**CMSC 628 Assignment 1**

Michael Davis, Julia Ford

This app is designed to estimate a user’s activity over a 120-second period. Based on the phone’s physical sensor input, recorded over the 120-second interval, the app guesses whether the user is sitting, lying down, or walking (or any upright behavior). While the accelerometer, gyroscope, and location sensors were allowed for this project, we were limited by hardware. Our most accurate accelerometer belongs to a phone running Android version 16, whereas the most recent API is version 22. Additionally, none of our devices had functional gyroscope sensors or would accept the Maps API, so we had to design an algorithm based on accelerometer input alone. While the algorithm is relatively simple and not very accurate, it is attempting to make a guess based on very limited input information.

The bulk of our program is SensorFragment, a custom ListFragment that implements the SensorEventListener class. It contains 3 timed handlers: one for periodically sending a warning message if necessary, one for reading input data at a fixed rate, and one for guessing the user’s activity. When this third handler updates (on the 120-second mark), it stores the info in a custom EntryItem object and also outputting the data to a file. The EntryItem’s constructor inputs the start and end times (as long formatted millisecond times) and a string (“Sitting”, “Lying Down”, or “Walking/Standing”). Then, EntryAdapter (our custom ArrayAdapter) translates this item into a View and then SensorFragment displays it.

When the app starts and loads the SensorFragment, it checks to see if the sensors are functional. If not, it returns a static string from the Warnings class, and will display this as a Toast text when the warning-handling handler updates.

To keep the app’s data continuous between loaded SensorFragment instances (one is generated when the phone is reoriented or the app is minimized), data is saved to static variables. By using timed handlers instead of relying on onSensorChanged’s default behavior, this ensures that the app continues to run even when not displayed or when minimized. This is not the cleanest practice, but it is effective in this case.

When the device’s sensors detect new data and the onSensorChanged function is called, it first checks to see that enough time has passed (a minimum of ~0.1 seconds). If so, it checks the x, y, and z values of the accelerometer input. For most Android phones, the y-axis represents the phone’s vertical axis, the z-axis represents the phone’s axis facing from the screen to the user, and the x-axis represents the phone’s horizontal axis parallel to the user. The algorithm relies on the assumption that during the time of recording, the phone is in the user’s pants pocket. If the user is upright, then gravity will have the greatest effect on the device’s y-axis. If the user is sitting with the phone in their pocket, it is most likely on its side, and gravity will have the greatest effect on the device’s x-axis. If the user is lying down, they may be on their side or holding the phone upwards, in which case gravity will probably have the greatest effect on the device’s z-axis. When onSensorChanged runs its main logic, the maximum of the x, y, and z values are determined, and an appropriate integer variable is incremented. These integers represent the likelihood that a certain action was being performed during the 120-second interval. At the end of the interval, the greatest of these three likelihoods is determined, and we assume that it represents the activity that the user was performing the most during the interval.

The UI is very basic. Each time entry shows the time as “HH:mm – HH:mm” form and then the text of the estimated activity. There can be at most 10 entries at a time; when an eleventh entry is to be added, the first entry is removed, ensuring a cap of 10 items.