Rushabh Shah
CPRE 388
Section F

HW 2

1. Question 1
    a. What is the design decision (as discussed in class) behind onResume → onPause → onResume loop in the activity lifecycle? Support your answer with examples of activity transition scenarios.

➔ The onResume → onPause → onResume loop is designed to enhance user experience by saving all current state of the UI and dynamic content entered by user when onPause is triggered. This would reduce amount of computation needed when user returns to this activity. For example, suppose a user is watching a video on youtube. If meantime, he receives a pop up notification from a messaging application. In this case, onPause is called to ensure the video does not continue to play in background while user interacts with the popup. Thus in the onPause callback, we would capture the time/duration that user has completed and pause the video for user, until he gets back to this activity.

    b. How does onResumeA → onPauseA → onResumeB → onPauseB → onResumeA …. transition affect user interactivity when user switches between two activities A and B, back and forth? Note that other callbacks onStopA, onStopB which may be called are not of concern here.

➔ For both activities, if Bundles are not saved while switching back and forth, user may experience lag / delay in application response.

    c. onStop() is called right after onPause() callback if an activity is completely obscured. Why have two callback functions instead of merging it into one call back onPause_Stop()?

➔ As the example of video application is stated in answer 1.a, if user receives a notification while he/she is watching a video. User does not necessarily have to interact with that notification right away. User could just cancel/ignore that notification and continue watching the video. In this case, in the absence of onPause method, android framework would need to perform onRestart → onStart → onResume to get to the video activity. And during this computation, dynamic data, entered by user, might get lost. onPause eliminated the need of that extra computation of UI and user data.

    d. An app could be still in memory after onPause() or onStop() provided low memory threshold is not triggered. What state should be saved by the app in these two callback functions? What state is saved by the Android framework in these two callback functions? Explain the mechanism used by the framework to save the state of an activity. Provide an example app or activity and relate your answer to the actions in the activity/app.

➔ When onPause is triggered, android framework will stop all animation to release CPU resources and commit any unsaved changes such as an email draft. When onStop event is triggered, android framework will kill the activity. Thus in onStop callback, it is necessary to save all user data in (Bundle savedInstanceState) and perform writes to database if applicable. If user was composing an email and suddenly wanted to switch to another activity / application, the running draft of that email should be saved in savedInstanceState and, if applicable, in the database as well. Since user is opening another activity, all data in this activity will lost in the case user quits this application.

    e. How does the activity life cycle change for the above scenarios? What changes have to be incorporation into the life cycle? Answers should provide scenarios where changes will apply and clearly argue for such changes.

➔ In the case that low memory threshold is not triggered and onPause or onStop callback is invoked, we could add a functionality called onPreserved that will be invoked before onStart, onRestart and

onResume. Thus, if we have enough memory and if we are able to successfully save data as user left the activity last time, we can eliminate much computation and get straight to the activity from that saved instance in onPreserved.

f. Give two scenarios for an activity to get killed by the framework. What are the differences between the two scenarios?

➔ 1. User manually quits the application. (Predictable)
2. When memory shortage occurs. (Unexpected and unpredictable)
The first scenario is common and expected, whereas termination due to memory shortage will anytime the system is low on memory, which is not predictable. In the latter, application might even lose some data, depending on the severity of the situation.

g. What is the purpose of onDestroy callback function?

➔ This is called when the application is shutting down and it will be unloaded from memory by android framework. This is typically called when memory shortage occurs and framework needs to kill the application in order to gain memory.

2. Design the following layout (XML description) using LinearLayout and/or GridLayout. Each alphabet button displays an image for that alphabet. For example, Button A will display an image of apple etc.

➔ See the end of this file:

3. Inflate the above layout in the activity onCreate function and write a shared callback function using setonClickListener to implement the logic of the alphabet buttons for displaying images.

➔ See the end of this file:

4. Buttons B1 and B2 have onTouchListener registered with the framework and overlap in the layout shown below. Which button will receive the touch event in the overlapping portion shown in the figure below? Explain the sequence of events resulting in your answer.

➔ When user touches overlap, framework will notify in the order of the hierarchy. Assuming B1 is in the parent layer and B2 is in children layer of the hierarchy, framework will notify B1's touchListener, if B1 consumes the event, it will return. Otherwise if it does not consume this event, it will go down the hierarchy to B2. Similarly, if B2 does not consume the event it will return to the main view.