Ariel Ahdoot
Time Incrementer Documentation

Introduction

I created a TimeIncrementer object using Java and implemented it in the file TimeIncrementer.java. Its purpose is to provide a counter that can be used by other developers in their applications. After being initialized, the TimeIncrementer can only be incremented by 1 unit of time at a time, and its core functionality uses a time-based format (60 seconds in a minute, 60 minutes in an hour, 24 hours in a day).

TimeIncrementer Object

The TimeIncrementer object contains an hour field, a minute field, and a second field, and all three are set as private variables so that none of them can be changed directly by a random user. It contains a constructor, a public incrementation function, a public decrementation function, a public display function, and a private update function, whose functionalities will be described below.

TimeIncrementer Constructor

The TimeIncrement constructors takes three arguments: an int for the hour, an int for the minute, and an int for the second. If the hour argument is less than 0 or greater than 24, an IllegalArgumentException is thrown: similarly, if the minute argument or second argument is less than 0 or greater than 59, an IllegalArgumentException is thrown. If the inputs are all legal, then the new TimeIncrementer object's hour field will be set equal to the hour argument, the minute field will be set equal to the minute argument, and the second field will be set equal to the second argument.

Incrementation function

The incrementation function, called Increment, takes a string argument that indicates which unit of time which will be incremented. The caller can input “h” or “hour” if they want to increment the hour, “m”, “min”, or “minute” if they want to increment the minute, and “s”, “sec”, or “second” if they want to increment the second. Whichever unit is chosen is incremented by 1, and then the update function is called before returning.

Decrementation function

The incrementation function, called Decrement, takes a string argument called unit that indicates which unit of time which will be decremented. The caller can input “h” or “hour” if they want to decrement the hour, “m”, “min”, or “minute” if they want to decrement the minute, and “s”, “sec”, or “second” if they want to decrement the second. If the unit chosen is the hour, and the current hour field of the TimeIncrementer object is equal to 0, then the function returns immediately without any decrementation, because the TimeIncrementer does not account for negative time. Otherwise, whichever unit is chosen is decremented by 1, and then the update function is called before returning.

Update function

The update function, called Update, is a private function that is only called once one of the time fields of the TimeIncrementer object is incremented or decremented. Its main purpose is to deal with instances in which the TimeIncrementer's minute field or second field is less than 0 or greater than 59, as well as instances in which the hour field is greater than 24. If the second field is greater than 59, the second field is set to 0 and the minute field is incremented by 1, and if the second field is -1, the second field is set to 59 and the minute field is decremented by 1. Similarly, if the minute field is greater than 59, the minute field is set to 0 and the hour field is incremented by 1, and if the minute field is -1, the minute field is set to 59 and the hour field is decremented by 1. Finally, if the hour field is 25, the hour

field is set to 1. All of these modifications are made to guarantee that the TimeIncrementer object follows a time-based format.

Display Function

The display function, called Display, takes two boolean arguments, standard and military. The first argument, standard, determines whether the current time should be outputted in standard form, and the second argument, military, determines whether the current time should be outputted in military form. Both forms, one of the two forms, or neither of the forms can be outputted, depending on the input provided by the caller.

Source Files, Test Cases, and Compilation

The two source files are TimeIncrementer.java, which contains the TimeIncrementer class and its functions, and Test.java, which contains a function that runs through 20 test cases that validate the entire library's functionality.

The 20 test cases and their explanations, expected inputs, expected outputs, and outputs, are contained in the text file Testcases.txt

Using a terminal, once you are in the timeIncrementer directory, you should be able to compile the source files either individually by inputting

“javac TimeIncrementer.java”

and then

“javac Test.java”

or simultaneously by inputting

“javac *.java”.

Finally, you should be able to run Test.java by inputting

“java Test”.