James Walker
Einstein Essibu
Leilani Rivera
Alina Zacaria
Zefan Amare

Project 4: Implementing a Stopwatch using the State Machine Design Pattern

From the commencement of this project, there was an understanding that the implementation would not require a heavy amount of code. Through examining our state diagram, we comprehended that the implementation would take place mostly within the individual state classes. With this in mind, we approached the project with an emphasis on the stopwatch states and the state machine.

The first task we undertook was editing the XML file “activity_main” to match the program requirements. We achieved this by deleting the “minutes” and “semicolon” from the user interface and later making the necessary changes to the “StopWatchAdapter” class. One challenge we faced from the onset of the project was comprehending the state machine in terms of its levels of abstraction. IntelliJ’s features allowed us to view the instantiations of particular methods across the project, which was immensely helpful.

By looking at our diagram, we understood that the two “Lap” classes wouldn’t be of much use to us. We refactored these to classes and instead implemented “Increment”, “Decrement”, and “Alarming” states. Following our diagram, in “StoppedState” we implemented the transitions to “DecrementingState” under the condition that the method “getRunTime” returned an integer value greater than 0. In “IncrementState” we implemented the logic that set the limit of incrementations to 99, allowing for a three-second delay, which would trigger the “DecrementingState.” The “DecrementingState” would in turn trigger the “AlarmingStates” if “getRunTime” returned 0 and would reset if the button were to be pushed. “AlarmingState” triggers the android ringtone and is reset if the button is pushed (onStartStop). Though we had some trouble implementing the logic to reset the clock, we were able to meet the project's functional requirements.

We definitely feel the state diagram activity aided us in our implementation. Though we might not have implemented the methods in the chronological order of the state diagram, our implementation follows its logic. We feel that taking part in the diagram activity before implementation facilitated our ability to understand and manipulate the mutually transitory states.

During the design process, we were able to utilize some Agile principles, albeit partly out of necessity due to time constraints. Especially during the closing moments of the project, individual team members collaborated closely, making the completion of the project possible. Our team showed a capacity to adapt and complete tasks on a short schedule. We feel the design process enriched our understanding of the State Machine pattern.