

Creación de aplicaciones con Google Android

[android](#), [cursos](#)

datos curso

- ONLINE (<https://cibernarium.barcelonactiva.cat/web/guest/ficha-actividad?activityId=823431>)
- Octubre 2017
- Cristian Barrientos Beltrán (<https://github.com/cristianbarri>)

Modulo 1

Sistema operativo Android

- software libre: sistema operativo, middleware, aplicaciones, API's, SDK
 - SDK completo, multiplataforma, emulador, depuración, plugin Eclipse
 - reprovechamiento de componentes
 - gráficos 2D nativos, 3D con librerías
 - SQLite
- enfocado a teléfonos, etc...
- Java + XML
- Interfaces de usuarios complejas
- soporte multimedia, comunicaciones, sensores
- versiones:
 - 1.0, 1.1, 1.5, 1.6
 - 2.0, 2.1, 2.2, 2.3.x
 - 3.x, 4.0.x, 4.1, 4.2, 4.3
 - 4.4
 - 5.0
 - 6.0
 - (otras)
 - <http://developer.android.com/intl/es/about/dashboards/index.html>

Estructura de un proyecto

- AndroidManifest.xml
 - configuración
 - permisos solicitados
 - versión android mínima
 - Activity (pantalla)
 - Servicios
 - Broadcast receivers
 - Content providers
 - Intent-filters
- Proguard.cfg
 - Optimizar/ofuscar código (cambio nombre variable)
 - dificultar ingeniería inversa
 - reducir medida de la APK
 - se ejecuta automáticamente (build) en modo release (para entregar)
- carpeta **src**
 - package

- carpeta **res**
 - carpeta Drawable (imágenes, audios)
 - carpeta Layouts
 - carpeta Values

componentes

- Activities
 - componente principal
 - tiene asignada una vista (layout)
 - java
 - el sistema puede cerrar nuestra activity si necesita memoria
- Intents
 - intención o solicitud de realizar una tarea
 - servicios de comunicación entre componentes dentro de la misma App o con ajenas
- Servicios
 - funcionamiento en segundo plano
 - dos tipos
 - started: se lanza y no vuelve
 - bound: interacción
- Interfaz de usuario
 - view, viewgroups
 - permite la interacción con el usuario
 - UI's muestran las funcionalidades de nuestra app
 - amigables
 - cada una tiene su ciclo de vida (pausar, destruir, etc...)
- Content Providers
 - proveedores de contenido, comunicación con otras apps para hacer uso de los datos
- Broadcast Receivers
 - escuchando eventos, cambios estado teléfono
 - receptor de difusión
 - las modificaciones se notifican con un **intent**

Modulo 2

Lenguajes de programación

- Java:
 - sintaxis c, c++
 - todo son objetos
 - declaración variables
 - sentencias de condición
 - sentencias de control
 - Arrays / vectores
 - Package:
 - clases afines de la aplicación
 - como una biblioteca
 - físicamente representan una carpeta
 - Herencia entre clases
 - public, protected, private
 - ejecución en máquina virtual
 - OOP encapsula conceptos en clases
 - una clase define:

- variables, atributos
- funciones, métodos (getters, setters, otros)
- XML
 - extensible markup language
 - diseñado para transportar y almacenar datos
 - tags por definir
 - autodescriptivo
 - es texto, almacena
 - estricto!
 - estructura jerárquica (árbol)

Activities

- la mayoría están formadas por activities
- cada A corresponde a una vista
- no tienen control sobre su cierre
 - tiene 3 estados: resumed (cima de la pila), paused (visible, sin foco, no interacción), stopped (invisible)
- Heredan desde la clase Activity
- se han de declarar en Manifest
- se crea una PILA de A
- métodos vinculados con la Activity:
 - onCreate
 - onStart
 - onResume
 - onPause
 - onStop
 - onRestart
 - onDestroy
 - este método también es llamado con un cambio de orientación de pantalla

Intents

- elementos de comunicación entre A
- representación de intención o solicitud de una tarea a otra App
 - `startActivity(intent);`
 - `startService(intent);`
- iniciar servicios, llamar a otras actividades, envío de mensajes de difusión
 - `sendBroadcast(intent);`
- 2 tipos:
 - implícitos
 - se le indica la acción y de manera implícita ejecuta la aplicación relacionada
 - explícitos
 - indicamos que componente exacto queremos lanzar
- objeto Intent contiene atributos:
 - Componente: nombre del componente, path de la clase (x.xx.xxx.xxxx), cada punto separa las carpetas donde encontrar
 - solo tiene sentido para intents explícitos
 - `getComponent`, `setComponent`
 - Acción:
 - solo tiene sentido para intents implícitos
 - lista de acciones en web Google
 - Dato: modificar los datos de un Intent
 - `setData`, `getData`

- Categoría: detalles adicionales de la acción ejecutada
 - `addCategory`, `removeCategory`, `getCategories`
 - varias categorías se pueden asignar
- Extras: info de un componente a otro (JSON?), llaves clave, valor

Modulo 3

Listeners

- Ligados a la vista
- 4 métodos
 1. crear un elemento listener y definir la función `*onClick*` (método existente no implementado, uso de `@Override`)
 2. la clase de la activity ejecute el `*onClick*`, haciendo un `implements` en la definición de la clase (no `@Override`)
 3. crear nuestra propia clase Listener, heredando de Listener (para cosas muy específicas que no se puedan aprovechar de los existentes)
 4. a través del XML, definiendo el valor de la función

Activities (II)

- Guardar el estado de las Activities (por cierre no controlado)
- métodos proporcionados por Android
 - `onSaveInstanceState` → llamado antes del evento «onPause»
 - `onRestoreInstanceState` → llamado después del evento «onCreate»
- almacenar valores
 - parejas clave-valores
 - valores primitivos
 - `putString`, `putInteger`, ...
 - `getString`, `getInteger`, ...

Notificaciones

- Toast
 - notificaciones fugaces
 - solo se muestra 1
 - se apilan
- Snackbar
 - versión mejorada del Toast
 - permite añadir acciones
 - requiere de Android Design Support Library (no sirve versiones antiguas de Android)
- StatusBar
 - acciones asociadas (ir a una activity de nuestra app)
 - se puede configurar su tiempo de vida
 - implementación
 - Instanciar `NotificationManager`
 - inicializar nuestra notificación (builder) y inicializar el Intent Explícito
 - generar stack artificial de activities y adjuntarle el Intent
 - generar la notificación de nuestro builder y hacer que se muestre.

Almacenaje

- interno:
 - privados para la aplicación
 - al desinstalar la app, se borran
 - se guardan en raw.res
 - funciones útiles:
 - `getFilesDir()`
 - `getDir()`
 - `deleteFile()`
 - `fileList()`
- externo:
 - memoria que no pertenece a nuestro dispositivo
 - para que pueda acceder a la SD, hay que darle permisos (manifest)
 - todos los usuarios y apps tienen acceso a estos ficheros
- shared preferences
 - guarda los datos de nuestra app
 - primitivos
 - clave-valor
 - se mantienen aunque se destruya la app
 - clase `SharedPreferences()`

modulo 4

SQL

- información guardada en tablas
- columnas, filas
- SQLite Database
- `bd.query(argumentos)`
- SQL
 - `CREATE TABLE...`
 - `DROP TABLE...`
 - `SELECT...`
 - `INSERT...`
 - `UPDATE...`
 - `DELETE..`

SQLite Open Helper

- ayuda para el SQL en Android
- se puede usar para instrucciones SQL
- o ejecutar los métodos al uso
- `CREATE... onCreate, onUpgrade`
- `SELECT... getReadableDatabase, db.query`
- `INSERT... getWritableDatabase, db.insert, values.put(par-valor)`
- `UPDATE... db.update, similar a INSERT`
- `DELETE... db.delete`

Preparación para la venta de apps i AdMob

- Cuenta desarrollador GooglePlay (25\$) de por vida

- Google Wallet para vender apps (gratuita), vincular cuenta bancaria (tarda 3-4 días en validarse)
- AdMob, sistema anuncios, Paypal
 - permite colocar anuncios, Google gestiona los anuncios
 - 3000 visualizaciones, 1€
 - hay que llegar a 70€ para cobrar
- crear aplicaciones útiles
- entra por la vista
- desarrolla el programa, mejora después
- probar en muchos dispositivos
- desarrollar muchas aplicaciones para ver cual triunfa
- formas alternativas de financiamiento: anuncios, sin pagar por la app, y si con el tiempo funciona, de pago

From:

<https://matebcn.eu/> - **miguel angel torres egea**

Permanent link:

<https://matebcn.eu/info:cursos:cibernarium:android>

Last update: **07/07/2018 19:41**

