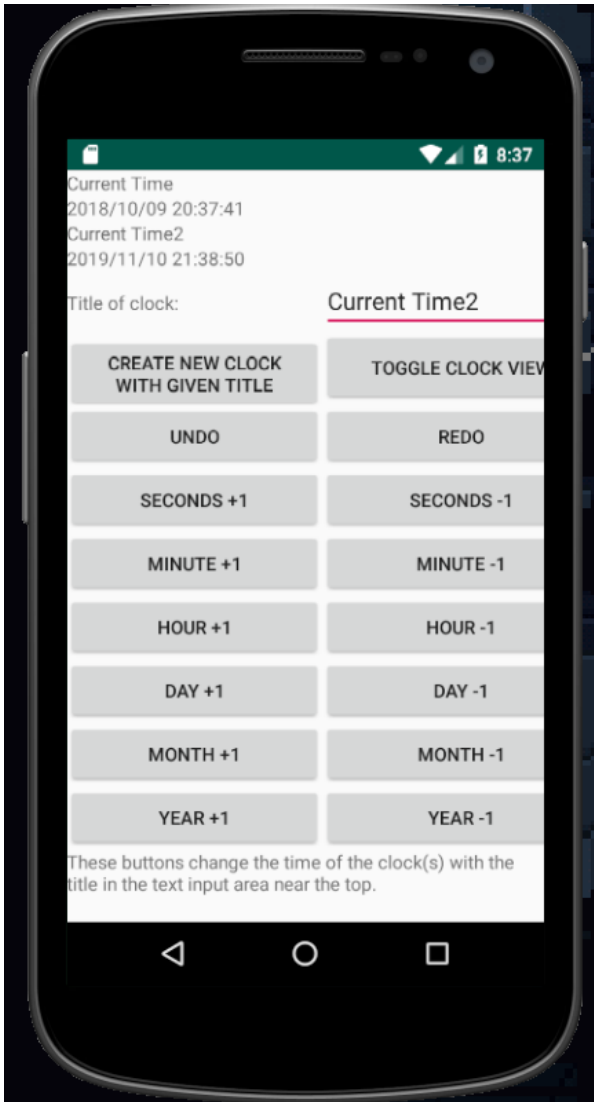
Jackson Bush
10/9/18
CS 4110

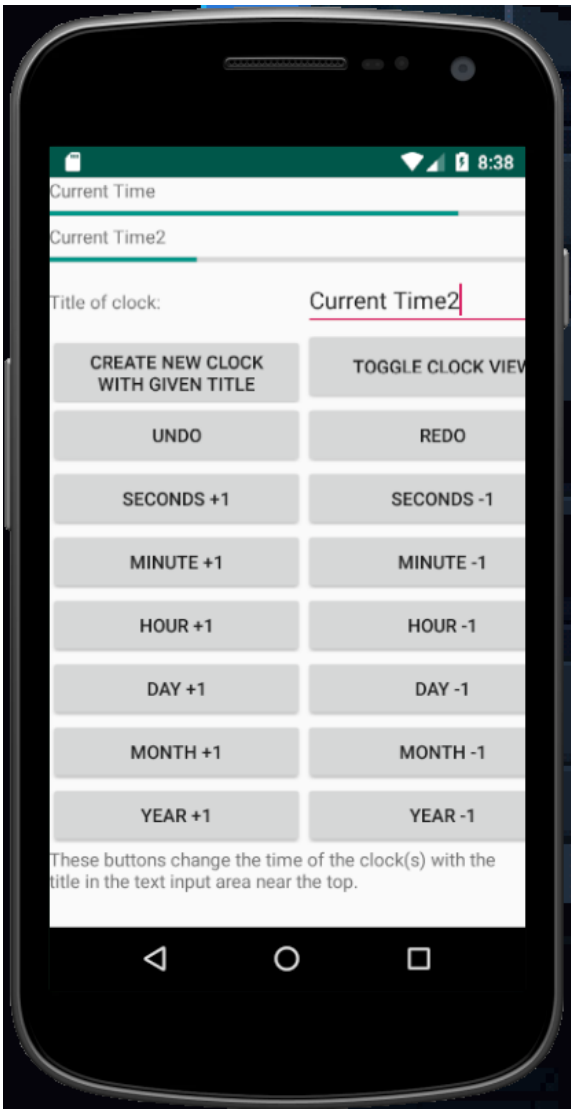
Homework 2: Design Patterns

The main features of my application are:

- "CREATE NEW CLOCK WITH GIVEN TITLE" button - Adds a new clock view to the above clock space and gives it the title specified in the title text input box.
- "TOGGLE CLOCK VIEW" button - Changes all of the digital clock views into loading bar views which represent the percent of the day that has passed. I found this to be much easier than implementing analog clocks because the android analog clock widget cannot be given a time.
- "UNDO" button - Undoes the previous action. I did not realize until I had already written the code that the undo button only needs to undo time changes, so my undo button also undoes the action of creating a new clock as well as the action of toggling the clock view.
- "REDO" button - Redoes the action most recently undone. The redo button works fine for re-toggling the clock view and remaking changes to the clock, but is buggy when redoing the action of creating a new clock. I realized that this is not required for this assignment but decided to keep this functionality.
- Add time and subtract time buttons - these buttons increment or decrement the time of all clocks that have the same title as the title given in the title text input area.

I found that implementing the model view controller design pattern made my program structure quite complicated. I can very clearly see why using MVC on a large project would be a good idea because the flow of control is organized. If you want to know what is getting changed, you look at the view. If you want to know how it is getting changed, you look at the controller. If you want to know what it is getting changed to, you look at the model. This design pattern gets more practically useful as the scale of the program increases, but if I was ad-hocking this homework, I would have put all of the code in mainActivity. It could be that I am just new to MVC, or because I am somehow doing it wrong, but it seemed to add complexity and redundancy that was not necessary. For example, TimeManager.java has a method run() whose purpose is to update the view every second. The problem I found was that this method only works with the context activity of the main activity, so MainActivity needs a function that receives the views that are being changed and the new content to be given to the views. But, because of MVC, the model must pass data through the controller first. Sometimes while writing this code it seemed that this was an unnecessary step, as complexity would have been reduced and less code would have had to be written if the model could talk directly to the views. But again, I think this is likely because my program is relatively simple and flow of control is much easier to follow than a much larger application.


8:37
Current Time
2018/10/09 20:37:41
Current Time2
2019/11/10 21:38:50
Title of clock:
Current Time2
CREATE NEW CLOCK WITH GIVEN TITLE
TOGGLE CLOCK VIEW
UNDO
REDO
SECONDS +1
SECONDS -1
MINUTE +1
MINUTE -1
HOUR +1
HOUR -1
DAY +1
DAY -1
MONTH +1
MONTH -1
YEAR +1
YEAR -1
These buttons change the time of the clock(s) with the title in the text input area near the top.


8:38
Current Time
Current Time2
Title of clock:
Current Time2
CREATE NEW CLOCK WITH GIVEN TITLE
TOGGLE CLOCK VIEW
UNDO
REDO
SECONDS +1
SECONDS -1
MINUTE +1
MINUTE -1
HOUR +1
HOUR -1
DAY +1
DAY -1
MONTH +1
MONTH -1
YEAR +1
YEAR -1
These buttons change the time of the clock(s) with the title in the text input area near the top.