#### LOCK CLASS

The Lock class contains the functions and variables for handling the lock interface and its associated hardware controls

|  |  |  |
| --- | --- | --- |
| Properties | | |
| iLock | ILock \* | iLock is a pointer to a ILock type interface which is implemented in the HardwareControl class. The iLock will essentially point to a HardwareControl object which it can use to interact with the hardware IO. |
| lock | Boolean | The lock is a Boolean property that describes the status of the lock switch on the board and needs to be otherwise true in order to run a washing program. |
| Operations | | |
| Lock(): *default constructor of the class* | | |
| Lock(ILock \*) : *constructor of the class which takes an ILock pointer and assigns it to iLock* | | |
| lockMachine() : void  *This is the polling function to check what the lock switch status is and will turn on or turn off the lock.* | | |
| checkLock(): Boolean  *Returns status of lock switch on the board.* | | |
| setLock(Boolean ): void  *Is the setter function for lock property.* | | |
| setInterface(ILock \*) : void  *Assigns the pointer to ILock object(actually HardwareControl object) in its argument to the iLock property.* | | |
| Remarks:  All functions are implemented. The classes are yet to be tested. | | |

#### SOAP CLASS

The Lock class contains the functions and variables for handling the lock interface and its associated hardware controls

|  |  |  |
| --- | --- | --- |
| Properties | | |
| iSoap | ISoap \* | iSoap is a pointer to an ISoap type interface which is implemented in the HardwareControl class. The iSoap will essentially point to a HardwareControl object which it can use to interact with the hardware IO. |
| soapCpt1 | Boolean | The soapCpt1 is a Boolean property that describes the status of the soap compartment 1 switch on the board and needs to be otherwise true in order to run a washing program. |
| soapCpt2 |  | The soapCpt2 is a Boolean property that describes the status of the soap compartment 2 switch on the board and needs to be otherwise true in order to run a washing program. |
| Operations | | |
| Soap(): *default constructor of the class* | | |
| Soap(ISoap \*) : *constructor of the class which takes an ISoap pointer and assigns it to iSoap* | | |
| checkCpt1() : boolean  *This is the polling function that will check if soap switch 1 is turned on and will turn on soap 1 LED accordingly.* | | |
| checkCpt2() : boolean  *This is the polling function that will check if soap switch 2 is turned on and will turn on soap 2 LED accordingly.* | | |
| lockCpt1(Boolean ): void  *Is the setter function for soapCpt1 property, and will turn the soap 1 LED on or off according to the Boolean argument provided.* | | |
| lockCpt2(Boolean ): void  *Is the setter function for soapCpt2 property, and will turn the soap 2 LED on or off according to the Boolean argument provided.* | | |
| setInterface(ISoap \*) : void  *Assigns the pointer to ISoap object(actually HardwareControl object) in its argument to the iSoap property.* | | |
| Remarks:  All functions are implemented. The classes are yet to be tested. | | |

#### PROGRAMEXECUTOR CLASS

The Program Executor class contains the functions and variables for handling the intermediate classes that interact with various hardware control interfaces.

|  |  |  |
| --- | --- | --- |
| Properties | | |
| mProgramSettings | ProgramSettings \* | mProgramSettings is a pointer to an ProgramSettings type object which contains information on the various programs available and their associated costs. |
| mCoinWallet | CoinWallet \* | mCoinWallet is a pointer to an CoinWallet type object which contains functions and variables related to the amount of money the user puts in the laundry machine. |
| mWater | Water | mWater is an object of type Water which contains various variables and functions related to controlling the water-related hardware of the machine. |
| mTemperature | Temperature | mTemperature is an object of type Temparature which contains various variables and functions related to controlling the heater and temperature related hardware of the machine. |
| mSoap | Soap | mSoap is an object of type Soap which contains various variables and functions related to controlling the two soap compartments and their related hardware on the machine. |
| mLock | Lock | mLock is an object of type Lock which contains various variables and functions related to controlling the lock of the machine and its related hardware. |
| mMotor | Motor | mMotor is an object of type Motor which contains various variables and functions related to controlling the motor and its related hardware of the machine. |
| mBuzzer | Buzzer | mBuzzer is an object of type Buzzer which contains various variables and function related to controlling the buzzer and its related hardware of the machine. |
| Operations |  |  |
| Soap(): *default constructor of the class* | | |
| ProgramExecutor(IBuzzer\* b, IMotor\* m, ILock\* l, ISoap\* s, ITemperature\* t, IWater\* w) : *constructor of the class which takes an ISoap, ILock, IMotor, IBuzzer, Itemperature and IWater pointer and assigns it to the respective Lock, Buzzer, Motor, Soap, Temperature and Water objects* | | |
| Start(ProgramSettings \*) : boolean  *This function assigns the program function object in its arguments to the mProgramSettings variable.* | | |
| StepSwitches() : boolean  *This is the polling function that reads the soap and lock switches and turns the corresponding LED’s accordingly.* | | |
| StepCoinWallet() : boolean  *Calls the polling function of its mCoinWallet object to read and interpret the button presses for the coins and turn on or off the corresponding LED’s.* | | |
| IsReady(char) : boolean  *Resets the program for ProgramSettings to the one provided to it in its arguments. It then compares the amount of money in its CoinWallet object against the cost of the program of the ProgramSettings objects and returns a true if the money is sufficient or a false otherwise.* | | |
| setCoinWallet(CoinWallet\* ): void  *Assigns the pointer in its argument to the mCoinWallet property.* | | |
| stopDelay(int Speed): void  Is a delay function to be used when stopping the motor and changing direction. This is because the motor needs some time to stop before it can change direction, and the delay will vary depending on the current speed of the motor. | | |
| Centrifugate(char prog): void  This function runs the centrifugation recipe for each of the programs as provided in its argument. The centrifugation is described in the lab manual for each program and involves spinning the laundry at high speeds for some time. | | |
| Prewash(char prog):void  This function runs the pre-wash recipes for each given program in its argument as provided by the lab manual. | | |
| Mainwash\_Phase1(char prog): void  This function runs the main wash part 1, depending on the program given in its argument as provided in the recipe described in the lab manual. | | |
| Mainwash\_Phase2(char prog): void  This function runs the main wash part 2, depending on the program given in its argument as provided in the recipe described in the lab manual. | | |
| DoFullRotating(int NbrOfTimes, int Speed, int DelayVal):void  This function rotates the motor first in clockwise then in counter-clockwise direction at Speed given in the arguments and for a given amount of time given in its argument DelayVal. It repeats the steps NbrOfTimes as also given by the arguments. | | |
| tempDelay(int d):void  tempDelay replaces the regular delay in that it breaks up a regular delay function into smaller intervals during which it can poll the temperature and turn the heater on or off according to what the desired temperature has been set. The total delay lasts for the amount of time d given in milliseconds. | | |
| Remarks:  All functions are implemented. The classes are yet to be tested. | | |
|  | | |