Universidad Nacional Autónoma de México

Facultad de Ciencias

Programación de Dispositivos Móviles

Segundo Examen Parcial

Castañon Maldonado Carlos Emilio

Preguntas:

1 **Explica que es un adapter.**

Un adaptador es una clase que convierte un formato de datos a otro, en la lista de elementos visuales en Android, un adaptador es una clase que convierte datos de una fuente de datos a un formato que se puede mostrar en una pantalla.

Hay muchos tipos diferentes de adaptadores en Android. Algunos tipos comunes de adaptadores incluyen:

- ListAdapter: Convierte una lista de datos en una lista de vistas.
- CursorAdapter: Convierte un cursor de datos en una lista de vistas.
- SimpleCursorAdapter: Una subclase de CursorAdapter que facilita la vinculación de un cursor a una lista de vistas.

Los adaptadores se utilizan en muchos lugares diferentes en Android, por ejemplo, se utilizan para mostrar listas de datos en ListViews, para mostrar listas de datos en Spinners, y para mostrar listas de datos en GridViews.

2 **Explica cuales son los pasos para programar una lista visual de elementos en Android a través de la clase ListActivity.** Los pasos a seguir son:

i) Crear un nuevo proyecto Android en Android Studio.
ii) En el archivo de diseño del proyecto, añadir un widget ListView.
iii) Crear un adaptador para vincular los datos al ListView.
iv) Establecer el adaptador en el ListView.
v) Añadir datos al adaptador.
vi) Ejecutar la aplicación.

3 **¿Cuáles son los dos métodos para crear una lista visual de elementos en Android?**

Usando el widget ListView

➢ El widget ListView es un widget incorporado que se puede utilizar para mostrar una lista de elementos.
Para utilizar el widget ListView debemos seguir estos pasos:

i) Lo añadimos al archivo de diseño.
ii) Creamos un adaptador para vincular los datos a ListView.
iii) Establecemos el adaptador en el ListView.

Usando la clase ListActivity

➢ La clase ListActivity es una actividad integrada que se puede utilizar para mostrar una lista de elementos.
Para utilizar la clase ListActivity debemos seguir estos pasos:

i) Extendemos la clase ListActivity en la clase activity.
ii) Anulamos el método onCreate().
iii) Añadimos el widget ListView al archivo de diseño de la actividad.
iv) Creamos un adaptador para enlazar los datos al ListView.

4 **Explica cómo es la programación del componente visual Action Bar en Android.**

Recordemos que la Action Bar es un componente visual en Android que proporciona una barra de herramientas para acciones comunes a una actividad. La Action Bar se puede utilizar para mostrar un título, opciones de navegación y otras acciones.

Para programar la Action Bar debemos seguir los siguientes pasos:

i) Como primer paso la añadimos a su archivo de diseño de actividad.

ii) La inicializamos en su clase de actividad.

iii) Una vez inicializada la Action Bar, podremos agregarle acciones.

Una vez que hayamos añadido acciones a la Action Bar, podremos manejar los eventos que se producen cuando se hace clic en las acciones, a lo que podremos hacer esto sobrescribiendo el método onOptionsItemSelected() en su clase de actividad.
El método onOptionsItemSelected() toma un argumento: el objeto MenuItem que fue presionado, además de que podemos utilizar el objeto MenuItem para obtener el id de la acción en la que se hizo clic.

5 **Explica cómo es la programación del componente visual Navigation Drawer en Android.**

Recordemos que el Navigation Drawer es un componente visual de Android que proporciona un panel lateral para las opciones de navegación. El Navigation Drawer se puede utilizar para mostrar una lista de elementos de navegación, como una lista de pantallas en una aplicación.

Para programar el Navigation Drawer debemos seguir los siguientes pasos:

i) Como primer paso la añadimos a su archivo de diseño de actividad.

ii) La inicializamos en su clase de actividad.

iii) Una vez inicializado el Navigation Drawer, podremos agregarle elementos de navegación.

Una vez que hayamos agregado elementos de navegación al Navigation Drawer, podremos manejar los eventos que ocurren cuando se hace clic en los elementos de navegación.

6 **¿Cómo se pueden cambiar los colores que están definidos por default en la barra de acciones, en la barra de notificación y de la barra de navegación del dispositivo?**

Podemos cambiar los colores predeterminados de la Action Bar, la barra de notificaciones y la barra de navegación editando el archivo styles.xml. El archivo styles.xml se encuentra en la carpeta res/values de el respectivo proyecto Android.

Para cambiar los colores por defecto de la Action Bar, la barra de notificaciones y la barra de navegación, necesitamos encontrar las siguientes líneas de código en el archivo styles.xml:

```
<item name="colorPrimary">@color/colorPrimary</item>
<item name="colorPrimaryDark">@color/colorPrimaryDark</item>
<item name="colorAccent">@color/colorAccent</item>
```

Para cambiar los colores de la Action Bar, la barra de notificaciones y la barra de navegación, basta con cambiar los valores de los atributos colorPrimary, colorPrimaryDark y colorAccent.

Por ejemplo, para cambiar el color de la barra de acciones a azul, hay que cambiar el valor del atributo colorPrimary a `0000FF`.

7 **¿En qué consiste el tipo de almacenamiento SharedPreferences en Android?**

SharedPreferences es una API de almacenamiento clave-valor que permite almacenar pequeñas cantidades de datos en forma de pares clave-valor, SharedPreferences es una buena opción para almacenar datos que deben persistir durante los reinicios de la aplicación, como las preferencias del usuario o el estado de inicio de sesión.

SharedPreferences es un simple almacén clave-valor que se almacena en el sistema de archivos, no es una base de datos y no admite consultas complejas, sin embargo, es muy fácil de usar, y es una buena opción para el almacenamiento de pequeñas cantidades de datos que deben persistir a través de reinicios de aplicaciones.

Para recuperar datos, se utiliza el método getString(), el método getString() recibe dos argumentos: la clave y un valor por defecto.
El valor por defecto se devuelve si la clave no existe en el archivo SharedPreferences.

8 **¿Cuáles son las clases principales que se necesitan para crear y accesar una Base de Datos SQLite en Android?**

Hay tres clases principales necesarias para crear y acceder a una base de datos SQLite en Android:

➢ **SQLiteOpenHelper**

La clase SQLiteOpenHelper es una clase auxiliar que proporciona métodos para crear, abrir y gestionar una base de datos SQLite.
Para utilizar la clase SQLiteOpenHelper, primero necesitamos crear una subclase de clase SQLiteOpenHelper.
En la subclase, necesitaremos sobreescribir el método onCreate() y el método onUpgrade().
El método onCreate() se ejecuta cuando la base de datos se crea por primera vez.
El método onUpgrade() se ejecuta cuando se actualiza el esquema de la base de datos.

➢ **SQLiteDatabase**

La clase SQLiteDatabase es la clase principal para interactuar con una base de datos SQLite.
Para utilizar la clase SQLiteDatabase, primero necesitamos obtener un objeto SQLiteDatabase de la clase SQLiteOpenHelper.
Una vez que tengamos un objeto SQLiteDatabase, podemos utilizarlo para realizar varias operaciones en la base de datos, como crear tablas, insertar datos y consultar datos.

➢ **Cursor**

La clase Cursor se utiliza para representar los resultados de una consulta en una base de datos SQLite.
Para utilizar la clase Cursor, primero hay que llamar al método query() en el objeto SQLiteDatabase.
El método query() devuelve un objeto Cursor.
Ahora, podemos utilizar el objeto Cursor para recorrer los resultados de la consulta.

9 **¿Cómo se inserta un registro en una Base de Datos SQLite en Android?**

Para insertar un registro en una base de datos SQLite en Android, se puede utilizar el método insert() de la clase SQLiteDatabase.
El método insert() toma cuatro argumentos:

- ➢ El nombre de la tabla en la que insertar el registro.
- ➢ Un valor nulo para representar la clave primaria del registro.
- ➢ Un objeto ContentValues que contiene los valores a insertar en el registro.
- ➢ Un valor nulo para representar la estrategia de resolución de conflictos.

Recordemos que el objeto ContentValues es un mapa que contiene los valores que se insertarán en el registro.
Las claves del objeto ContentValues son los nombres de las columnas, y los valores del objeto ContentValues son los valores a insertar en las columnas.

10 **Explica cómo es la programación de archivos en Android para el almacenamiento de datos**

En Android, se pueden utilizar varias opciones para el almacenamiento de datos en forma de archivos, algunas de las opciones más comunes son:

➢ **Almacenamiento interno (Internal Storage):** El almacenamiento interno proporciona un espacio privado y seguro para almacenar archivos específicos de la aplicación. Cada aplicación tiene su propio directorio de almacenamiento interno, al que solo puede acceder la propia aplicación. Se puede crear y leer archivos en este almacenamiento utilizando la clase File y los métodos proporcionados por el contexto de la aplicación.

➢ **Almacenamiento externo (External Storage):** El almacenamiento externo proporciona un espacio compartido que puede ser accedido tanto por tu aplicación como por otras aplicaciones. Este almacenamiento puede ser una tarjeta SD o el almacenamiento interno compartido del dispositivo. Para acceder al almacenamiento externo, es necesario solicitar los permisos necesarios en el archivo de manifiesto de la aplicación.

Ahora, otra opción comúnmente utilizada es SQLite, ya que nos ofrece una forma eficiente de almacenar y administrar datos estructurados en una base de datos dentro de una aplicación.

A continuación un ejemplo paso a paso en SQLite:

I) Creación y apertura de la base de datos:

```
// Crear o abrir la base de datos
SQLiteDatabase database = context.openOrCreateDatabase("mi_basededatos.db",
    Context.MODE_PRIVATE, null);
```

II) Creación de una tabla en la base de datos:

```
// Crear una tabla si no existe
database.execSQL("CREATE TABLE IF NOT EXISTS usuarios (id INTEGER PRIMARY KEY,
    nombre TEXT, edad INTEGER)");
```

III) Inserción de datos en la tabla:

```
// Insertar datos en la tabla
ContentValues values = new ContentValues();
values.put("nombre", "Maximo Decimo Meridio");
values.put("edad", 29);
database.insert("usuarios", null, values);
```

IV) Consulta de datos desde la tabla:

```
// Consultar datos de la tabla
Cursor cursor = database.rawQuery("SELECT * FROM usuarios", null);
if (cursor.moveToFirst()) {
    do {
        int id = cursor.getInt(cursor.getColumnIndex("id"));
        String nombre = cursor.getString(cursor.getColumnIndex("nombre"));
        int edad = cursor.getInt(cursor.getColumnIndex("edad"));
        // Hacer algo con los datos obtenidos
    } while (cursor.moveToNext());
}
cursor.close();
```

V) Cierre de la base de datos:

```
// Cerrar la base de datos
database.close();
```