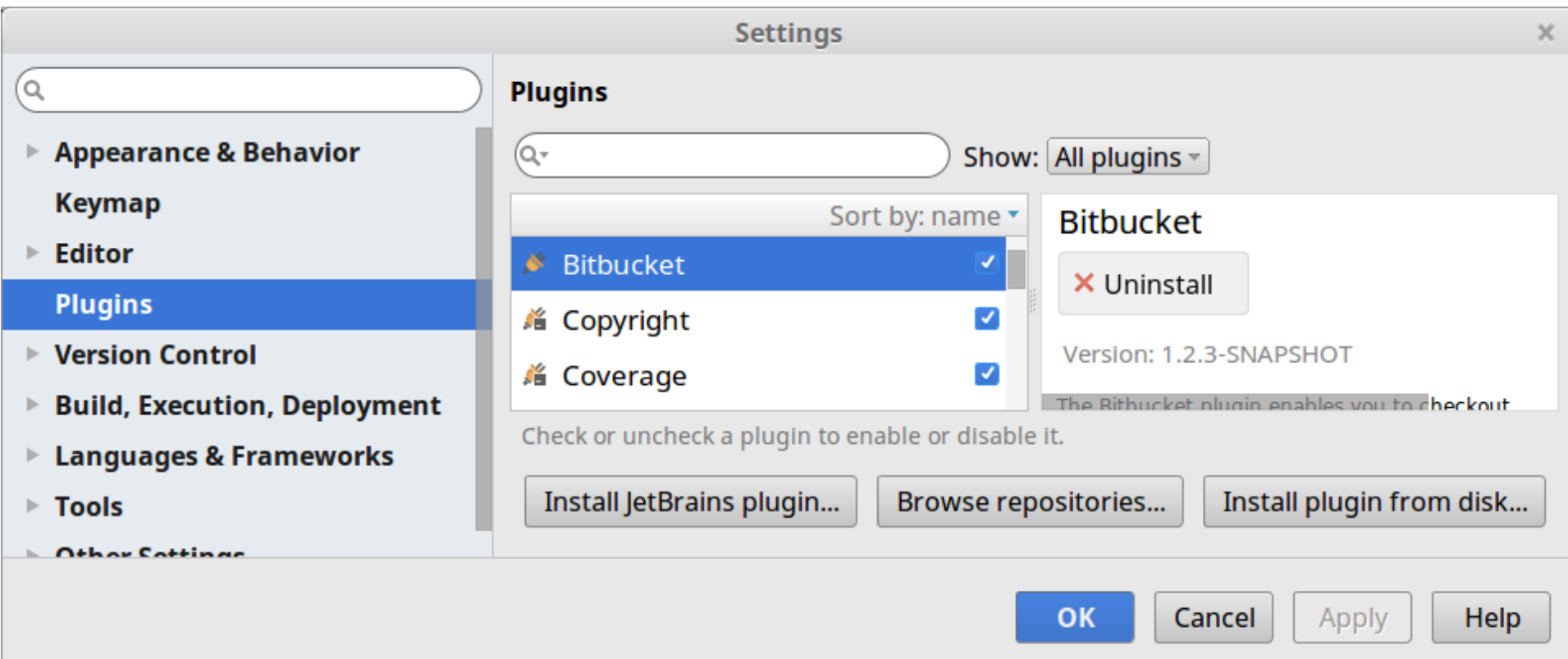
All plans include:

- | | |
|---|--|
| <input checked="" type="checkbox"/> Unlimited private repos | <input checked="" type="checkbox"/> Dedicated support |
| <input checked="" type="checkbox"/> Code reviews | <input checked="" type="checkbox"/> Branch level permissions |
| <input checked="" type="checkbox"/> JIRA integration | <input checked="" type="checkbox"/> REST API |

Plugin para Android Studio: [Bitbucket connector](#)

Sistema de control de versiones: Git

Settings / Plugins



Sistema de control de versiones: Git

Terminal en Android Studio:

```
Terminal
+ paco@Latitude ~/AndroidStudioProjects/Primera $ git add .
+ paco@Latitude ~/AndroidStudioProjects/Primera $ git commit -m 'Inicial'
X [master (root-commit) 13766e6] Inicial
  36 files changed, 637 insertions(+)
  create mode 100644 .gitignore
  create mode 100644 .idea/.name
  create mode 100644 .idea/compiler.xml
  create mode 100644 .idea/copyright/profiles_settings.xml
  create mode 100644 .idea/encodings.xml
  create mode 100644 .idea/gradle.xml
  create mode 100644 .idea/misc.xml
```

TODO 6: Android Monitor Terminal 9: Version Control

git add .

git commit -m 'Cambios'

git remote add origin https://paco_portada@bitbucket.org/paco_portada/primera-butterknife.git

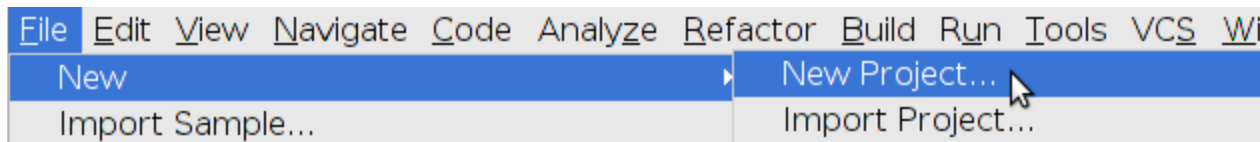
git push -u origin --all

Integrate BitBucket Plug-in Android Studio

Estructura de un proyecto Android

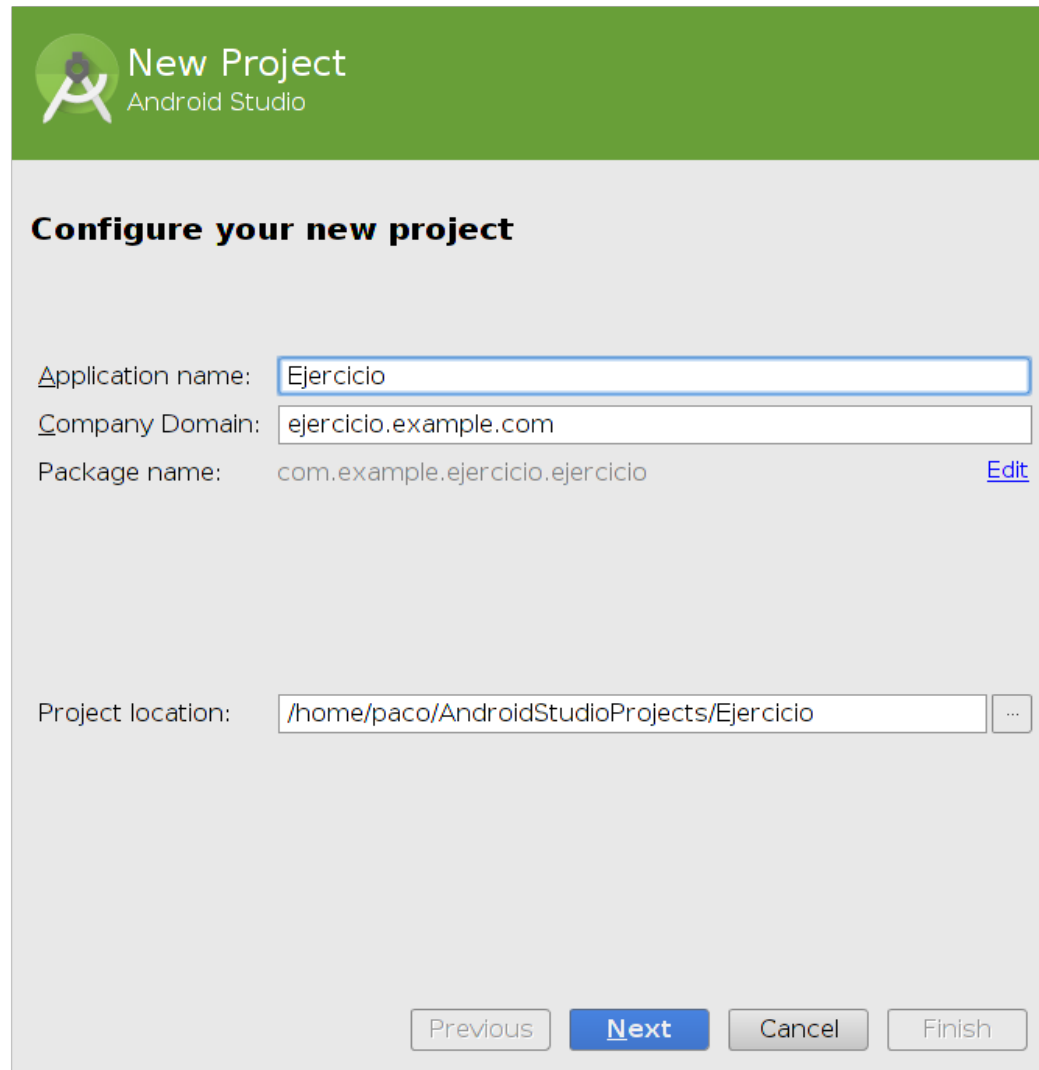
Para empezar a comprender cómo se construye una aplicación Android vamos a crear un nuevo proyecto Android y echaremos un vistazo a la estructura general del proyecto creado por defecto.


Para crear un nuevo proyecto abriremos Android Studio e iremos al menú
File / New / Project.



De esta forma iniciaremos el asistente de creación del proyecto, que nos guiará por las distintas opciones de creación y configuración de un nuevo proyecto.

Estructura de un proyecto Android




 **New Project**
Android Studio

Configure your new project

Application name:

Company Domain:

Package name: [Edit](#)

Project location: 

Estructura de un proyecto Android

Target Android Devices

Select the form factors your app will run on

Different platforms may require separate SDKs

☒ Phone and Tablet

Minimum SDK: API 16: Android 4.1 (Jelly Bean)

☐ Wear

Minimum SDK

☐ TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

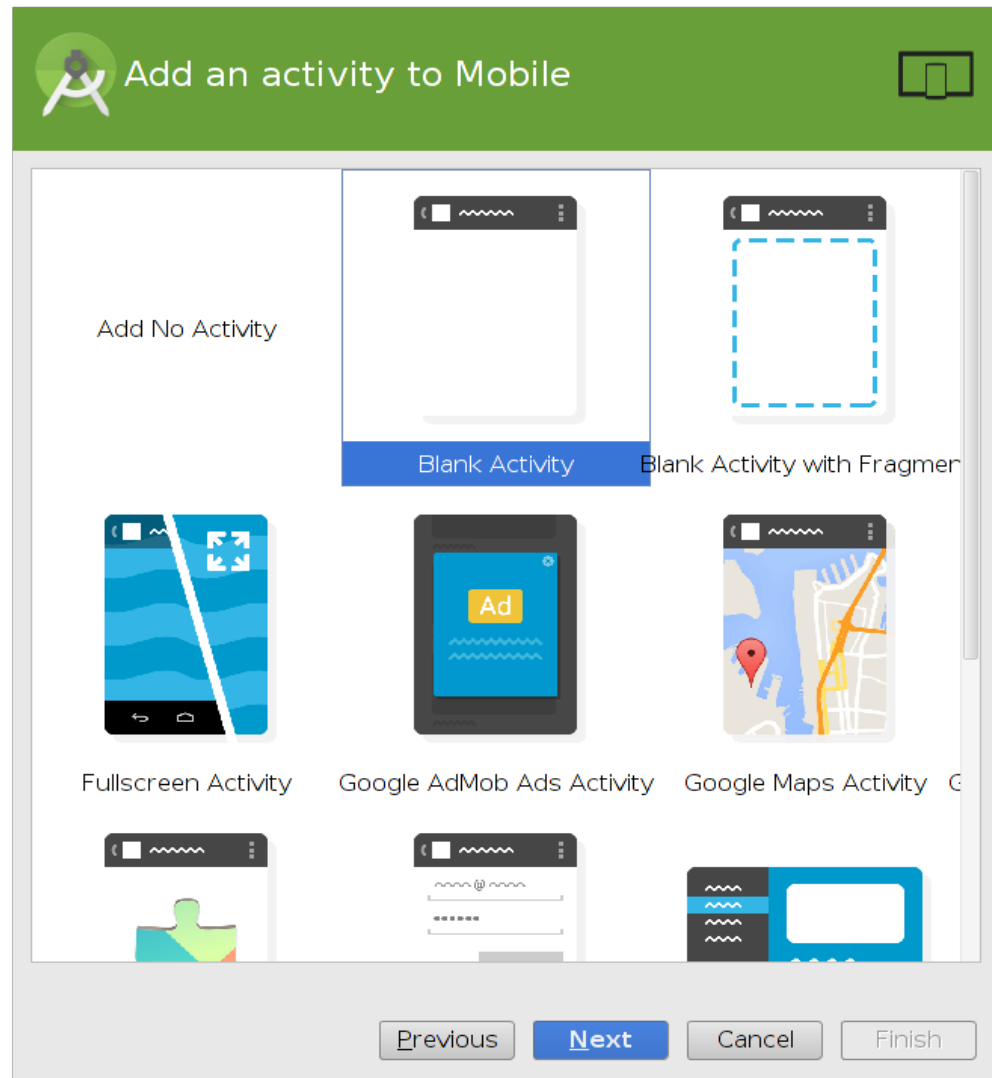
☐ Android Auto

☐ Glass (Not Installed) [Download](#)



Minimum SDK

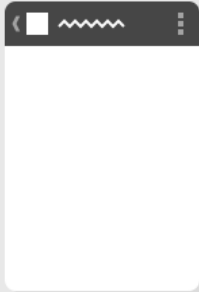
Previous Next Cancel Finish

Estructura de un proyecto Android



Estructura de un proyecto Android

 Customize the Activity 



Blank Activity

Creates a new blank activity with an action bar.

Activity Name:

Layout Name:

Title:

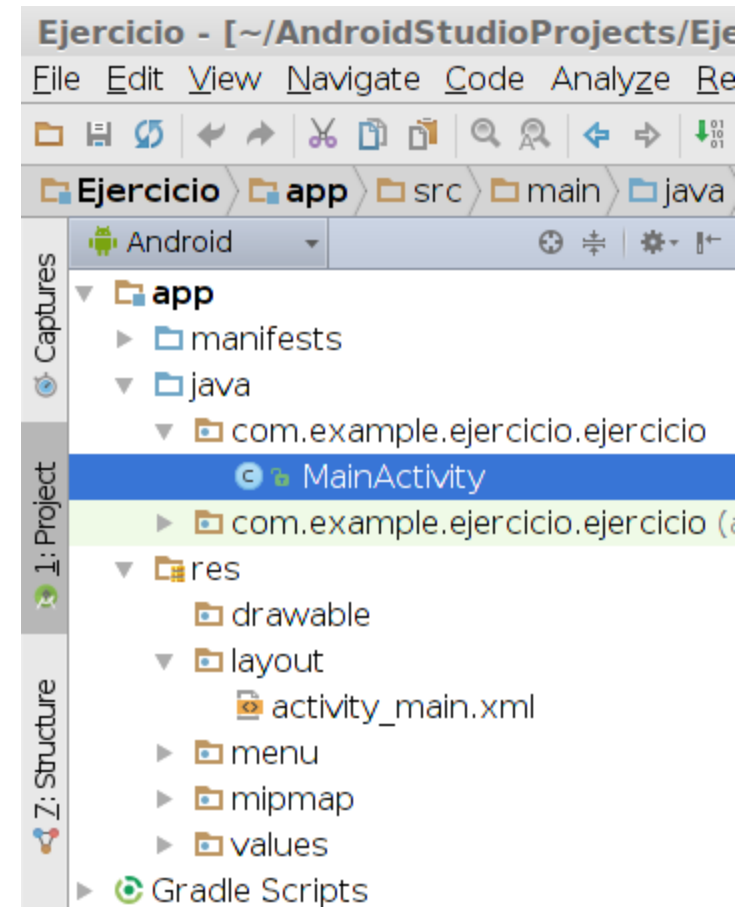
Menu Resource Name:

The name of the activity. For launcher activities, the application title.

Estructura de un proyecto Android

Una vez configurado todo, pulsamos el botón Finish y se creará toda la estructura del proyecto y los elementos indispensables que debe contener.

En la imagen vemos los elementos creados inicialmente para un nuevo proyecto Android:



Estructura de un proyecto Android

Los elementos principales de esta estructura son:

Carpeta /java

Esta carpeta contendrá todo el código fuente de la aplicación, código de la interfaz gráfica, clases auxiliares, etc.

Inicialmente, se creará por nosotros el código básico de la pantalla (Activity) principal de la aplicación, que recordemos que en nuestro caso era MainActivity, y siempre bajo la estructura del paquete java definido.

Carpeta /res

Contiene todos los ficheros de recursos necesarios para el proyecto: imágenes, vídeos, etc.

Se puede consultar la lista completa en la [documentación oficial del Android](#)

Fichero AndroidManifest.xml

Contiene la definición en XML de los aspectos principales de la aplicación, como por ejemplo su identificación (nombre, versión, icono, ...), sus componentes (pantallas, mensajes, ...), las librerías auxiliares utilizadas o los permisos necesarios para su ejecución.

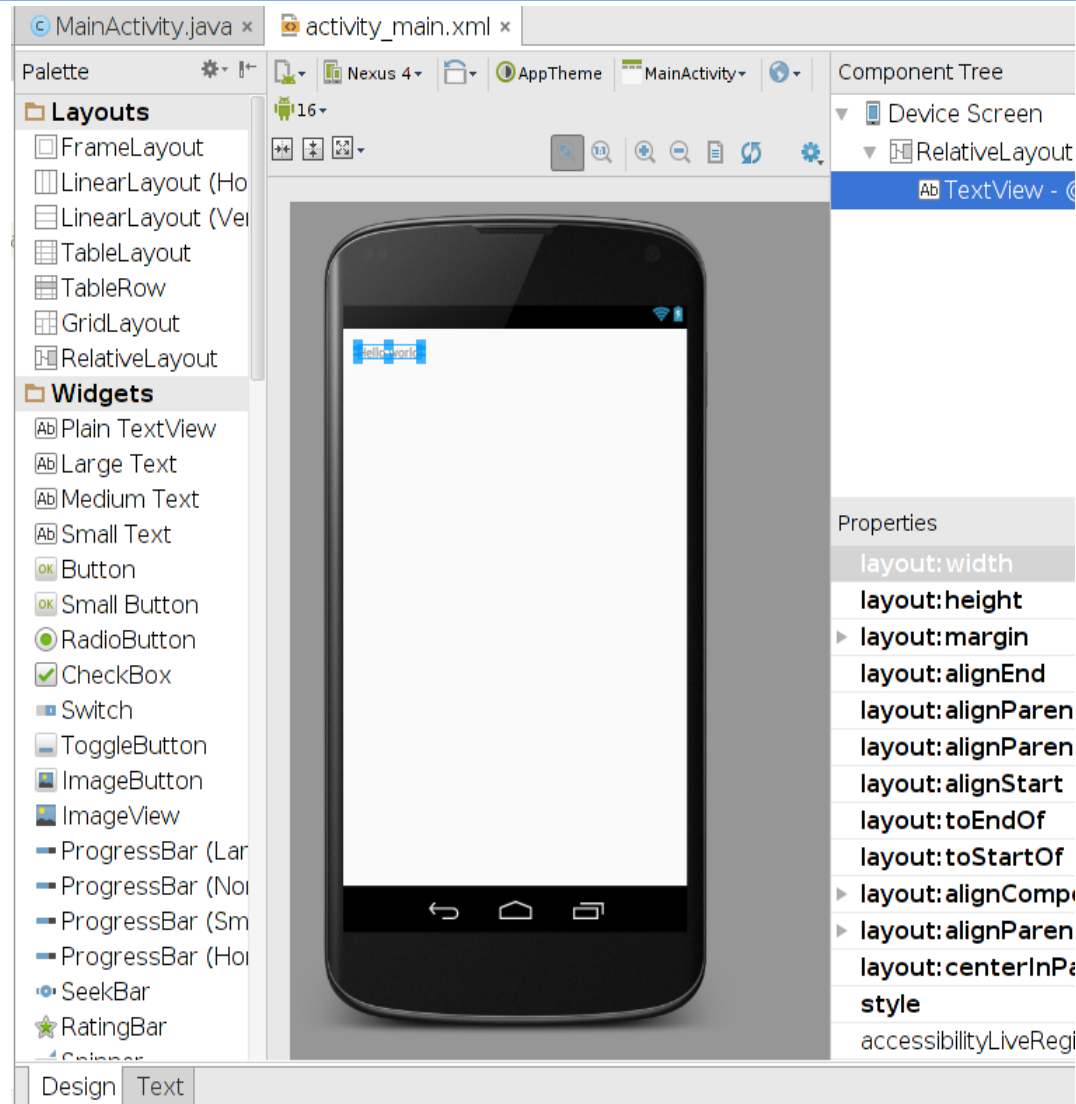
Estructura de un proyecto Android

Entre los recursos creados por defecto, cabe destacar el layout "**activity_main.xml**", que contiene la definición de la interfaz gráfica de la pantalla principal de la aplicación.

Si hacemos doble clic sobre el fichero, Android Studio Eclipse nos mostrará su editor gráfico (tipo arrastrar y soltar) con los elementos de la interfaz gráfica, que contiene tan sólo una etiqueta de texto con el mensaje "Hello World!".

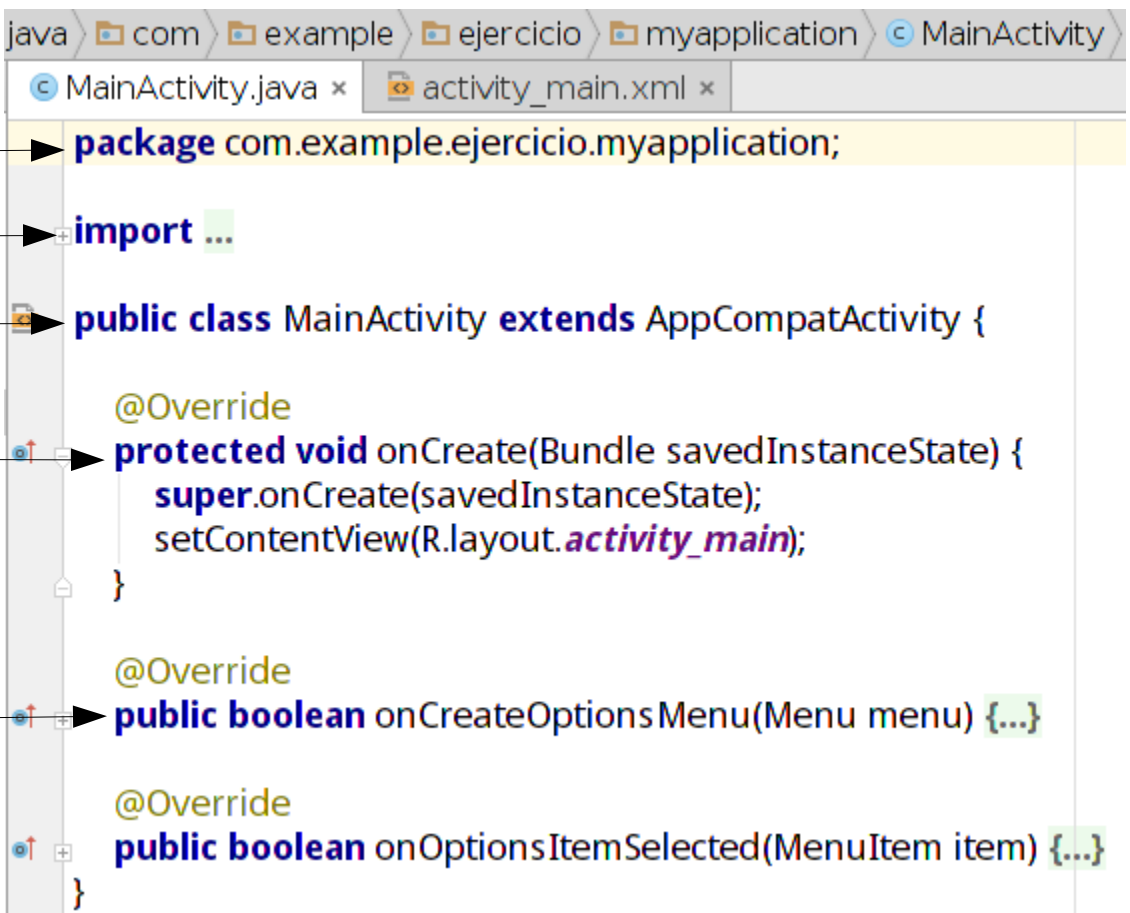
En lugar de usar el editor gráfico se puede modificar directamente el fichero XML asociado (al que se puede acceder pulsando sobre la pestaña inferior derecha, en `activity_main.xml`).

Estructura de un proyecto Android



Estructura de un proyecto Android

Y éste es el código de nuestra primera aplicación:



The screenshot shows the MainActivity.java file in an Android Studio IDE. The package path in the breadcrumb is java > com > example > ejercicio > myapplication > MainActivity. The code is as follows:

```
package com.example.ejercicio.myapplication;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {...}

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {...}
}
```

Annotations on the left side of the code:

- Nombre del paquete —→ package com.example.ejercicio.myapplication;
- Paquetes que se importan —→ import ...
- Clase principal basada en Activity o AppCompatActivity —→ public class MainActivity extends AppCompatActivity {
- Método onCreate —→ @Override protected void onCreate(Bundle savedInstanceState) {
- Menú del ActionBar o Toolbar —→ @Override public boolean onCreateOptionsMenu(Menu menu) {...}

Componentes de una aplicación Android



Vamos a ver los distintos tipos de componentes de software con los que podremos construir una aplicación Android.

En Java o .NET estamos acostumbrados a manejar conceptos como ventana, control, eventos o servicios como los elementos básicos en la construcción de una aplicación.

En Android vamos a disponer de esos mismos elementos básicos aunque con un pequeño cambio en la terminología y el enfoque.

Repasemos los componentes principales que pueden formar parte de una aplicación Android

Componentes de una aplicación Android

Activity

Las actividades (activities) representan el componente principal de la interfaz gráfica de una aplicación Android.

Se puede pensar en una actividad como el elemento análogo a una ventana o pantalla en cualquier otro lenguaje visual.

Más información en developer.android.com

Creación de una Actividad:

Para crear una actividad, se debe crear una subclase de la clase **Activity**(o una subclase existente).

En tu subclase debes implementar los métodos a los que llama el sistema cuando la actividad realiza transiciones entre los diferentes estados de su ciclo de vida, tales como cuando la actividad es creada, parada, reiniciada o destruida.

El método más importante es **onCreate()**.

Componentes de una aplicación Android

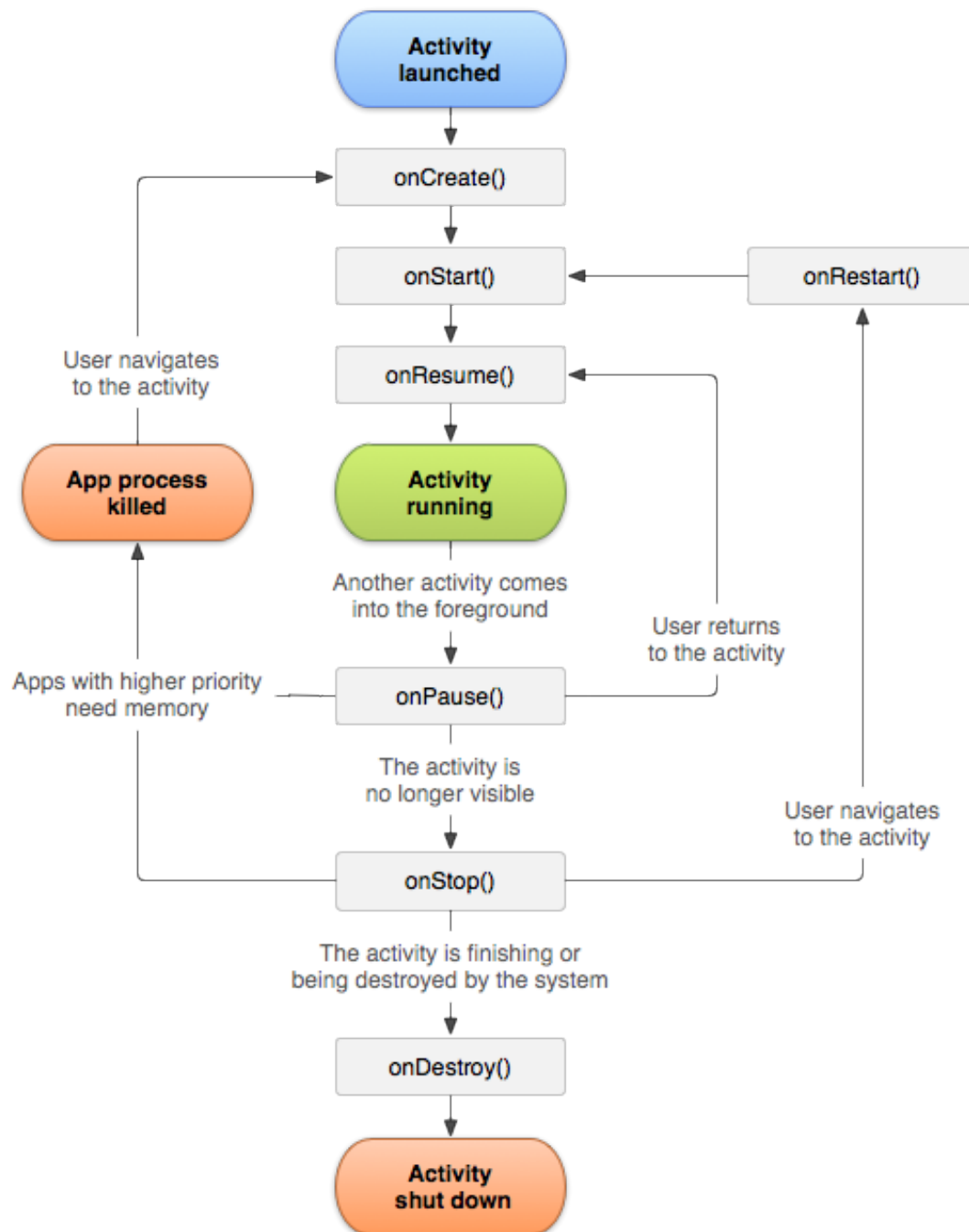
onCreate()

Se debe implementar este método. El sistema lo llama cuando se crea la actividad.

En tu implementación debes inicializar los componentes principales de tu actividad. Lo más importante es llamar al método **setContentView()** para definir el layout para la interfaz de usuario de la actividad.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    ...
}
```



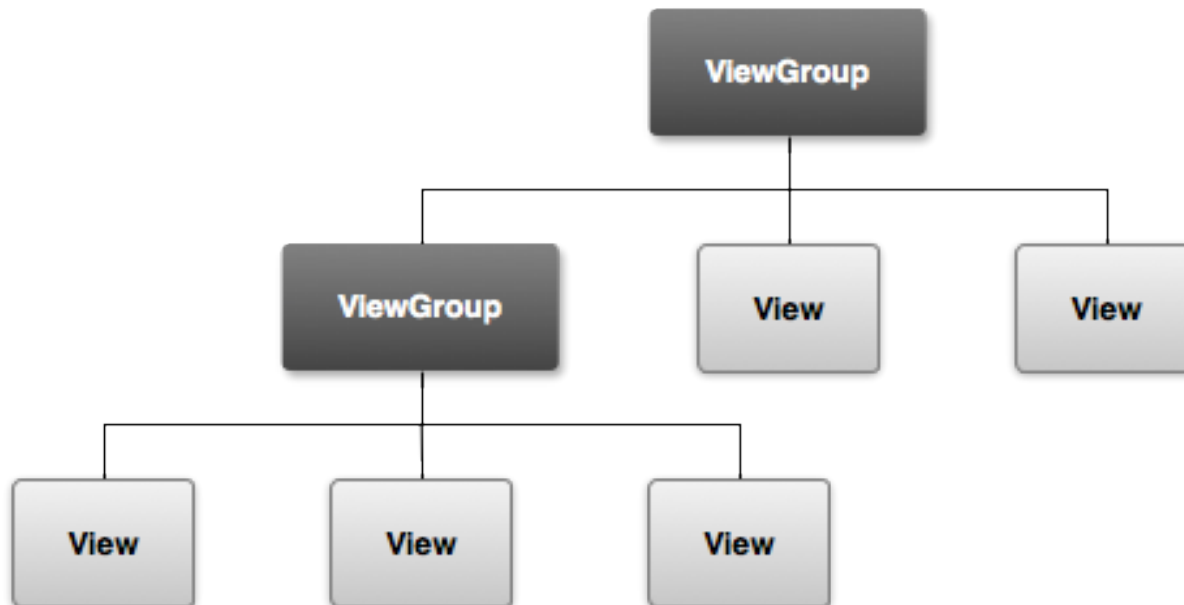
Componentes de una aplicación Android

View

Las vistas (view) son los componentes básicos con los que se construye la interfaz gráfica de la aplicación, análogo por ejemplo a los controles de Java o .NET.

De inicio, Android pone a nuestra disposición una gran cantidad de controles básicos, como cuadros de texto, botones, listas desplegables o imágenes, aunque también existe la posibilidad de extender la funcionalidad de estos controles básicos o crear nuestros propios controles personalizados.

Más información en [Building a Simple User Interface](#)



Componentes de una aplicación Android

Intent

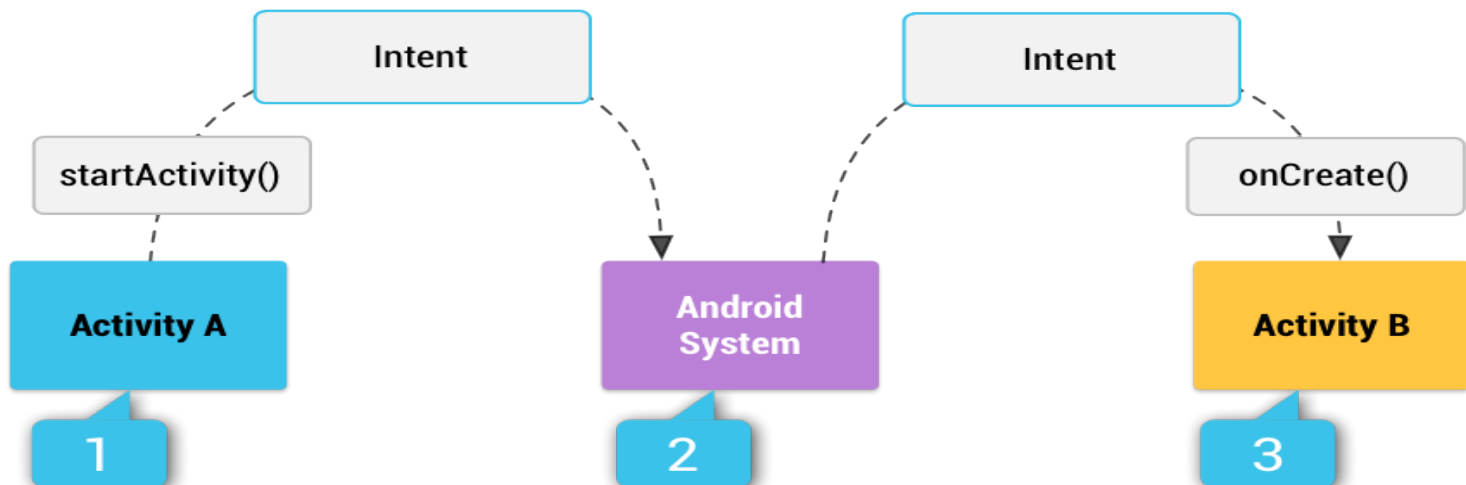
Un intent es el elemento básico de comunicación entre los distintos componentes Android. Se pueden entender como los mensajes o peticiones que son enviados entre los distintos componentes de una aplicación o entre distintas aplicaciones.

Mediante un intent se puede mostrar una actividad desde cualquier otra, iniciar un servicio, enviar un mensaje broadcast, iniciar otra aplicación, etc.

Hay 2 tipos de intents:

- Implícitos
- Explícitos

Más información en [Intents and Intent Filters](#)



Componentes de una aplicación Android

Service

Los servicios son componentes sin interfaz gráfica que se ejecutan en segundo plano. En concepto, son similares a los servicios presentes en cualquier otro sistema operativo.

Los servicios pueden realizar cualquier tipo de acciones, por ejemplo actualizar datos, lanzar notificaciones, o incluso mostrar elementos visuales (p.ej. actividades) si se necesita en algún momento la interacción con el usuario.

Content Provider

Un content provider es el mecanismo que se ha definido en Android para compartir datos entre aplicaciones.

Mediante estos componentes es posible compartir determinados datos de nuestra aplicación sin mostrar detalles sobre su almacenamiento interno, su estructura, o su implementación.

De la misma forma, nuestra aplicación podrá acceder a los datos de otra a través de los content provider que se hayan definido.

Componentes de una aplicación Android

Broadcast Receiver

Un broadcast receiver es un componente destinado a detectar y reaccionar ante determinados mensajes o eventos globales generados por el sistema (por ejemplo: "Batería baja", "SMS recibido", "Tarjeta SD insertada", ...) o por otras aplicaciones (cualquier aplicación puede generar mensajes broadcast, es decir, no dirigidos a una aplicación concreta sino a cualquiera que quiera escucharlo).

Widget

Los widgets son elementos visuales, normalmente interactivos, que pueden mostrarse en la pantalla principal (home screen) del dispositivo Android y recibir actualizaciones periódicas.

Permiten mostrar información de la aplicación al usuario directamente sobre la pantalla principal.

Java para desarrollo Android

Es conveniente dominar Java para programar en Android.

En este artículo se repasan los conceptos básicos del lenguaje Java con el fin de recordar y fortalecer los conocimientos necesarios para desarrollo Android:

Introducción A Java Para Desarrollo Android



Java para desarrollo Android

Libro para aprender Java con ejercicios:



[Aprende Java con ejercicios en GitHub](#)

Ejemplos

- **Mi primera aplicación**
- **Divisas**
- **2 actividades**
- **Contador de cafés**



Mi primera aplicación

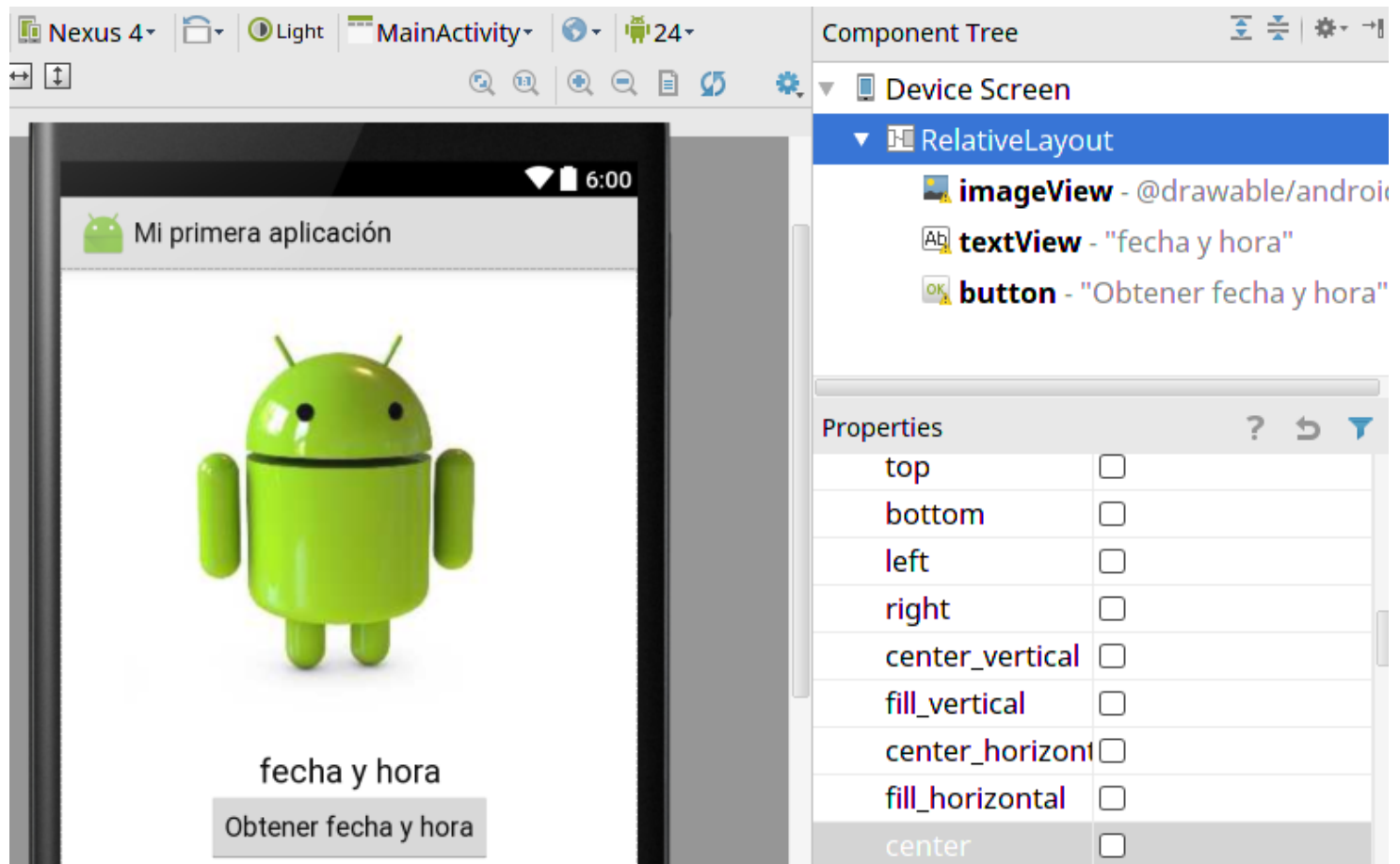
Crear una aplicación que muestre la fecha y hora actual cada vez que se pulse un botón



Mi primera aplicación

Se creará una nueva aplicación.

Se añadirán al layout una imagen (en la carpeta drawable), una etiqueta y un botón



Mi primera aplicación

El fichero xml del layout será similar a este:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context="com.example.paco.ejercicio1.MainActivity">

    <ImageView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/imageView"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:src="@drawable/android" />
```

Mi primera aplicación

```
<TextView
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:text="fecha y hora"
    android:id="@+id/textView"
    android:layout_marginTop="27dp"
    android:layout_below="@+id/imageView"
    android:layout_alignParentStart="true"
    android:gravity="center" />

<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Obtener fecha y hora"
    android:id="@+id/button"
    android:layout_below="@+id/textView"
    android:layout_centerHorizontal="true" />
</RelativeLayout>
```

Mi primera aplicación

fichero strings.xml:

```
<resources>
  <string name="app_name">Mi primera aplicación</string>
  <string name="obtener_fecha_y_hora">Obtener fecha y hora</string>
  <string name="fecha_y_hora">fecha y hora</string>
</resources>
```

fichero styles.xml:

```
<resources>

  <!-- Base application theme. -->
  <style name="AppTheme" parent="Theme.AppCompat.Light.DarkActionBar">
    <!-- Customize your theme here. -->
    <item name="colorPrimary">@color/colorPrimary</item>
    <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
    <item name="colorAccent">@color/colorAccent</item>
  </style>

</resources>
```


Mi primera aplicación

En el código se usará la interfaz `OnClickListener`, se modificará el método `onCreate` y se añadirán los métodos `onClick(View v)` y `actualizar`:

```
public class MainActivity extends AppCompatActivity implements OnClickListener {
    //Definir objetos

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //Hacer referencia a los elementos de la vista
    }

    public void onClick(View v){
        //Tratar el evento click del botón
    }

    private void actualizar(){
        //Actualizar la fecha y hora
    }
}
```

Mi primera aplicación

```
public class MainActivity extends AppCompatActivity implements OnClickListener {
    Button boton;
    TextView texto;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        boton = (Button) findViewById(R.id.button);
        texto = (TextView) findViewById(R.id.textView);
        boton.setOnClickListener(this);
        actualizar();
    }

    public void onClick(View v){
        if (v == boton)
            actualizar();
    }

    private void actualizar(){
        texto.setText(new Date().toString());
    }
}
```

Mi primera aplicación

fichero build.gradle (app):

```
apply plugin: 'com.android.application'
```

```
android {  
    compileSdkVersion 24  
    buildToolsVersion "24.0.1"
```

```
    defaultConfig {  
        applicationId "com.example.ejercicio1"  
        minSdkVersion 22  
        targetSdkVersion 24  
        versionCode 1  
        versionName "1.0"
```

```
    }  
    buildTypes {  
        release {  
            minifyEnabled false  
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'  
        }  
    }  
}
```

```
dependencies {  
    compile fileTree(dir: 'libs', include: ['*.jar'])  
    testCompile 'junit:junit:4.12'  
    compile 'com.android.support:appcompat-v7:24.1.1'  
}
```

Mi primera aplicación

Button

Represents a push-button widget. Push-buttons can be pressed, or clicked, by the user to perform an action.

A typical use of a push-button in an activity would be the following:

```
public class MyActivity extends Activity {
    protected void onCreate(Bundle icle) {
        super.onCreate(icle);

        setContentView(R.layout.content_layout_id);

        final Button button = (Button) findViewById(R.id.button_id);
        button.setOnClickListener(new View.OnClickListener() {
            public void onClick(View v) {
                // Perform action on click
            }
        });
    }
}
```

However, instead of applying an [OnClickListener](#) to the button in your activity, you can assign a method to your button in the XML layout, using the [android:onClick](#) attribute. For example:

```
<Button
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:text="@string/self_destruct"
    android:onClick="selfDestruct" />
```

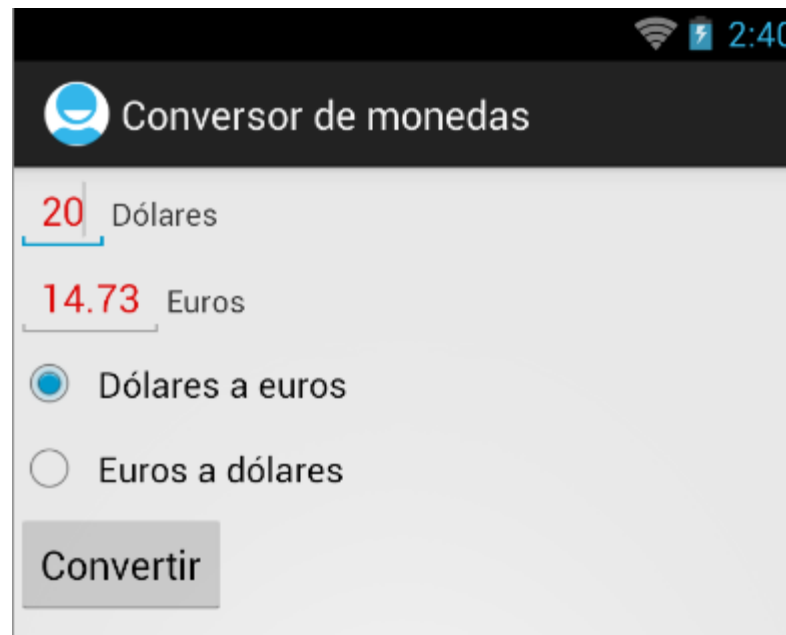
Mi primera aplicación

Subir el proyecto a un repositorio en Bitbucket.org



Divisas

Crear una aplicación que permita convertir dólares a euros y viceversa



Convertir texto a número

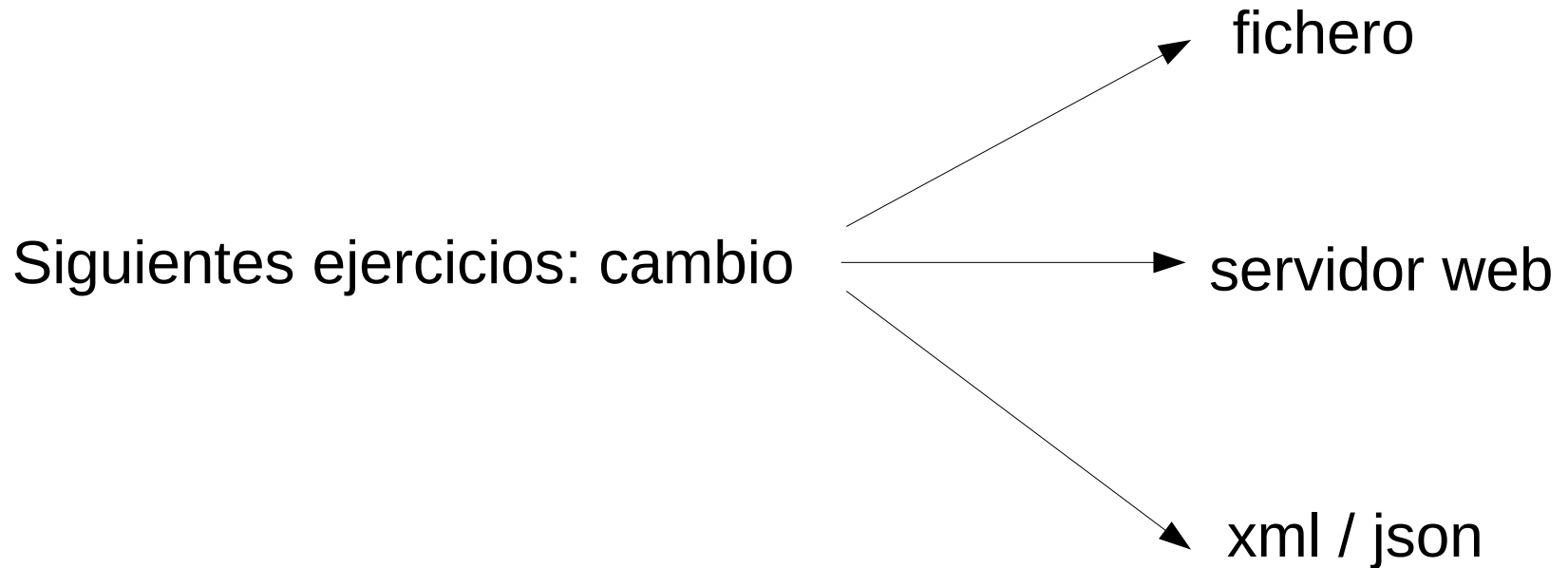
Pasar una cadena de texto a número y viceversa:

```
public String convertirADolares(String cantidad) {  
    double valor = Double.parseDouble(cantidad) / cambio;  
    return Double.toString(valor);  
    //return String.valueOf(valor);  
}
```

Poner un formato con dos decimales:

```
public String convertirAEuros(String cantidad) {  
    double valor = Double.parseDouble(cantidad) * cambio;  
    return String.format("%.2f", valor);  
}
```

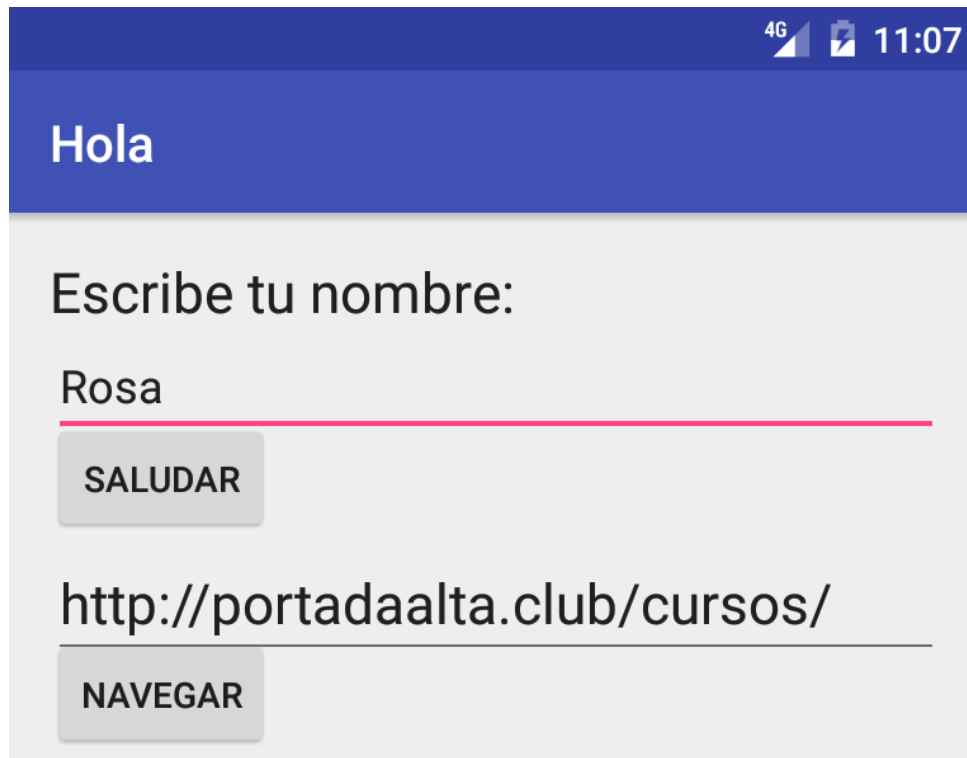
Convertir texto a número



2 actividades

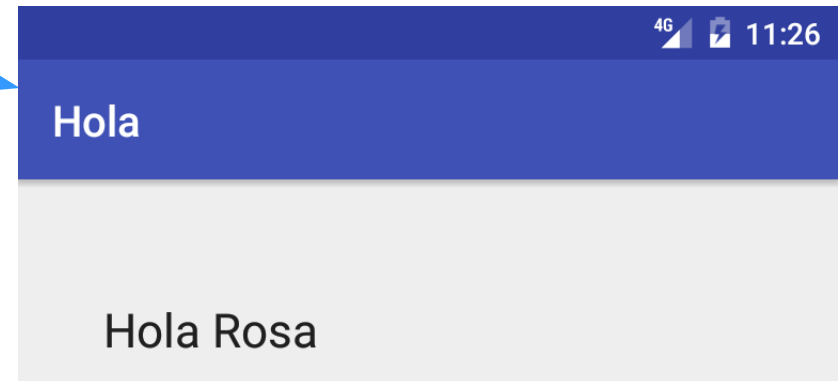
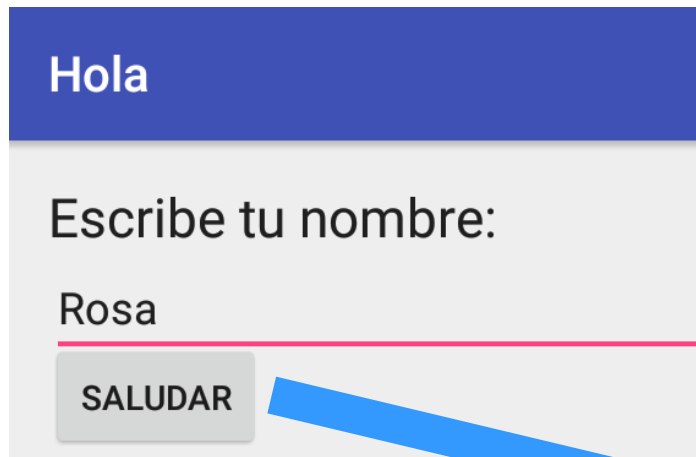
Crear una aplicación que pida un nombre y muestre en una segunda ventana un saludo con ese nombre.

También se podrá introducir una URL para navegar.



The screenshot shows a mobile application interface. At the top, there is a dark blue header bar with the text "Hola" in white. Below the header, the background is light gray. The first section contains the text "Escribe tu nombre:" followed by a text input field containing the name "Rosa". Below the input field is a gray button with the text "SALUDAR". The second section contains a text input field with the URL "http://portadaalta.club/cursos/". Below this input field is a gray button with the text "NAVEGAR". The top status bar of the phone is visible, showing "4G", signal strength, battery level, and the time "11:07".

2 actividades

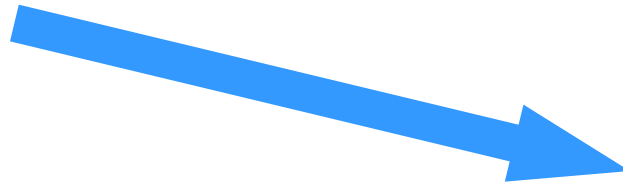


Más información sobre Intents

Intent explícito

Primera actividad

```
public static final String DATO = "nombre";  
...  
i = new Intent (this, Segunda.class);  
i.putExtra(DATO, texto.getText().toString());  
startActivity(i);
```



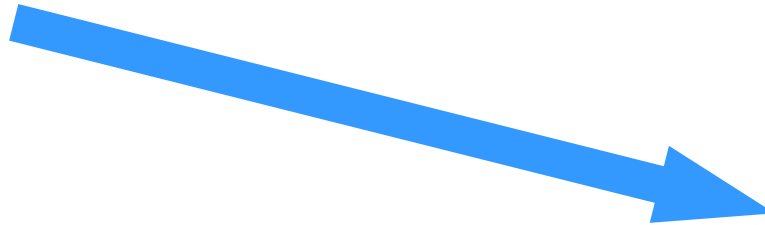
Segunda actividad

```
TextView saludo;  
public static final String DATO = "nombre";  
...  
saludo = (TextView) findViewById(R.id.textView);  
Intent i = this getIntent();  
saludo.setText("Hola " + i.getStringExtra(DATO));
```

Intent implícito

Actividad

```
public void openWebPage(String url) {  
    Uri webpage = Uri.parse(url);  
    Intent intent = new Intent(Intent.ACTION_VIEW, webpage);  
    if (intent.resolveActivity(getPackageManager()) != null) {  
        startActivity(intent);  
    }  
}
```



Navegador

Más información sobre Common Intents

Contador de cafés

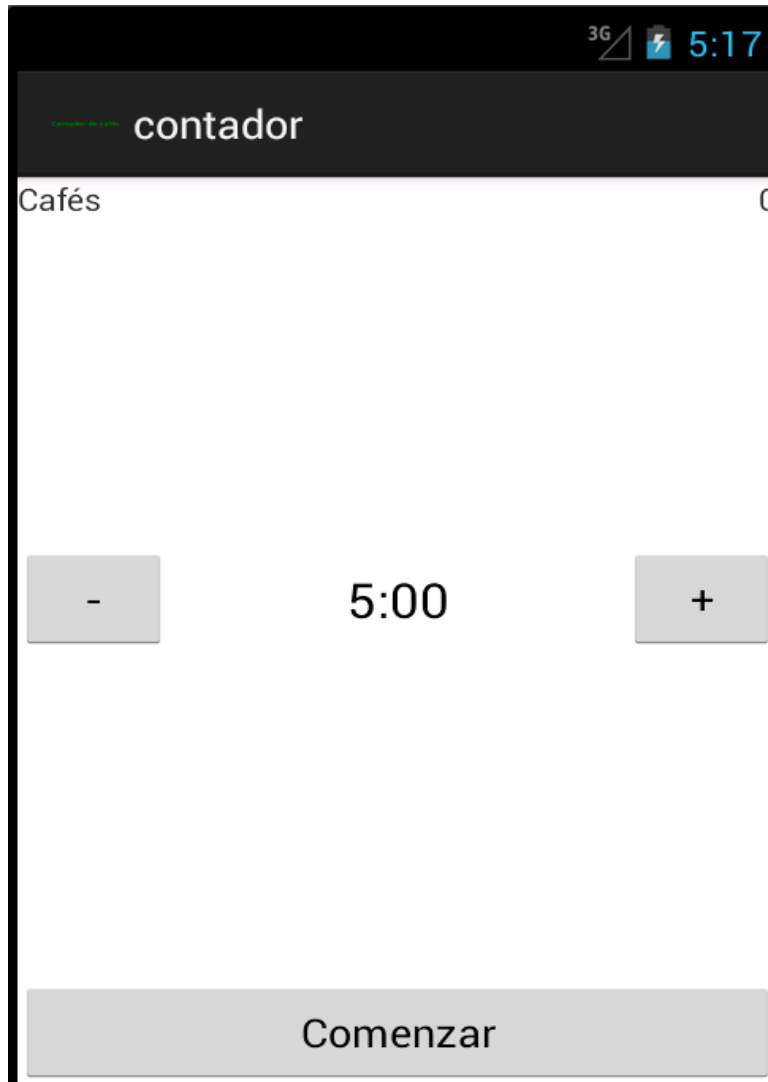
Crear una aplicación que muestre un temporizador para controlar la duración de la pausa para tomar un café.

Habrà un botón para iniciar el temporizador, se podrá configurar la duración de la pausa (en minutos) y se avisará al terminar el tiempo.

Además, contará los cafés tomados diariamente: se incrementará el contador al terminar cada pausa.



Contador de cafés



Clase CountdownTimer

Lanza un contador descendente hasta un tiempo (en milisegundos) en el futuro, con notificaciones regulares cada cierto intervalo de tiempo (en milisegundos)

Ejemplo de uso:

```
public class MyCountDownTimer extends CountDownTimer {  
  
    public MyCountDownTimer(long startTime, long interval) {  
        super(startTime, interval);  
    }  
    @Override  
    public void onTick(long millisUntilFinished) {  
        texto.setText(minutos + ":" + segundos);  
        . . .  
    }  
    @Override  
    public void onFinish() {  
        texto.setText("Pausa terminada!!");  
        . . .  
    }  
}
```

```
//contadorTiempo indica los minutos
```

```
miContador = new MyCountDownTimer(contadorTiempo * 60 * 1000, 1000);  
miContador.start();
```

Fin

¿Dudas?

¿Sugerencias?



paco@portadaalta.es