**Luis Enriquez**

**Week 3 Test**

1. **What are Broadcast receivers?**

A broadcast receiver is a component that responds to system-wide broadcast

announcements.

1. **What are static and dynamic broadcast receiver?**

The static requires a tag declaration in the manifest, not all of his events can be

registered statically, and some of its events may require permissions. As for the

Dynamic, uses “ context.registerReceiver()” to dynamically register an instance, and

must be Unregistered when the activity stops.

1. **How to send a broadcast?**

You can send a broadcast using “ sendOrderedBroadcast(Intent, String) ” method,

which sends broadcasts to one receiver at a time. Also you can send a broadcast

using “ sendBroadcast(Intent) ” method, which sends broadcasts to all receivers in an

undefined order. And lastly, you can use “ LocalBroadcastManager.sendBroadcast”

method, which sends broadcasts to receivers that are in the same app as the sender.

1. **How to make sure other apps can’t listen to your broadcasts?**

Using the local Broadcast Manager.

1. **How to create a broadcast receiver? Explain step by step.**

For creating a Static Broadcast:

* You create the BroadcastReceiver class that extends BroadcastReceiver.
* You define the onReceive Method.
* Then you need to go to the manifest and declare it with the <Receiver> tag, you assign the name of the class, you can enabled it right now, or at runtime, and set the exported tag to true.
* Next is add the <intent-filter> tag, so you can add the actions that the BroadcastReceiver is going to use.

For creating a Dynamic Broadcast:

* You create the BroadcastReceiver class that extends BroadcastReceiver.
* You define the onReceive Method.
* You declare the BroadcastReceiver at the start of the MainActivity Class
* Then you go to the onCreate method of the mainActivity, and instantiate the BroadcastReceiver.
* After that you assign the intent-filter of the BroadcastReceiver in the OnStart method of the MainActivity, add the action, and finally, you register the BroadcastReceiver with the corresponding Intent-Filter.

1. **How to identify a broadcast?**

With the intent-Filter.

1. **What is an intent filter?**

Specifies the types of intents that an activity, service, or broadcast receiver can respond to.

1. **What are the ways you can add an action to the intent/intentFilter?**

On the Android Manifest and on the OnStart Method of the main activity.

1. **What are System Broadcasts?**

These are Broadcast emissions that the system sends for diverse situation that can be present on the hardware, for example, when the device is low on battery, or it is charging, or enters a call.

1. **Can you add multiple action while sending a broadcast?**

Yes you can, using multiple Intent-Filters.

1. **Can you add multiple actions to the intentFilter while receiving a broadcast? How would you handle that?**

You only can have an action per Intent-Filter, but, you can have as many Intent-Filters as you need per receiver. To do so, you only need to add a new intent filter per action like this:

<intent-filter>  
 <action android:name="MY\_SERVICE\_BROADCAST\_1" />  
</intent-filter>

<intent-filter>  
 <action android:name="MY\_SERVICE\_BROADCAST\_2" />  
</intent-filter>

1. **What are services?**

A Service is an Android application component without a UI that runs on the main thread (of the hosting process).

1. **Do all services run on the Main thread?**

No, you can specify them to run on a different thread.

1. **What are the type of services?**

Foreground, Background and Bound Services.

1. **What method each of the services need to implement?**

The onBind Method.

1. **What are services used for?**

The services are used for different tasks, like for example, you could use a service when you need to deliver a value to the user, but the time to return this value will take a while, or it is delivering the value under user control, for example the controls of a music player, or you have a task that must be running constantly, like a GPS.

1. **Does each service needs to be declared in the manifest?**

Yes.

1. **What is a started service? When do we use it?**

A started service is one that another component starts by calling startService(), which results in a call to the service's onStartCommand() method.

1. **What is a bound service? When do we use it?**

A bound service is one that allows application components to bind to it by calling bindService(), so they create a long-standing connection. It generally doesn't allow components to start it by calling startService().

1. **What is a Scheduled service? When do we use it?**

Is a service that is scheduled using the JobScheduler, so it can be done at a given time.

1. **What is a jobScheduler? Why do we use it?**

Is an API for scheduling various types of jobs against the framework that will execute in the application’s own process. We use them when we need to run a task that can run periodically at a given time.

1. **How do we perform a job using the JobScheduler? Explain in detail.**

* First we need to declare the service class in our project, and declare it in the manifest.
* Here, on the service class, using the “onStartCommand: ” method of the service, we will code the dessired tasks to be executed.
* Now that we have our service, we must use it. Back on the desired activity to use it, first we need to create a ComponentName, this will take our created service as a parameter.
* Next we need to declare our JobInfo Builder, which will take as a parameter the Component we just created.
* Now we need to declare our JobScheduler using the getSystemService method, which will take as a parameter the JOB\_SCHEDULER\_SERVICE from the current context.
* Now that we finally have all the ingredients, we use the just created JobScheduler, to Schedule the build of our JobInfo, that we have previously created with the jobservice.

1. **What happens to the bound service when you unbind from it?**

When the last client using the service unbinds from the service, the service will be destroyed, unless, it has also been started with the “startService()” method, in that case, the service will still be alive, until we use the “OnStop” method.

1. **How do we communicate with a bound service? How many ways?**

You can communicate with the IBinder argument of the onServiceConnected() method. The system calls the service's onBind() method to generate the IBinder only when the first client binds. The system then delivers that same IBinder to all additional clients that bind to that same service, without calling onBind() again.

You then declare a Messenger using the IBinder.

1. **How do we use the Binder to communicate with a bound service?**

We need to declare a Messenger that will take the data from the Binder.

1. **What are arguments we need pass to the bindService()**

The IntentService, the ServiceConnection, and the flags (as an Int.).

1. **What is a foreground service?**

A foreground service is a special type of a bound service or a started service, these services perform tasks that the user must be actively aware of.

1. **What is a fragment lifecycle?**

OnAttach(), OnCreate(), onCreateView(), onActivityCreated(), onStart(), onResume(), onPause(), onStop(), onDestroyView(), onDestroy(), onDetach().

1. **What is an activity lifecycle?**

It is the cycle that the each activity takes from its creation, trough its destruction, and is:

* Activity Launched,
* onCreate(),
* onStart()
* onResume()
* onPause()
* onStop()
* onRestart()
* onDestroy()

1. **What is the difference between a merge and a rebase?**

Merge result is identical with that of running git fetch ; git merge or git pull, with merge you commit all your changes to the repositorie. Meanwhile Rebase result is identical with that of running git fetch ; git rebase or git pull --rebase, with Rebase you can rebase the feature branch onto master branch using the following commands.

1. **What is a web service? Name some.**

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. There are many webservices, like the one from the OpenWeatherApp, that gives us the current weather of any place in the world, you can also get webservices from a Flight company to see available flights, and the same goes for hotels.

1. **What are the verbs used in REST service?**

POST, GET, PUT, PATCH, DELETE

1. **How can you make a REST call in Android without third party libs? Name the classes.**

* You need to create an IntentService that will handle the requests.
* On the Intent you need to get the url of the call, and use it to stablish a connection with the HttpURLConnection method with the provided URL.
* After that you need to declare an InputStream with the getInputStream Method.
* Now that you have the stream, you must create a scanner to read the data, and use a stringbuilder to take the data from the scanner.
* Now that we have the info, we create a new intent and pass the data and send a broadcast.

1. **What happens when you make a network call on the main thread?**
2. **Explain how you can make a REST call using okhttp3?**
3. **How can you make synchronous/asynchronous call using okhttp3?**
4. **How to add headers/query parameters to a request in okhttp?**
5. **What is Retrofit? How do you setup RetrofitHelper class?**
6. **How to use dynamic URLs in Retrofit?**
7. **What is an interceptor? How do we use an interceptor in Retrofit?**
8. **How can you use different verbs in Retrofit?**
9. **What are the ways to transform JSON response to DataBean classes?**
10. **How can we serialize a REST JSON response to the data bean classes?**
11. **Can you add the serialization library to the Retrofit instance? How?**
12. **How to cache the response from a rest call?**