

JJD311: Desarrollo Avanzado de Apps para Android 5 Lollipop – skill test

Nombre completo: _____ Nota: _____

Q1 – Señale qué afirmaciones son ciertas con respecto a los sistemas de almacenamiento en Android:

- a)
- b)
- c)
- d)
- e)

Q2 – En Android, la Google Maps API Key se necesita para:

- a)
- b)
- c)
- d)

Q3 – La gestión de bases de datos en Android se realiza mediante:

- a) Mongo DB
- b) SQL
- c) SQLite
- d) MySql
- e) Android SQL

Q4 – En referencia a las operaciones de *networking* en Android, señale las opciones correctas:

- a)
- b)
- c)
- d)
- e)

Q5 – El gran inconveniente de usar entidades de tipo `AsyncTask` en Android es:

- a)
- b)
- c)
- d)

Q6 – En el prototipo de una `AsyncTask` del tipo `AsyncTask<String, Integer, Void>`, los argumentos indicados representan:

- a)
- b)
- c)
- d)

Q7 – El método `runOnUiThread(...)` sirve para:

- a)
- b)
- c)
- d)

Q8 – En un `ContentProvider`, el método `query(...)` se emplea para:

- a)
- b)

- c)
- d)

Q9 – A la hora de crear una base de datos en Android,

- a)
- b)
- c)
- d)

Q10 – Señala las opciones verdaderas en referencia a la clase `ConnectionManager`:

- a)
- b)
- c)
- d)
- e)

Q11 – Un `Fragment` es:

- a)
- b)
- c)
- d)

Q12 – Para comunicar un `Fragment` con la `Activity` que lo contiene, debe implementarse:

- a)
- b)
- c)
- d)

Q13 – El ciclo de vida conjunto de una `Activity` que invoca a un `Fragment`, hasta que éste último toma el foco (*foreground*) es:

- a)
- b)
- c)
- d)

Q14 – Un `Loader` en Android sirve para:

- a)
- b)
- c)
- d)

Q15 – Indique las opciones correctas a la hora de hacer un *parsing* a un fichero XML:

- a)
- b)
- c)
- d)
- e)

Q16 – La librería Picasso permite:

- a)
- b)
- c)
- d)

Q17 – En referencia a la inclusión de Google Maps en una aplicación Android, se puede definir un `Marker` como:

- a)
- b)

- c)
- d)

Q18 – En referencia a los mecanismos de seguridad empleados en el desarrollo en Android, podemos afirmar que (márcala aquella que NO es verdadera):

- a)
- b)
- c)
- d)

Q19 – Un `InputStream`, como el devuelto tras establecer una comunicación HTTP, cumple que (señale todas las que apliquen):

- a)
- b)
- c)
- d)

Q20 –

- a)
- b)
- c)
- d)

Q21 –

- a)
- b)
- c)
- d)

Q22 –

- a)
- b)
- c)
- d)

Q23 –

- a)
- b)
- c)
- d)

Q24 –

- a)
- b)
- c)
- d)

Q25 –

- a)
- b)
- c)
- d)

Q1 – ¿Qué función desempeña un `BroadcastReceiver` en Android?

- a) Reacciona a anuncios de difusión
- b) Realiza acciones o labores de procesamiento en 'background'

- c) Permite comunicar datos entre diferentes `Activity`/s
- d) Ninguna de las anteriores

Q2 – ¿Qué significa APK en Android?

- a) Hace referencia a los paquetes de la API de Android (`Android Packages`)
- b) Son las siglas de `Android Packaging Kit`, además de utilizarse como extensión de los archivos de aplicaciones
- c) `Android Prototype Key`, que se utiliza para firmar la aplicación justo antes de subirla al `Play Store`
- d) Ninguna de las anteriores

Q3 – ¿Qué es Android?

- a) Es un conjunto de componentes software (sistema operativo y APIs) especialmente diseñado para dispositivos móviles
- b) Es la denominación del dispositivo móvil propio de Google
- c) Es una máquina virtual, similar a la `JVM` de Java
- d) Ninguna de las anteriores

Q4 – ¿Cuál es el ciclo de vida de los `Service` en Android?

- a) `onCreate()` -> `onStartCommand()` -> `onDestory()`
- b) `onRecieve()`
- c) `final()`
- d) El ciclo de vida de un `Service` es el mismo que el de una `Activity`

Q5 – ¿Que es un `ViewGroup` en Android?

- a) Es la clase base que representa a cualquier otra clase en Android
- b) Representa cualquier `View` que pueda anidar otras vistas
- c) `Layouts`
- d) Ninguna de las anteriores

Q6 – ¿Qué es un `context` en Android?

- a) Es una entidad que guarda información de carácter global sobre la aplicación
- b) Se usa para crear o lanzar otros componentes
- c) En Android hay básicamente dos `context`, el devuelto por el método `getContext()` y el de `getApplicationContext()`
- d) Todas las anteriores

Q7 – ¿Cuántos tamaños de pantalla diferentes hay definidos en Android?

- a) Android da soporte a cualquier tamaño, así que la definición es global
- b) Android tiene tabulados los tamaños de pantalla, así que ningún dispositivo puede salirse de esa clasificación
- c) Android permite los tamaños *small*, *normal*, *large* y los *x-large*
- d) El tamaño es un parámetro que la plataforma Android ni si quiera integra en su definición

Q8 – ¿Cómo se pasan datos de una `Activity` a un `Service` en Android?

- a) Se pueden almacenar datos en una base de datos común y acceder a ellos tanto desde una `Activity` como desde un `Service`
- b) No se pueden pasar datos de una `Activity` a un `Service`
- c) Usando el método `putExtra()` en un `Intent`, y con `setResult()`
- d) A y C
- e) Ninguna de las anteriores

Q9 – ¿Cuál es la utilidad de un `ContentProvider` en Android?

- a) Enviar datos de una aplicación a otra aplicación
- b) Almacenar datos en una base de datos
- c) Permitir que los datos de una aplicación se puedan compartir con otras
- d) Ninguna de las anteriores

Q10 – Indica los componentes principales de una aplicación en Android

- a) `Activity`
- b) `Resources`

- c) Service
- d) BroadcastReceiver
- e) uses-permission
- d) ContentProvider

Q11 – ¿En qué hilo se ejecuta un BroadcastReceiver?

- a) Worker Thread
- b) Main Thread
- c) Activity Thread
- d) Ninguna de las anteriores

Q12 – ¿Cuál es el ciclo de vida de una Activity hasta llegar al *foreground* (foco)?

- a) onCreate() -> onStart() -> onResume() -> onStop() -> onRestart
- b) onCreate() -> onStart() -> onResume() -> onStop()
- c) onCreate() -> onStart() -> onResume()
- d) Ninguna de las anteriores

Q13 – ¿Cómo se pasan datos entre Activity/s?

- a) Intent
- b) ContentProvider
- c) BroadcastReceiver
- d) Ninguna de las anteriores

Q14 – ¿Cuáles de las siguientes afirmaciones son ciertas acerca de los atributos de tipo *margin* y *padding* usados en el diseño de layouts en Android?

- a) El atributo *margin* solo está disponible para las vistas de tipo Layout (LinearLayout, RelativeLayout, etc.)
- b) El atributo *padding* se usa para declarar un cierto offset (en dp o px) en el contenido de un View
- c) A y B
- d) Ninguna de las anteriores

Q15 – ¿Qué método debe emplearse para iniciar una Activity y que esta devuelva valores de retorno?

- a) startActivityToResult()
- b) startActivityForResult()
- c) Bundle()
- d) Ninguna de las anteriores

Q16 – ¿Cómo se puede detener un Service en Android?

- a) finish()
- b) system.exit()
- c) Sólo puede realizarse manualmente
- d) stopSelf() and stopService()

Q17 – ¿Qué es un mensaje *log* en Android?

- a) Los mensajes log se emplean para depurar (*debug*) una aplicación
- b) Es igual que un printf()
- c) Similar a un Toast
- d) Ninguna de las anteriores

Q18 – ¿Cuáles son los posibles valores de retorno (*resultCode*) cuando se llama a `startActivityForResult()` en Android?

- a) RESULT_OK
- b) RESULT_CANCEL
- c) RESULT_CRASH
- d) A y B
- e) B y C

Q19 – ¿Qué es el *Manifest* en una aplicación Android?

- a) Tiene información acerca de los ficheros de layout a usar en la aplicación
- b) Indica únicamente los permisos y *Activity*/s que se integran en la aplicación
- c) Su contenido define completamente una aplicación
- d) Ninguna de las anteriores

Q20 – ¿Cuál es el ciclo de vida de un *BroadcastReceiver* en Android?

- a) `sendIntent()`
- b) `onRecieve()`
- c) `implicitBroadcast()`
- d) `sendBroadcast()`, `sendOrderBroadcast()`, and `sendStickyBroadcast()`.

Q21 – ¿Qué es una *Activity* en Android?

- a) Una *Activity* permite al usuario interactuar con la pantalla y actuar en base a las acciones/pulsaciones que este realiza
- b) Una *Activity* Gestiona el contenido de la aplicación
- c) Una *Activity* alberga los elementos gráficos de la aplicación, y los ordena para presentárselos al usuario
- d) Ninguna de las anteriores

Q22 – ¿Cuáles son layouts disponibles en Android?

- a) `LinearLayout`
- b) `FrameLayout`
- c) `TableLayout`
- d) `RowsLayout`
- e) `RelativeLayout`

Q23 – Selecciona todas las respuestas que se correspondan con valores de retorno del método `onStartCommand()`, implementado por los *Service* en Android

- a) `START_STICKY`
- b) `BOUND_SERVICE`
- c) `START_NOT_STICKY`
- d) `START_REDELIVER_INTENT`
- e) `RESULT_OK`

Q24 – Señala las opciones verdaderas acerca de la gestión de recursos en Android:

- a) Los recursos del proyecto pueden colocarse en cualquier carpeta del árbol de directorios, siempre y cuando se referencien con la palabra clave `resources`
- b) Los recursos por defecto se ubican en carpetas que no tienen ningún modificador como sufijo
- c) Los modificadores (sufijos) empleados en las carpetas que guardan los recursos alternativos deben posicionarse en un orden no aleatorio
- d) La clase `R` almacena referencias a los recursos como identificadores de tipo `long`
- e) Un desarrollador puede combinar los modificadores de recursos como estime oportuno, incluso si no hay ningún terminal en el mercado que pueda presentar la correspondiente configuración

Q25 – Señala aquellas capas que conforman el *Android Stack*:

- a) `Application Framework`
- b) `Applications`
- c) `LogCat`
- d) `Kernel`
- e) `Activity Stack`