Solucio Examen Final Android

Projecte d'Enginyeria del Software (Universitat Politècnica de Catalunya)

Examen Final Android

Pregunta 1.
Android permet crear threads en les aplicacions. Per quins motius?
a. Per augmentar la concurréncia.
b. Per augmentar el paral.lelisme de l'aplicació i anar més rapid.
c. Per evitar que l'aplicació es bloquegi quan ha d'executar codi que suposa esperar de forma indefinida.
d. Totes les afirmacions són correctes

Raona la respostes.
Totes les afirmacions són correctes.
Com Android tan sols permet modificar i interaccionar a l'usuari des del "MainThread" o thread principal, si aquest queda bloquejat per una operació que implica una espera indefinida la usabilitat de l'aplicació disminueix, a l'usuari li fa l'efecte que l'aplicació s'ha bloquejat. És per aquest motiu que totes les operacions bloquejants les permet realitzar en threads workers sense problema, aquestes s'executen de forma concurrent amb el thread principal, d'aquesta manera la resposta de l'app 'es àgil i no bloquejant.

Tenir més d'un thread sempre augmenta el paral·lelisme sempre i quan tinguem més d'una CPU en el nostre computador i aquesta estigui lliure.

Tenir threads aumenta la concurrencia per definició.

Pregunta 2.
En quins quatre principis es basen les tecnologies agils?

1. Valorar més els individus i les seves interaccions que als processos i les eines
2. Valorar més el programari funcionant que la documentació exhaustiva
3. Valorar més la col·laboració amb el client que la negociació contractual
4. Valorar més la resposta davant el canvi que seguir un pla

Pregunta 3.
La llibreria per fer peticions HTTP des d'una aplicació Android que has utilitzat en el teu projecte és *HTTPUrlConnection*. A continuació tenim un codi que permet fer una connexió HTTP a un servidor implementat amb play framework.

```
1 String query =
String.format("http://192.168.1.145:9000/Android/Login");
URL url = new URL(query);
HttpURLConnection conn = (HttpURLConnection) url.openConnection();
conn.setReadTimeout(10000 );
conn.setConnectTimeout(10000);
conn.setRequestMethod ("POST"):
conn.setDolnput(true);
conn.setDoOutput(true);
1 conn.connect();

2 String params = "nom=lola&password=lolap";
```

```
OutputStream os = conn.getOutputStream();
BufferedWriter writer = new BufferedWriter(
new OutputStreamWriter(os, "UTF-8"));
writer.write(params.toString());
writer.flush();
writer.close();

2 os.close();

3 stream = conn.getinputStream();

BufferedReader reader = null;
StringBuilder sb = new StringBuilder();
reader = new BufferedReader(new InputStreamReader(stream));
String line = null;
while ((line = reader.readLine()) != null) {
sb.append(line);
}
3 result = sb.toString();
```

Com modificaries el codi per fer la mateixa petició amb el mètode GET?

Tans sol hi ha que fer les modificacions següents (subrallat):

```
1 String query =
String.format("http://192.168.1.145:9000/Android/Login?nom=lola&password=lolap"); // afegim els paràmetres del servei a la petició
URL
URL url = new URL(query);
HttpURLConnection conn = (HttpURLConnection) url.openConnection();
conn.setReadTimeout(10000 );
conn.setConnectTimeout(10000);
conn.setRequestMethod ("GET"); // enlloc de POST GET
conn.setDolnput(true);
conn.setDoOutput(true); //Aquesta instrucció no és necessària
conn.connect();

// Tot el bloc 2 desapareix
2 String params = "nom=lola&password=lolap";
OutputStream os = conn.getOutputStream();
BufferedWriter writer = new BufferedWriter(
new OutputStreamWriter(os, "UTF-8"));
writer.write(params.toString());
writer.flush();
writer.close();
2 os.close();
```

Pregunta 4.
Respecte al cicle de vida d'una Activity en Android:, quina de les seguents afirmacions son certes.

1. El metode OnCreate es mostra la configuració bàsica de l'activitat, com ara declarar la interfície d'usuari (definida en un fitxer XML de disseny).
Falsa
No es mostra, pero si que s'estableix el vincle entre el codi de lActivity i el disseny de la mateixa

2. Abans de passar a primer pla, una Activity passa almenys un cop per les etapes de Create, Start i Resume.
Certa

3. El sistema pot parar l'execució (matar) una aplicació Android si necessita memòria per executar aplicacions més prioritàries
Falsa.
Pot matar Activities pero no aplicacions

4. En primer pla només podem tenir una Activity com a màxim en un moment determinat.
Certa
Perquè l'Activity en primer pla te assignats els dispositius teclat i pantalla que son dispositius que no es poden compartir, per tant sols podem tenir una Activity en primer pla pero moltes en segon pla.

Raona les respostes

Pregunta 5.
Tenim accés al manifest d'una aplicació:

```
<manifest
xmlIns:android="http://schemas.android.com/apk/res/android"
package="pes.upc.lo
ginthreadasynctask">

<uses-permission android:name="android.permission.INTERNET"/>

<application android:usesCleartextTraffic="true"
android:fullBackupContent="true" android:ico
n="@mipmap/ic_launcher" android:label="@string/app_name"
android:roundicon="@mipmap/
ic_launcher_round" android:supportsRtl="true"
android:theme="@style/AppTheme">

<activity android:name=".ActivityC">
<intent-filter>
```

```
<action android:name="android.intent.action.MAIN"/>
<category android:name="android.intent.category._LAUNCHER"/>
</intent-filter>
</activity>
<activity android:name=".ActivityB">
</activity>
<activity android:name=".ActivityA">
</activity>
</application>

</manifest>
```

Contesta les preguntes següents:

1. Quantes pantalles diferents hi han?
Si per pantalla considerem un a Activity, llavors en podem tenir 3.

2. Quina és la primera pantalla que s'executa en primer pla?
ActivityC

3. En quin llenguatge esta escrit el fitxer manifest?
En xml

4. De que serveix tenir l'atribut android:usesCleartextTraffic="true" ?
Ens permet enviar dades per la xarxa sense que estiguin codificades i per tant sense seguretat.

Pregunta 6.
Quants threads s'executen de forma concurrent com a màxim en el codi següent:

```
@Override

protected void onCreate(Bundle savedinstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

intA=0;

new Thread(new Runnable() {
InputStream stream = null;
String str ='';
String result = null;
Handler handler = new Handler();
public void run() {
try {
// Codi del thead
A=1;
```

```
} catch (Exception e) {
e.printStackTrace();

}
}
}).start();
while (A==0) {
Log.i("Debug", "Esperant que pugui passar”);

}

new Thread(new Runnable() {
InputStream stream = null;
String str ='';
String result = null;
Handler handler = new Handler();
public void run() {
try {
/{ Codi del thead
} catch (Exception e) {
e.printStackTrace();
}
}
}).start();

}
```

Raona la resposta.

Com a màxim podem tenir tres threads concurrents, el *Main Thread* o thread princial i els dos workers que s'han creat amb *new thread()*.