Rajasana Wilhoit

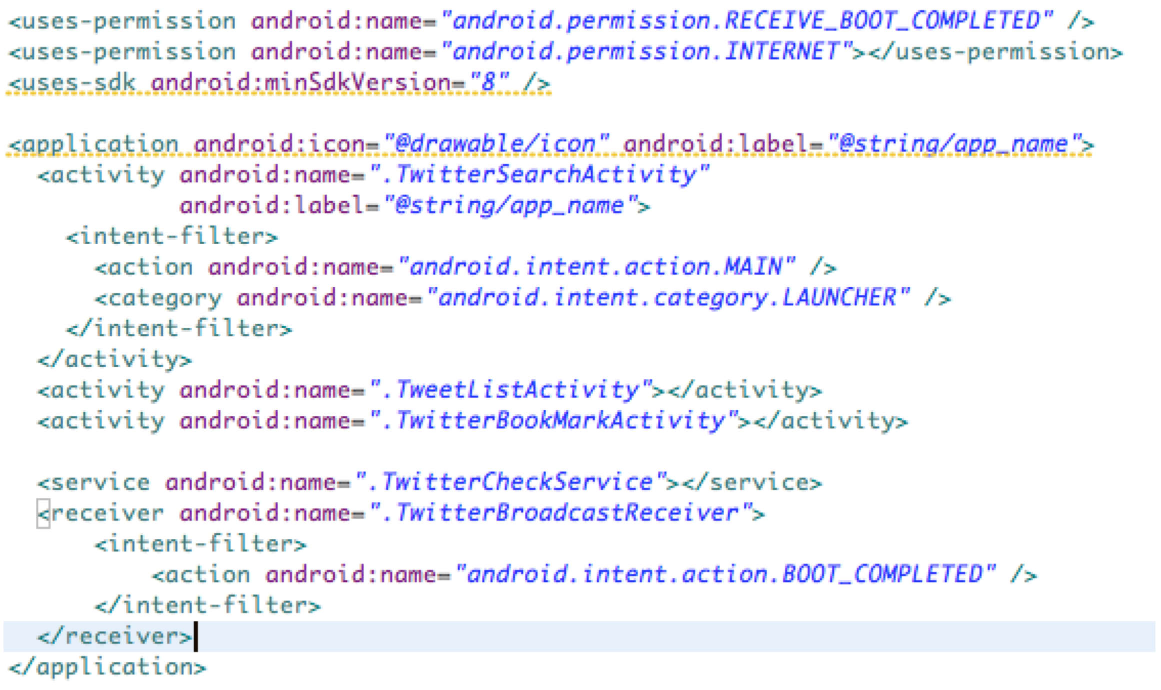
19 February 2013

Mobile Platforms and Development Environments

Professor Halel

Lab 2 ReadMe

AndroidManifest.xml



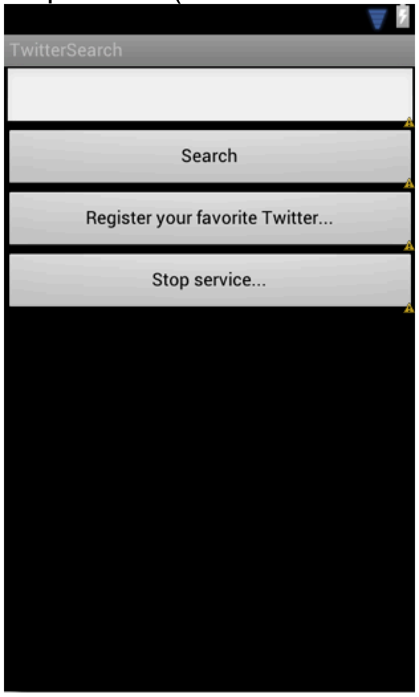
```xml
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.INTERNET"></uses-permission>
<uses-sdk android:minSdkVersion="8" />

<application android:icon="@drawable/icon" android:label="@string/app_name">
  <activity android:name=".TwitterSearchActivity"
            android:label="@string/app_name">
    <intent-filter>
      <action android:name="android.intent.action.MAIN" />
      <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
  </activity>
  <activity android:name=".TweetListActivity"></activity>
  <activity android:name=".TwitterBookMarkActivity"></activity>

  <service android:name=".TwitterCheckService"></service>
  <receiver android:name=".TwitterBroadcastReceiver">
      <intent-filter>
          <action android:name="android.intent.action.BOOT_COMPLETED" />
      </intent-filter>
  </receiver>
</application>
```

For this project, I added the new TwitterBookMarkActivity activity. I then added the TwitterCheckService service, which does most of the work in this project. I also added the broadcast receiver TwitterBroadcoastReceiver for when the user turns off the device.

TwitterSearchActivity.java

```
final Button registerButton = (Button) findViewById(R.id.registerButton);
registerButton.setOnClickListener(new View.OnClickListener() {
  public void onClick(View view) {
    Intent intent = new Intent();
    intent.setClass(TwitterSearchActivity.this, TwitterBookMarkActivity.class);
    System.out.println("Clicked register button.");

    startActivity(intent);
  }
});

final Button stopButton = (Button) findViewById(R.id.stopButton);
stopButton.setOnClickListener(new View.OnClickListener() {
    public void onClick(View view) {
      System.out.println("Clicked stop button.");
      if(TwitterBookMarkActivity.getTwitterServiceIntent() != null)
      {
          Toast.makeText(TwitterSearchActivity.this, "Service Stopped.", Toast.LENGTH_SHORT).show();
          System.out.println("DEBUG: Calling stop service!");
          stopService(TwitterBookMarkActivity.getTwitterServiceIntent());
      }
      else
      {
          System.out.println("DEBUG: NOT Calling stop service!");
          Toast.makeText(TwitterSearchActivity.this, "No Service Running", Toast.LENGTH_SHORT).show();
      }
    }
  });
```

To the TwitterSearchActivity I just added the register button, which takes you to the register page, and a stop service button, which will call stopService(TwitterBookMarkActivity.getTwitterServiceIntent())



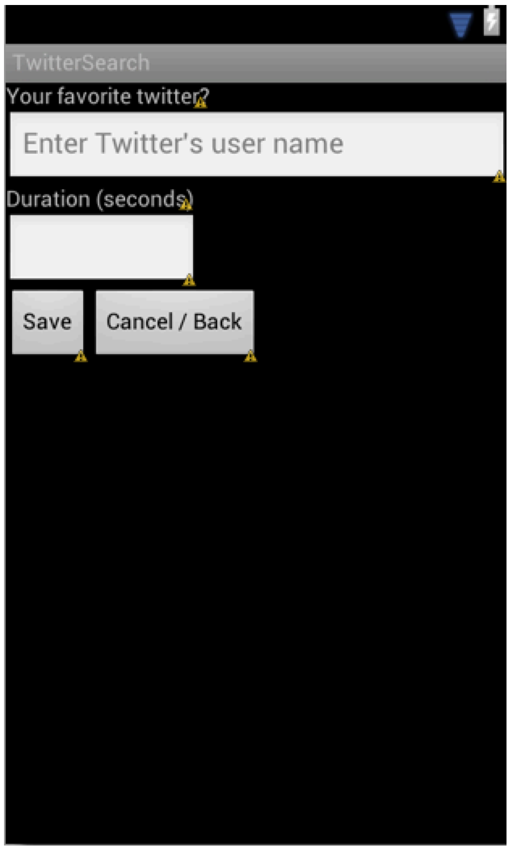
TwitterBookMarkActivity.java

```java
public class TwitterBookMarkActivity extends Activity {

  private static Intent twitterServiceIntent;
  static File favoriteTwitterFile;
  static File durationFile;

  @Override
  public void onCreate(Bundle savedInstanceState) {
    System.out.println("Created TwitterBookMarkActivity");
    super.onCreate(savedInstanceState);
    setContentView(R.layout.tweetbookmark);

    final Button saveButton = (Button) findViewById(R.id.saveButton);   //Save Button Code
    saveButton.setOnClickListener(new View.OnClickListener() {
        public void onClick(View view) {

            //Save the user's favorite Twitter user
            final EditText twitterUserText = (EditText) findViewById(R.id.favoriteTwitterUser);
            //Save the duration
            final EditText twitterCheckDuration = (EditText) findViewById(R.id.serviceDuration);

            String FAVORITETWITTERFILENAME = "favoritetwitter.txt";
            String SERVICEDURATIONFILENAME = "twitterduration.txt";

            String favoriteTwitterUser = twitterUserText.getText().toString();
            String twitterServiceDuration = twitterCheckDuration.getText().toString();
            BufferedReader in = null;
            FileOutputStream fos = null;
            System.out.println("DEBUG: About to write to files");
            try
            {
                fos = openFileOutput(FAVORITETWITTERFILENAME, Context.MODE_PRIVATE);
                fos.write(favoriteTwitterUser.getBytes());
                fos.close();
                System.out.println("DEBUG: Wrote to first file");

                favoriteTwitterFile = getBaseContext().getFileStreamPath("favoritetwitter.txt");
                in = new BufferedReader(new FileReader(favoriteTwitterFile.getAbsolutePath()));
                String favoriteUser = in.readLine();
                System.out.println("DEBUG: Result 1: " + favoriteUser);
```

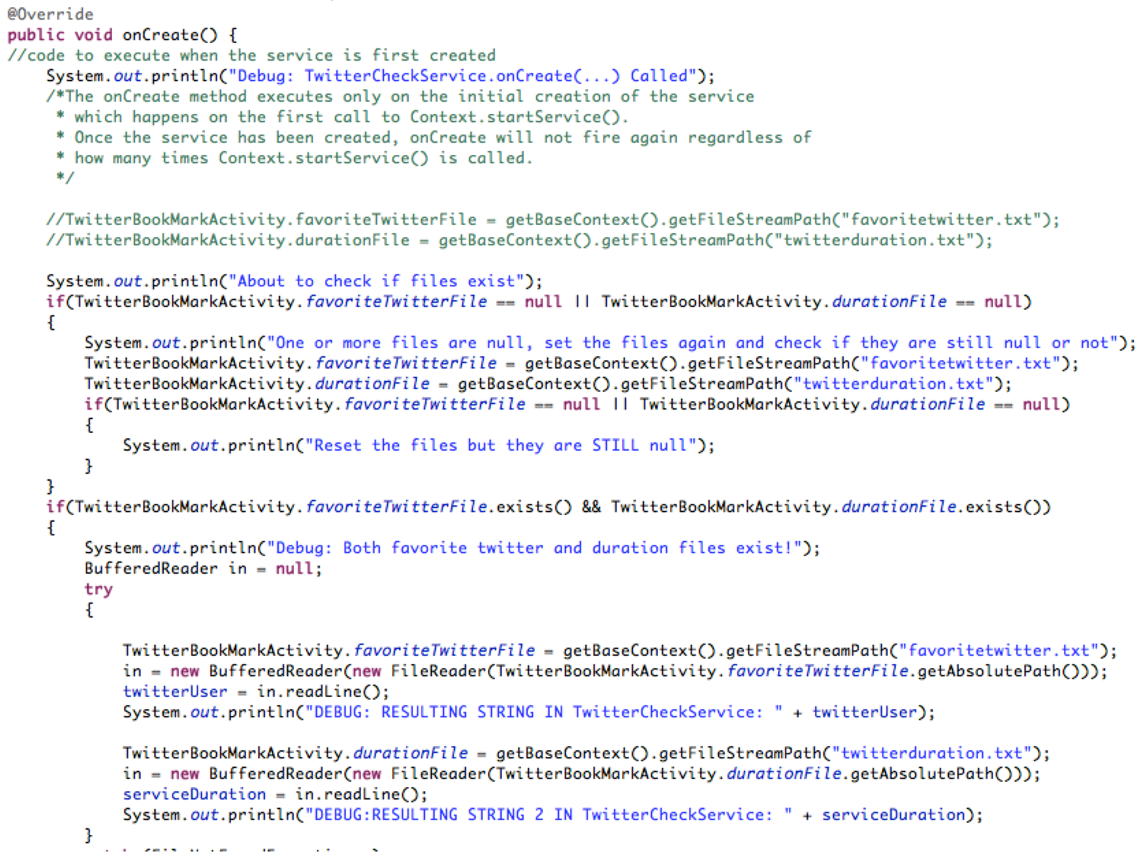
This is the bookmark activity, which calls the service. The major task in this activity is reading the values that the user put into the input and saving them to a .txt file.

```java
   setTwitterServiceIntent(new Intent(TwitterBookMarkActivity.this, TwitterCheckService.class));
   if(getTwitterServiceIntent() != null)
   {
       System.out.println("Debug: Starting service");
       startService(getTwitterServiceIntent()); //Start the service
   }
   else
   {
       System.out.println("Debug: getTwitterServiceIntent() == null, Service not started.");
   }

   }
 });
```

I made the intent for my TwitterService a private variable and used getters and setters. This allowed me to easily set the intent in the TwitterBookMarkActivity file and access it from the TwitterCheckService file.

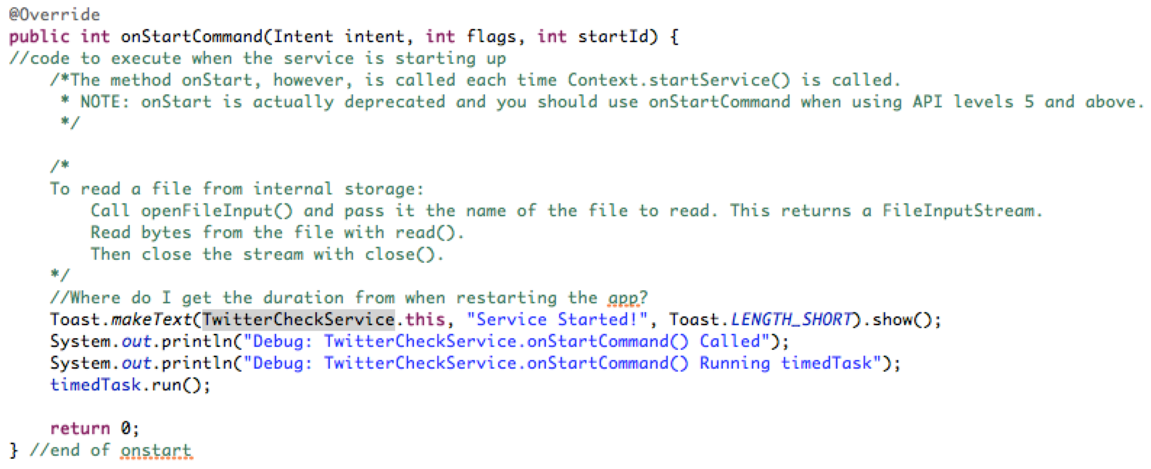

TwitterSearch
Your favorite twitter?
Enter Twitter's user name
Duration (seconds)
Save
Cancel / Back

TwitterCheckService.java



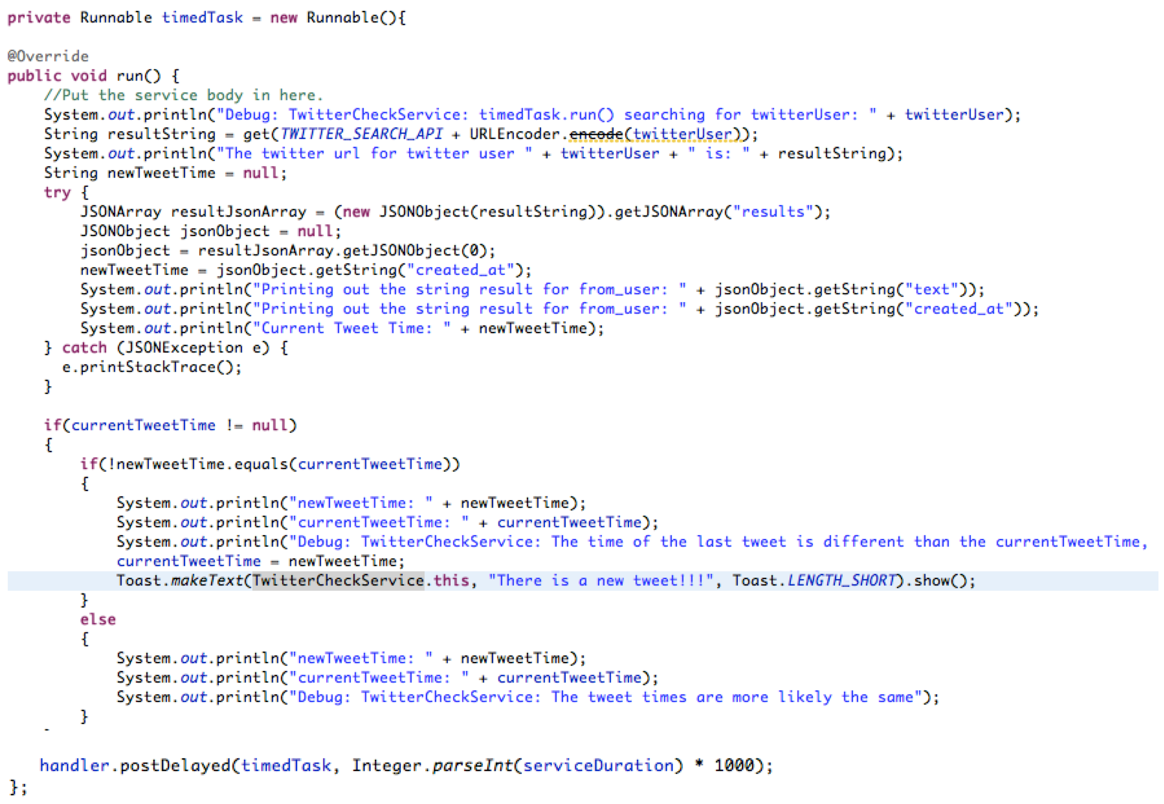
```
@Override
public void onCreate() {
//code to execute when the service is first created
    System.out.println("Debug: TwitterCheckService.onCreate(...) Called");
    /*The onCreate method executes only on the initial creation of the service
     * which happens on the first call to Context.startService().
     * Once the service has been created, onCreate will not fire again regardless of
     * how many times Context.startService() is called.
     */

    //TwitterBookMarkActivity.favoriteTwitterFile = getBaseContext().getFileStreamPath("favoritetwitter.txt");
    //TwitterBookMarkActivity.durationFile = getBaseContext().getFileStreamPath("twitterduration.txt");

    System.out.println("About to check if files exist");
    if(TwitterBookMarkActivity.favoriteTwitterFile == null || TwitterBookMarkActivity.durationFile == null)
    {
        System.out.println("One or more files are null, set the files again and check if they are still null or not");
        TwitterBookMarkActivity.favoriteTwitterFile = getBaseContext().getFileStreamPath("favoritetwitter.txt");
        TwitterBookMarkActivity.durationFile = getBaseContext().getFileStreamPath("twitterduration.txt");
        if(TwitterBookMarkActivity.favoriteTwitterFile == null || TwitterBookMarkActivity.durationFile == null)
        {
            System.out.println("Reset the files but they are STILL null");
        }
    }
    if(TwitterBookMarkActivity.favoriteTwitterFile.exists() && TwitterBookMarkActivity.durationFile.exists())
    {
        System.out.println("Debug: Both favorite twitter and duration files exist!");
        BufferedReader in = null;
        try
        {

            TwitterBookMarkActivity.favoriteTwitterFile = getBaseContext().getFileStreamPath("favoritetwitter.txt");
            in = new BufferedReader(new FileReader(TwitterBookMarkActivity.favoriteTwitterFile.getAbsolutePath()));
            twitterUser = in.readLine();
            System.out.println("DEBUG: RESULTING STRING IN TwitterCheckService: " + twitterUser);

            TwitterBookMarkActivity.durationFile = getBaseContext().getFileStreamPath("twitterduration.txt");
            in = new BufferedReader(new FileReader(TwitterBookMarkActivity.durationFile.getAbsolutePath()));
            serviceDuration = in.readLine();
            System.out.println("DEBUG:RESULTING STRING 2 IN TwitterCheckService: " + serviceDuration);
        }
```

In the onCreate() method I read the information I need from my .txt files about what the username and service interval duration is.



```
@Override
public int onStartCommand(Intent intent, int flags, int startId) {
//code to execute when the service is starting up
    /*The method onStart, however, is called each time Context.startService() is called.
     * NOTE: onStart is actually deprecated and you should use onStartCommand when using API levels 5 and above.
     */

    /*
    To read a file from internal storage:
        Call openFileInput() and pass it the name of the file to read. This returns a FileInputStream.
        Read bytes from the file with read().
        Then close the stream with close().
    */
    //Where do I get the duration from when restarting the app?
    Toast.makeText(TwitterCheckService.this, "Service Started!", Toast.LENGTH_SHORT).show();
    System.out.println("Debug: TwitterCheckService.onStartCommand() Called");
    System.out.println("Debug: TwitterCheckService.onStartCommand() Running timedTask");
    timedTask.run();

    return 0;
} //end of onstart
```

Once I have the information about the username and service interval duration, I call onStartCommand, which runs my timedTask runnable for the first time.



```
private Runnable timedTask = new Runnable(){

@Override
public void run() {
    //Put the service body in here.
    System.out.println("Debug: TwitterCheckService: timedTask.run() searching for twitterUser: " + twitterUser);
    String resultString = get(TWITTER_SEARCH_API + URLEncoder.~~encode~~(twitterUser));
    System.out.println("The twitter url for twitter user " + twitterUser + " is: " + resultString);
    String newTweetTime = null;
    try {
        JSONArray resultJsonArray = (new JSONObject(resultString)).getJSONArray("results");
        JSONObject jsonObject = null;
        jsonObject = resultJsonArray.getJSONObject(0);
        newTweetTime = jsonObject.getString("created_at");
        System.out.println("Printing out the string result for from_user: " + jsonObject.getString("text"));
        System.out.println("Printing out the string result for from_user: " + jsonObject.getString("created_at"));
        System.out.println("Current Tweet Time: " + newTweetTime);
    } catch (JSONException e) {
      e.printStackTrace();
    }

    if(currentTweetTime != null)
    {
        if(!newTweetTime.equals(currentTweetTime))
        {
            System.out.println("newTweetTime: " + newTweetTime);
            System.out.println("currentTweetTime: " + currentTweetTime);
            System.out.println("Debug: TwitterCheckService: The time of the last tweet is different than the currentTweetTime,
            currentTweetTime = newTweetTime;
            Toast.makeText(TwitterCheckService.this, "There is a new tweet!!!", Toast.LENGTH_SHORT).show();
        }
        else
        {
            System.out.println("newTweetTime: " + newTweetTime);
            System.out.println("currentTweetTime: " + currentTweetTime);
            System.out.println("Debug: TwitterCheckService: The tweet times are more likely the same");
        }
    -

   handler.postDelayed(timedTask, Integer.parseInt(serviceDuration) * 1000);
};
```

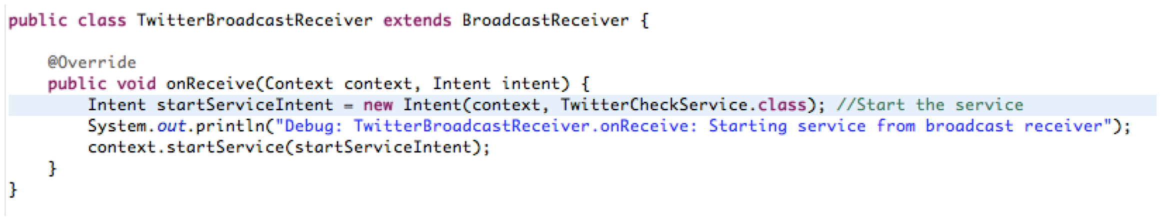
I use a handler and a runnable to keep the service going in intervals.

I used the same code for the Twitter API as in the TwitterSearchActivity. However, instead of getting an entire array of results for the given username, I just got the 0^{th} element. I took the date from the 0^{th} element and stored it in a newTweetTime string. Later I check if the newTweetTime is not equal to the currentTweetTime. If the newTweetTime and the currentTweetTime are not equal, then the newTweetTime must be the time of a new tweet, in which case we set the currentTweetTime to the time of the new tweet and display our toast message.

Once all of the implementation code is done I set the runnable to be run again after the given interval.

I used the Twitter username kakijun for my project and usually tested it in intervals of 10 seconds.

TwitterBroadcastReceiver.java



```
public class TwitterBroadcastReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        Intent startServiceIntent = new Intent(context, TwitterCheckService.class); //Start the service
        System.out.println("Debug: TwitterBroadcastReceiver.onReceive: Starting service from broadcast receiver");
        context.startService(startServiceIntent);
    }
}
```

TwitterBroadcastReceiver is a class that restarts the service in the event that the device has been turned off.