Software construction - Assignment 4

Deadline: Friday November 22nd 2019 at 6pm.

Since we are in the twenty-first century, imagine having to create a home automation system that can be controlled completely from your smartphone. Your application must be able to control the following devices with their respective features:

- Oven:
  - Switch on
  - Set a timer (in seconds)
  - Set a temperature
  - Set up a program (ventilated, grill, etc)
  - Start cooking (only if the other parameters are set)
  - Check timer (if the oven is running, check the active timer, otherwise the last one set)
  - Interrupt the program (only possible if the oven is in operation)
  - Switch off
- Microwave:
  - Switch on
  - Set timer (in seconds)
  - Set temperature
  - Start Baking (only if the other parameters are set)
  - Check timer (if the microwave is in operation, check the active timer, otherwise the last one set)
  - Interrupt program (only possible if microwave is in operation)
  - Switch off
- Dishwasher:
  - Switch on
  - Start dishwasher:
  - Choose program (glasses, plates, pans, mixed, etc; each program has a different timer)
  - Automatic timer based on the type of program
  - Check timer (if a program is set, it says the time required; if the dishwasher is running, it says the remaining time)
  - Stop dishwasher
  - Switch off
- Washing machine:
  - Switch on
  - Select degrees
  - Select type of washing (Double Rinse, Intense, Quick, Spin)
  - Automatic timer based on the type of program
  - Turn off the washing machine (only if the program is finished)
  - Switch off
- Cleaning robot:
  - Set timer (in seconds)

    - Goes back to charging base when time expires
    - Start the vacuum cleaner (the robot must be in the charging base and every time you leave it, the robot cleaner is 100% charged)
    - Check percentage of cleaning completion
    - Check battery status (if the battery goes to 0 the vacuum cleaner returns to the charging base)
    - Check battery charging status
    - Complete outstanding cleaning
    - End cleaning and go back to charging base (at any time)

The program must **always** be active and be able to manage **all** the devices **simultaneously**.

On the main process, all available devices should be visible. An user can choose a device and open the menu to its (currently active) functions.

Once in the specific device menu, the user can interact with it unless he comes back to the main menu. Unless otherwise specified, the functions of each device are completely independent (example: I can set the type of cooking without having to start the oven, to do it later).

All the functions for a specific device are available if the device has been started (with the **switchOn function**). To put the device in a "power-save" mode you have to use the **switchOff function**. If the power is off, the only available function is switchOn. When you "Interrupt the program" you simply stop the current operation/status of the device, without switching it off. When a program ends, the device **does not reset its state**: in this way you can check if a device has finished or not the program. For this reason you have 2 function: stop/interrupt that reset the current program, swithOff that shouts down the device. SwitchOff completely shuts down the device, while **turn off is meant as a stop command: it resets the current device state**. However, note that the washing machine can not be turned off when a program is not over yet. A program cannot be run if the device is running another program: to load a new program the device must be interrupted or the previous program must be ended.

You must:

- Implement a class "Smartphone" to control all the devices, using the **Command** design pattern.
- Use carefully **interfaces** and **inheritance** when appropriate.
- Create a **class diagram** to show the design of your program.

You can use additional design patterns, but it is not mandatory.

To implement this assignment you might find useful to use java Threads (refer to the "Thread and Timestamp example" folder for the material covered during the lab).

Another option could be to use CurrentTimeMillis() method from Java.lang.System. You can find a quick tutorial here:
https://www.tutorialspoint.com/java/lang/system_currenttimemillis.htm

Note: Please explain any design choice that you took and you would like us to consider during the correction in README.md file in the assignment folder.

Please notice that we will only look at the code on the Master branch in your GitHub repository at the time of the deadline.