Lab 13.1 – Application Fundamentals

Overview:

In this lab you will utilize many of the fundamental application components of the Android architecture. You will use the AlarmManager to schedule jobs, the BroadCast Receiver to respond to an alarm and start a Service, and the Notification manager to alert the user to new data.

Step	Description
1.	Create a utility class to hold notifiction related methods. Create method to add an alarm for scheduling the event update to be run at a regular interval.
a.	Create a new packaged called "com.garagze.notification". In the package, create a new class called "EventUpdateUtils.java".

b.	In "EventUpdateUtils" create a method called "scheduleEventUpdate". This method will be called when we want to set an alarm to run the update.
	
	`public static void scheduleEventUpdate(Context context) {`
	`  Log.v(TAG, "Creating alarm for event update");`
	
	`  Intent intent =`
	`  new Intent(context, EventUpdateBroadcastReceiver.class);`
	
	`  int requestId = (int) System.currentTimeMillis();`
	`  PendingIntent pendingIntent =`
	`  PendingIntent.getBroadcast(context.getApplicationContext(),`
	`  requestId, intent,`
	`  PendingIntent.FLAG_UPDATE_CURRENT);`
	
	`  // Get a reference to the alarm manager`
	`  AlarmManager alarmManager =`
	`  (AlarmManager) context.getSystemService(Context.ALARM_SERVICE);`
	
	`  // We want the alarm to go off 15 seconds from now.`
	`  Calendar startTime = Calendar.getInstance();`
	`  startTime.setTimeInMillis(System.currentTimeMillis());`
	`  startTime.add(Calendar.SECOND, 15);`
	
	`  // Schedule the alarm, the alarm should repeat every 30 seconds`
	`  int repeatSeconds = 30;`
	`  alarmManager.setRepeating`
	`  (AlarmManager.RTC, startTime.getTimeInMillis(),`
	`  repeatSeconds * 1000, pendingIntent);`
	`}`
	
	This code will contain a compiler error until you create the "EventUpdateBroadcastReceiver" is a later step.

c.	In "GaragzeApplication" schedule the event update in the "onCreate" method. `EventUpdateUtils.scheduleEventUpdate(context);`

2.	Create the BroadcastReceiver class that will be run by the alarm manager.

a.	In the package "com.garagze.notification" create a new class called " EventUpdateBroadcastReceiver" which extends " android.content.BroadcastReceiver". Override the "onReceive" method which will be executed when the intent is scheduled by the alarm manger.
	
	This method will start a service which will update the events. Do not execute an AsyncTask directly in this method since the AsyncTask will be killed once the BroadcastReceiver is done executing which is immedietly after it would start the AsyncTask.
	
	`@Override`
	`public void onReceive(Context context, Intent intent) {`
	`  Log.v(TAG, "Starting event update broadcast receiver");`
	`  Intent myIntent = new Intent(context, EventUpdateService.class);`
	`  context.startService(myIntent);`
	`}`
	
	The code will not compile until you create the "EventUpdateService".

3.	Create the service which will run the event update.

a.	In the package "com.garagze.notification" create a new service class called "EventUpdateService" and extend "IntentService"
	
	`public class EventUpdateService extends IntentService {`
	
	`  private static final String TAG =`
	`  EventUpdateService.class.getSimpleName();`
	
	`  public EventUpdateService() {`
	`  super("EventUpdateService");`
	`  // TODO Auto-generated constructor stub`
	`  }`
	
	`  @Override`
	`  protected void onHandleIntent(Intent intent) {`
	`  Log.v(TAG, "Running method onHandleIntent");`
	`  List<Event> events = EventService.getNewEvents(this);`
	`  for (Event event : events) {`
	`  Log.v(TAG,`
	`  "Creating notification for event: " + event.getStreet());`
	`  EventUpdateUtils.createNotification(this, event.getStreet());`
	`  }`
	`  }`
	`}`

4.	Create a new utility method for creating a notification.

a.	In "EventUpdateUtils" add the following method.
	
	`public static void createNotification`
	`  (Context context, String eventStreet)`
	`{`
	
	`  Log.v(TAG, "Creating notification");`
	`  //Get a reference to the NotificationManager:`
	`  String ns = Context.NOTIFICATION_SERVICE;`

```
        NotificationManager mNotificationManager =
          (NotificationManager) context.getSystemService(ns);

        // Instantiate the Notification:
        int icon = R.drawable.notify;
        CharSequence tickerText = "New garage sale at " + eventStreet;
        long now = System.currentTimeMillis();
        Notification notification = new Notification(icon, tickerText, now);

        // Causes notification to be deleted when it is selected
        notification.flags |= Notification.FLAG_AUTO_CANCEL;

        // Define the Notification's expanded message and Intent:
        CharSequence contentTitle = "New Garage Sale";
        CharSequence contentText = "Sale at " + eventStreet;
        Intent intent = new Intent(context, MainActivity.class);
        int requestId = (int) System.currentTimeMillis();
        PendingIntent contentIntent =
           PendingIntent.getActivity(context, requestId, intent, 0);
        notification.setLatestEventInfo
           (context, contentTitle, contentText, contentIntent);

        // Pass the Notification to the NotificationManager:
        int notificationId = (int) System.currentTimeMillis();
        mNotificationManager.notify(notificationId, notification);

    }
```

5. Run the application and verify that notifications have been created.