Written Test | Android module – 2018-02-09 ("Recurso")

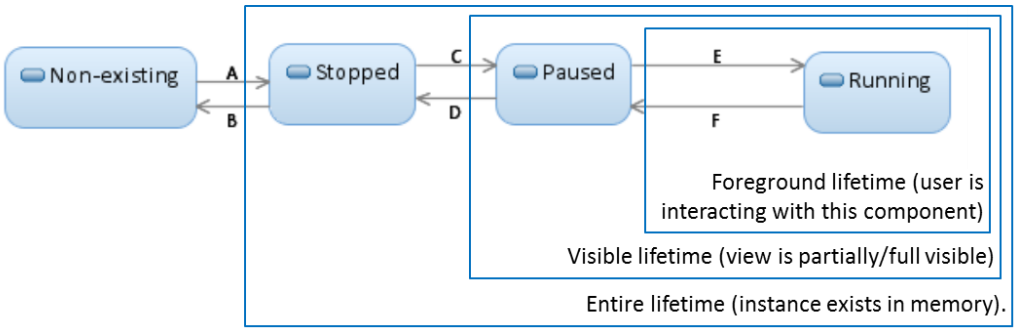
Q1.

The diagram shows a state machine that can be applied to the Android Activity component.

a) Relate the transitions in the state machine with the moment(s) a given layout, specified in a XML file, is expected to be prepared (graphic elements are created) and disposed (views are discarded).

b) How can a programmer open Activity B in response to a user action in Activity A, passing parameters (e.g.: when user click a button in Activity A you should open Activity B, passing the content of a text field)?

c) Which sequence of transactions (from the diagram) are associated with the previous programming case (with respect to Activity A and Activity B)?



Q2.

Consider the following excerpt from an Android manifest.

a) What component-specific concerns, if any, should the developer take when implementing the PhoneCallStateReceiver class?

b) Receivers are associated with an event-oriented programing style. Can this architectural-pattern be use to implement the programming feature describe in previous questions Q1.b) ? Provide a detailed answer.

```
<receiver android:name=".PhoneCallStateReceiver">
  <intent-filter>
    <action android:name="android.intent.action.PHONE_STATE"/>
    <action android:name="android.intent.action.NEW_OUTGOING_CALL"/>
  </intent-filter>
</receiver>

<receiver android:name=".SMSReceiver">
  <intent-filter>
    <action android:name="android.provider.Telephony.SMS_RECEIVED" />
  </intent-filter>
</receiver>
```

Q3.

Observe the following code excerpt (the lines were number for reference).

a) Describe the two ways to add a Fragment to an Activity. Which one is used in this example?

b) Explain the intended use of the three parameters in the call *transaction.replace* (line #3).

c) Well-programmed Fragments would favor reusability. How can a Fragment J communicate with another Fragment H in the same Activity, while preserving its ability to be reused in future Activities (which may use J but not H)?

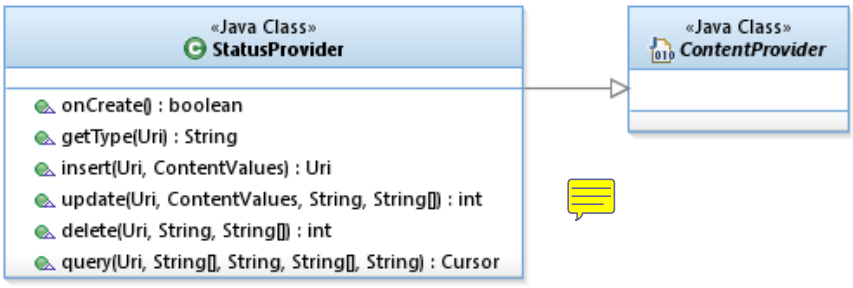
```
1.      MediaBrowserFragment fragment = new MediaBrowserFragment();
2.      FragmentTransaction transaction = getFragmentManager().beginTransaction();
3.      transaction.replace(R.id.container, fragment, FRAGMENT_TAG);
4.      if (mediaId != null) {
5.          transaction.addToBackStack(null);
6.      }
7.      transaction.commit();
```

Q4.

The StatusProvider class, presented in the diagram, implements a ContentProvider.

a) Several methods in the StatusProvider make use of a URI as the first parameter. What is the meaning/content of this parameter?

b) What kind of data storage mechanism is provided in the base implementation of the ContentProvider component? Which choices does the programmer have in this context? Provide a detailed answer.



Q5.

The following excerpt shows parts of the manifest file for a given app.

a) How could the component “SettingsActivity” start the component “StorageService”? Explain your assumptions/options and show the relevant code.

b) Which is the title that is likely to be shown (on the top bar) when of the PhoneInfoActivity is active and visible?

c) Which components will be started in a different (separate) process?

```
 […]
        <uses-permission android:name="android.permission.BLUETOOTH" />
        <uses-permission android:name="android.permission.INTERNET" />
[…]
        <service
            android:name=".services.StorageService"
            android:enabled="true"
            android:exported="true"/>
        <activity
            android:name=".SettingsActivity"
            android:label="@string/title_activity_settings"/>
        <receiver
            android:name=".br.GlobalBroadcastReceiver"
            android:enabled="true"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.BOOT_COMPLETED" />
                <action android:name="android.bluetooth.adapter.action.STATE_CHANGED" />
                <action android:name="android.intent.action.BATTERY_CHANGED" />
            </intent-filter>
        </receiver>
        <activity
            android:name=".PhoneInfoActivity"
            android:label="@string/title_activity_phone_info"
        </activity>
        <service
            android:name=".services.CheckUpService"
            android:enabled="true"
            android:exported="true"/>
        <activity
            android:name=".TargetActivity"
            android:label="@string/app_name"
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
[…]
```

Q6.

The following statements were taken from developer.android.com: “There are simply two rules to Android's single thread model: (1) do not block the UI thread; (2) Do not access the Android UI toolkit from outside the UI thread”.

a) What is the "UI thread"?

b) In the context of the stated “rules”, explain a programming strategy to address the problem of getting a resource from a remote API (may be affected by latency), while keeping the possibility to update a progress bar in the UI (assume that you will get the resource in several segments).

Q7.

Consider the following programming problems, related to the implementation of an application that allows the user to order parcel delivery services in his mobile device. The application allows the user to create a new order, define the package and the destination characteristics, and track the ongoing delivery.

Describe in detail the implementation strategy/options that you would recommend for the following programming problems, related to this example, as if you were explaining to a junior programmer. You do not need to show code (but you may).

a) "The first screen should use a composition of three Fragments, if running in a tablet, or just the order creation Fragment, if running on a standard *smartphone*."

b) "When creating the new order, the app should assume the current location of the user as the default pick-up location."

c) "The end-user should track the changing position of his/her package in a map and view regular updates of the actual location."