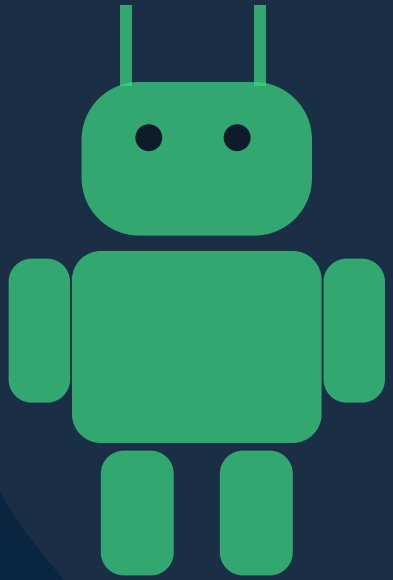
TEMA 2

Introducción a Android

Programación de Dispositivos Móviles

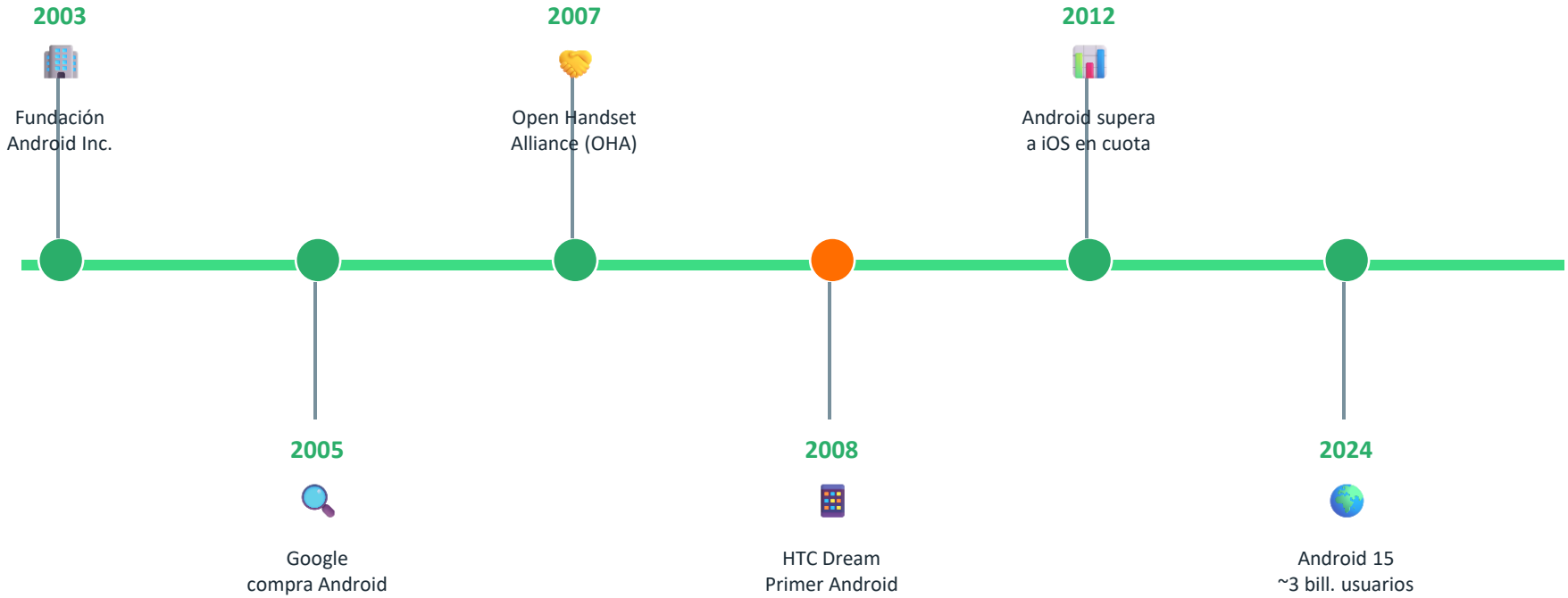


Historia y Arquitectura | Entorno de Desarrollo | Fundamentos de Apps | Interfaz de Usuario | Recursos

CONTENIDOS

Tema 2

Historia y Orígenes de Android



🤝 Open Handset Alliance: Google · HTC · Samsung · LG · Motorola · Sony · Qualcomm · Intel — Filosofía de código abierto y gratuito

Arquitectura de Android: Capas del Sistema

APP **APLICACIONES**

Apps del sistema y de terceros (Java/Kotlin)

API **FRAMEWORK DE APLICACIONES**

Activity Manager · Window Manager · Content Providers · Notifications

LIB **LIBRERÍAS NATIVAS + ANDROID RUNTIME (ART)**

SQLite · WebKit · OpenGL ES · Compilación AOT/JIT

HAL **CAPA DE ABSTRACCIÓN DE HARDWARE (HAL)**

Drivers y abstracción de hardware — interfaz estandarizada

KRN **NÚCLEO LINUX (Linux Kernel)**

Gestión de memoria · Procesos · Seguridad · Energía

Versiones de Android: De Cupcake a Android 15

Android 2.3
Gingerbread

2010

NFC, mejoras UI

Android 4.0
Ice Cream Sandwich

2011

Face Unlock, unificó phone/tablet

Android 5.0
Lollipop

2014

Material Design, ART, 64-bit

Android 6.0
Marshmallow

2015

Permisos en tiempo de ejecución

Android 8.0
Oreo

2017

Picture-in-picture, Project Treble

Android 10
Android Q

2019

5G nativo, modo oscuro oficial

Android 12
Material You

2021

Personalización dinámica, privacidad

Android 15
Android 15

2024

IA integrada, audio espacial

Distribución 2024: Android 14 ~30% · Android 13 ~25% · Android 12 ~18% · Android 11 o inferior ~27%

El Ecosistema Android

Google Play Store

- +3 millones de apps
- Google Play Protect
- Distribución global
- Políticas estrictas de publicación

Google Mobile Services (GMS)

- Maps, Gmail, YouTube, Drive
- Google Play Services (APIs)
- Requiere licencia fabricantes
- Servicios propietarios de Google

AOSP (Android Open Source)

- Código fuente completamente abierto
- Base para ROMs personalizadas
- Amazon Fire OS
- Huawei HMS (sin GMS)

Android Enterprise

- Gestión dispositivos corporativos
- MDM (Mobile Device Management)
- Perfiles de trabajo separados
- Seguridad empresarial

Android Studio: El IDE Oficial

Requisitos del Sistema

SO	Windows 8+ · macOS 10.14+ · Linux 64-bit
RAM	4 GB mínimo → 16 GB recomendado
Disco	2 GB mínimo → 8 GB+ (SSD recomendado)
Pantalla	1280×800 mínimo → 1920×1080+

Pasos de instalación:

1. Descargar desde developer.android.com/studio
2. Ejecutar instalador (.exe / .dmg / studio.sh)
3. Seleccionar tipo Standard y tema
4. Descargar Android SDK y componentes

Estructura del Proyecto

```
MiApp/
├── app/
│   ├── src/main/
│   │   ├── java/.../
│   │   │   └── MainActivity.kt
│   │   ├── res/
│   │   │   ├── layout/
│   │   │   │   └── activity_main.xml
│   │   │   ├── values/
│   │   │   │   ├── strings.xml
│   │   │   │   ├── colors.xml
│   │   │   │   └── themes.xml
│   │   │   └── drawable/
│   │   └── AndroidManifest.xml
│   └── build.gradle.kts
└── settings.gradle.kts
```

Emuladores, Dispositivos Reales y ADB

AVD — Android Virtual Device

Ventajas:

- Testing sin dispositivo físico
- Múltiples versiones simultáneas
- GPS, sensores y batería simulados
- Capturas y grabaciones fáciles

Limitaciones:

- Rendimiento inferior al real
- Consume recursos del PC
- No soporta todas las funciones

Activar depuración USB:
Ajustes → Acerca del teléfono → Tocar 'Número de compilación' 7 veces → Opciones de desarrollador

Comandos ADB Esenciales

```
# Gestión de dispositivos
adb devices
adb connect IP:PUERTO

# Apps
adb install app.apk
adb uninstall com.paquete.app

# Depuración
adb logcat
adb shell

# Archivos
adb pull /ruta local/
adb push local /ruta/

# WiFi (Android 11+)
adb pair IP:PUERTO
adb reboot
```

Los 4 Componentes Principales de Android

1 Activity

Representa una pantalla con interfaz de usuario

- Punto de entrada de interacción
- Toda app tiene al menos 1 (MainActivity)
- Gestiona interfaz visible al usuario

2 Service

Operación en segundo plano sin interfaz visible

- Started Service (independiente)
- Bound Service (vinculado a Activity)
- Ej: música, descargas, sync

3 Broadcast Receiver

Escucha eventos del sistema o de otras apps

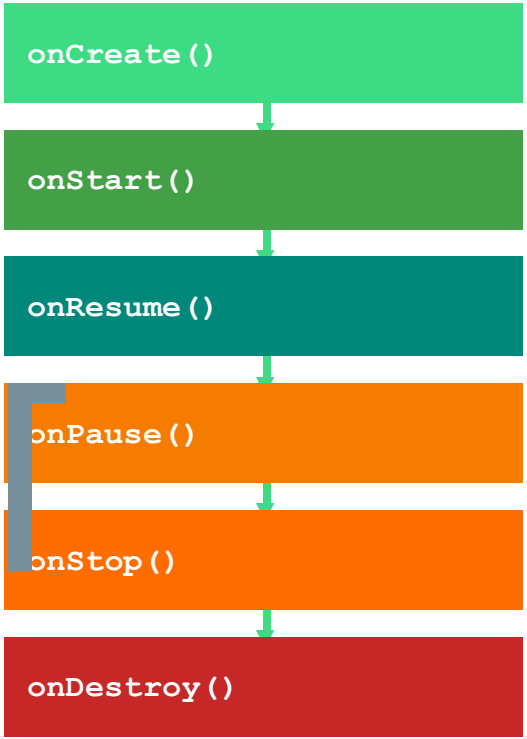
- Responde a mensajes del sistema
- Ej: batería baja, cambio de red
- Registro en Manifest o dinámico

4 Content Provider

Gestiona y comparte datos entre aplicaciones

- Abstracción sobre fuentes de datos
- Ej: contactos, galería, archivos
- API estándar: query/insert/update/delete

Ciclo de Vida de un Activity



Inicialización, setContentView

Activity visible al usuario

Activity interactivo (foreground)

Pierde foco, parcialmente visible

Activity no visible

Limpieza final de recursos

Implementación completa

```
class MainActivity :
    AppCompatActivity() {

  override fun onCreate(s: Bundle?) {
    super.onCreate(s)
    setContentView(R.layout.main)
    Log.d("LC", "onCreate")
  }
  override fun onStart() {
    super.onStart()
    Log.d("LC", "onStart")
  }
  override fun onResume() {
    super.onResume()
    Log.d("LC", "onResume")
  }
  override fun onPause() {
    super.onPause()
    Log.d("LC", "onPause")
  }
  override fun onDestroy() {
    super.onDestroy()
    Log.d("LC", "onDestroy")
```

Intents: Comunicación entre Componentes

🎯 Intent EXPLÍCITO

Especifica el componente destino exacto. Se usa dentro de la misma app.

```
// Navegar a otra Activity
val intent = Intent(
  this,
  SecondActivity::class.java
)
// Pasar datos
intent.putExtra("KEY", "Valor")
startActivity(intent)

// Recibir en SecondActivity
val valor = intent
  .getStringExtra("KEY")
```

🌐 Intent IMPLÍCITO

Declara una acción. El sistema elige la app adecuada.

```
// Abrir URL en navegador
val i = Intent(Intent.ACTION_VIEW)
i.data = Uri.parse("https://...")
startActivity(i)

// Llamar por teléfono
val i = Intent(Intent.ACTION_DIAL)
i.data = Uri.parse("tel:+34600...")
startActivity(i)

// Compartir texto
val i = Intent(Intent.ACTION_SEND)
i.type = "text/plain"
i.putExtra(EXTRA_TEXT, "Texto")
startActivity(createChooser(i, ""))
```

Layouts y ViewGroups en Android

⭐ ConstraintLayout

RECOMENDADO

Flexible y performante

Restricciones entre vistas

↕ LinearLayout

Orden horizontal

o vertical lineal

Simple y predecible

🔲 FrameLayout

Apila elementos

uno sobre otro

Ideal para fragmentos

↔ RelativeLayout

Posiciona relativo

a otros elementos

o al padre

⊞ GridLayout

Organiza en

cuadrícula de filas

y columnas

📐 ConstraintLayout — El Layout recomendado por Google:

```
<TextView android:id="@+id/tvTitle"
  app:layout_constraintTop_toTopOf="parent"
  app:layout_constraintStart_toStartOf="parent"
  app:layout_constraintEnd_toEndOf="parent"/>
<Button android:id="@+id/btnAction"
  app:layout_constraintTop_toBottomOf="@id/tvTitle"
```

🔍 View vs ViewGroup

View: Elemento básico — Button, TextView, ImageView, EditText

ViewGroup: Contenedor que organiza Views. Todos los layouts son ViewGroups

Un ViewGroup puede contener otros ViewGroups → jerarquía de vistas

Controles UI y Material Design

T TextView

Muestra texto

textSize, textColor

textStyle, fontFamily

EditText

Campo de texto

hint, inputType

maxLines, password

Button

Acción principal

Material Components

onClickListener

ImageView

Muestra imágenes

scaleType: centerCrop

fitCenter, fitXY

☑ CheckBox

Selección múltiple

setOnCheckedChange

isChecked()

RecyclerView

Listas avanzadas

Adapter + ViewHolder

LayoutManager

🎨 Material Design 3 (Material You): Sistema de diseño de Google — Tipografía, Color, Elevación, Movimiento, Componentes estandarizados → material.io/design

Sistema de Recursos en Android

strings.xml

Textos localizables de la app

```
<string
name="app_name">MiApp</string>
<string name="hello">¡Hola,
%1$s!</string>
```

colors.xml

Colores reutilizables en toda la app

```
<color
name="primary">#3DDC84</color>
<color
name="background">#FFFFFF</color>
```

dimens.xml

Márgenes, paddings y tamaños

```
<dimen
name="margin_standard">16dp</dimen>
<dimen
name="text_size_normal">14sp</dimen
>
```

themes.xml

Temas y estilos globales de la app

```
<style name="Theme.MiApp"

parent="MaterialComponents.DayNight
">
  <item
name="colorPrimary">@color/primary<
/item>
```

Localización i18n

Soporte multi-idioma automático

```
res/values/       ← default (EN)
res/values-es/    ← Español
res/values-fr/    ← Français
```

drawable/

Imágenes, vectores e iconos

```
<!-- Vector Drawable -->
<vector android:width="24dp">
  <path android:pathData="..."/>
```

Resumen del Tema 2

Puntos Clave

01	Android nació en 2003 y fue adquirido por Google en 2005. Hoy tiene ~3.000 millones de usuarios en todo el mundo
02	Arquitectura en 5 capas: Aplicaciones · Framework · Librerías/ART · HAL · Kernel Linux. ART usa compilación AOT+JIT
03	Android Studio es el IDE oficial. Se configura con SDK Manager. ADB permite depurar por USB o WiFi
04	4 componentes: Activity (UI), Service (background), BroadcastReceiver (eventos), ContentProvider (datos)
05	Los Intents conectan componentes. Los Layouts organizan la UI. Material Design 3 define el lenguaje visual de Android

ACTIVIDADES PRÁCTICAS

Ejercicios del Tema 2

1 Instalación

Instalar Android Studio, configurar SDK y crear el primer proyecto "Hello World"

2 Exploración

Navegar por la estructura del proyecto y abrir AndroidManifest.xml y build.gradle.kts

3 Emulador

Crear un AVD con distintas configuraciones (Pixel 7 / API 34) y lanzar la app

4 UI Básica

Diseñar layout con ConstraintLayout: TextView, Button, EditText y un ImageView

5 Recursos

Crear strings.xml y colors.xml y referenciarlos desde el layout y la Activity

6 Ciclo de Vida

Implementar Log.d() en todos los métodos del ciclo y observar en Logcat

7 Intents

Crear app que abra el navegador, el marcador telefónico y comparta contenido vía Intent