Name : Noor Fabi Shah Safa
ID : 2019-1-60-060

Mid-2

Ans-No-1(a)

We register a broadcast receiver ~~statically~~ dynamically. statically onReceive() works on background. When the App will come on frontend then the static Register will not work. As when the ~~app~~ app will be active Android system will control the page. So other background will not be highlighted. Only that activity will be ~~open~~ highlighted which ~~it~~ is open. Thats why we need dynamic register. When the ~~act~~ app is active we need to dynamic register on ~~onRe~~ onResume(). Otherwise if we need to send only notification then we can statically register on Android Menifest in <<receiver>>, in <intent filer> we need to register action. We can say Example -

① Statically - Suppose in messenger app if the app is closed then it will show a notification if someone give messoge. In contact list it will also show a notification. Intent filter is have its action. So Android system will check and match it. So CustomReceivers onReceive() function will Invoked, so it will get the intent and show the notification.

(ii) Dynamically- Suppose we are in messengers inbox. If someone message us and we are in that persons inbox then it will not show any notification. It will direct give the message in that inbox. We use registerReceiver(mReceiver, mIntentFilter) here. Here we have to give mReceivers object. and in mIntentFilter we have to give the filter which is add in Android menifest. After Register in onResume() we have to unregister it in onPause() state.

Ans-No-1(b)

Yes we can register both types of receivers on the same onclroid application.

Ans-No-1(c)

Yes an application need to register for receiving system broadcasts.

Ans-No - 3(a)

Logo and icon of app → internal storage
sellers photo → internal storage
brand logo → internal storage
icons of item categories and items → cloud storage
sellers login and registration info → Shared Preferences
product Information → SQLite database.

Ans-No - 3(b)

For Logo, sellers photo and icons I prefer internal storage as normally these logos and icons are not frequently changable. So we can store it in embadded store so that the data will not be removed if the SSD of phone will removed. The data will be on the phone for the long time as internal storage are not removable. It will not removed data if the app is uninstalled.

For Item categories and items icons I prefer cloud storage. mainly Remote server. As these icons will change freequently. It will take loads many times. In server, if the data is updated the real time database will upload it. and the other connected device will autometically get the updated

information. It will happen live.

for sellers login and Registration I prefer shared Preference as it stored premitive datas. Its used for lightweight usage. The data will save until the app is not uninstalled. If we shut down or restrart phone the data will netup. for product info I prefer SQLite as it stored structured data privately so that other app can not access this data. so it will maintain the privacy of the items information.

Ans-No-4 (a)

The main reasons for an application being unresponsive if we run long running tasks in UI thread. Then the screen will be frozen on hanged. If any task is greater than 5sec then it will freeze the screen. Also if we forcefully stop the app then it kills the current instance of the app quickly. So it become unresponsive. If we do long running task on UI thread the CPU task. gets busy. So it can not complete other instructions. So it becomes unresponsive. If we do UI task on worker thread it will also show errors.

Ans. No - 4(b)

To avoid unresponsive behaviour, we need to do long running task on Worker Thread. But in Worker thread we can not access UI components. So we need message handler which will give message to UI thread and UI will update data using the information. There are some techniques - such as -

(i) Access UI thread from other threads - In other thread Activity.runOnUiThread (~~Runn~~ Runnable) function will be called. We need to use handler to update view. If we want to ~~handle~~ update immediately then we have to use View.post (Runnable). If we want to delay some time then we have to use View.postDelayed (Runnable, long). Here UI thread first works, then Worker thread, then again UI thread.

(ii) Async Task - Async class is already implemented. We have to Extend it. We use onPostExecute(), OnPreExecute() onProgress() in UI thread. and ~~on~~ doInBackground() in worker thread. But now this method is depricated.

(iii) Executor and mainLooper - It is newer ~~vers~~ and Encouraged version. Here we use executor to create UI thread object in worker thread and handler for after downloading UI thread handler post it.

Ans-No-5 (a)

We can ~~el~~ enforce the android system to free allocated memory using Least Recently used (LRU) rules. Here the least one will killed first. In our Example Android system will kill ~~facebook~~ ston newspaper first at it used very early at 8am then it will kill ~~no~~ ~~clan newspaper, then~~ event management application and then it will kill facebook app as it again used 6:05 am after 7am.

Ans-No-2(a)

External storage will I select.

2(b)

External storage has big size to store data and we can also store it presistenly. It is removable. and also it can be accessed easily which are needed in semi-structured data. We can even store 128GB data in this storage. Thats why I prefer External storage.

2(c)

External storage is removable and can access data other devices from here easily. We can save semi-structured data like xml files here easily. Very large files can stored here. For structured data we need privacy. So we can use SSD card of our own to save the datas safely. Such as we can save stock information, dates on our SSD so that we can presistantly store them in large amount. We can access it in our own laptop also if we want.

Ans-No-5(b)

No it can kill download operation on music app. The android system can try downloading alive if it kill the ~~lecture note~~ music ~~app~~ & . & Service priority is 3. So if android system wants to complete the download then it have to kill music application to free the space.

Ans-No-1(d)

We can assure every player will get 2 minutes using intent actions. We ~~can use action~~ mention in intent that it will invoke after 2 minutes then ~~the~~ using time actions, so the ~~g~~ turn will be maintained. In app we have to give the intent permission also.