Md Rahman

1.What are loaders and how do we implement loaders?

2. What is an AsyncTaskLoader?

3. What is a Handler Thread for?

4. What are some common threading restrictions in android?

5. What are thread pools and thread pool executors?

1. Loaders allow you to load data from a content provider or another source of data for displaying in a fragment or fragment activity. Fetching data directly in the activity or fragment will cause a lack of responsiveness, especially when performing a lot of queries on the UI thread. With loaders, we can run seperate threads to prevent this.

To use a loader, we do the following:

Implement the LoaderManager.LoaderCallbacks interface. This interface contains callback methods that get called whenever a loader event occurs.

Create an instance of the Loader Manager, which is an abstract class for managing one or more Loader instances.

A CursorLoader object to load data from a content provider.

2. What is an AsyncTaskLoader?

An abstract loader that provides an AsyncTask. It provides a method loadInBackground() that runs on a separate thread. The results of this method are automatically delivered to the UI thread via onLoadFinished().

3. What is a Handler Thread for?

To understand this, we must understand Looper and Handler classes. A looper is a class used to run a message loop for a thread - it’s not a thread itself. The loop stops after the thread processes the messages. Interactions with a message loop is done with Handlers. Handlers allows you to send and process messages and runnable objects associated with a thread’s MessageQueue. Each handler is associated with a single thread and that thread’s message queue. It will deliver messages only to that thread.

A Handler Thread is a thread that has a Looper, and the looper can be used to create Handlers.

4. What are common threading restrictions in Android?

1. The UI thread should not be used for intensive tasks (like database queries). This can block the whole UI, making the application appear responsive. If blocked for more than 5 seconds, ANR dialog is launched.
2. Because the Android UI toolkit is not thread safe, all manipulation to your user interface must be done from your UI thread, not another thread.
3. When performing a configuration change, a thread that’s already running will not be able to send data back, making it lost and nearly impossible to deallocate that memory location.
4. Other restrictions include Deadlock (when two threads want to access the other’s resource but neither will let go of their own), and Race Condition, where multiple threads compete for the same resource, locking that resource out for any thread.

5. What are thread pools and thread pool executors?

Creating too many new threads can cause serious performance drops, especially when you consider that you must destroy each after you’re done using it. A thread pool reuses previously created threads to execute tasks. The delay it takes to create a thread when a new task arrives is eliminated since the thread already exists.

A ThreadPoolExecutor runs a task from the queue when a thread becomes free.