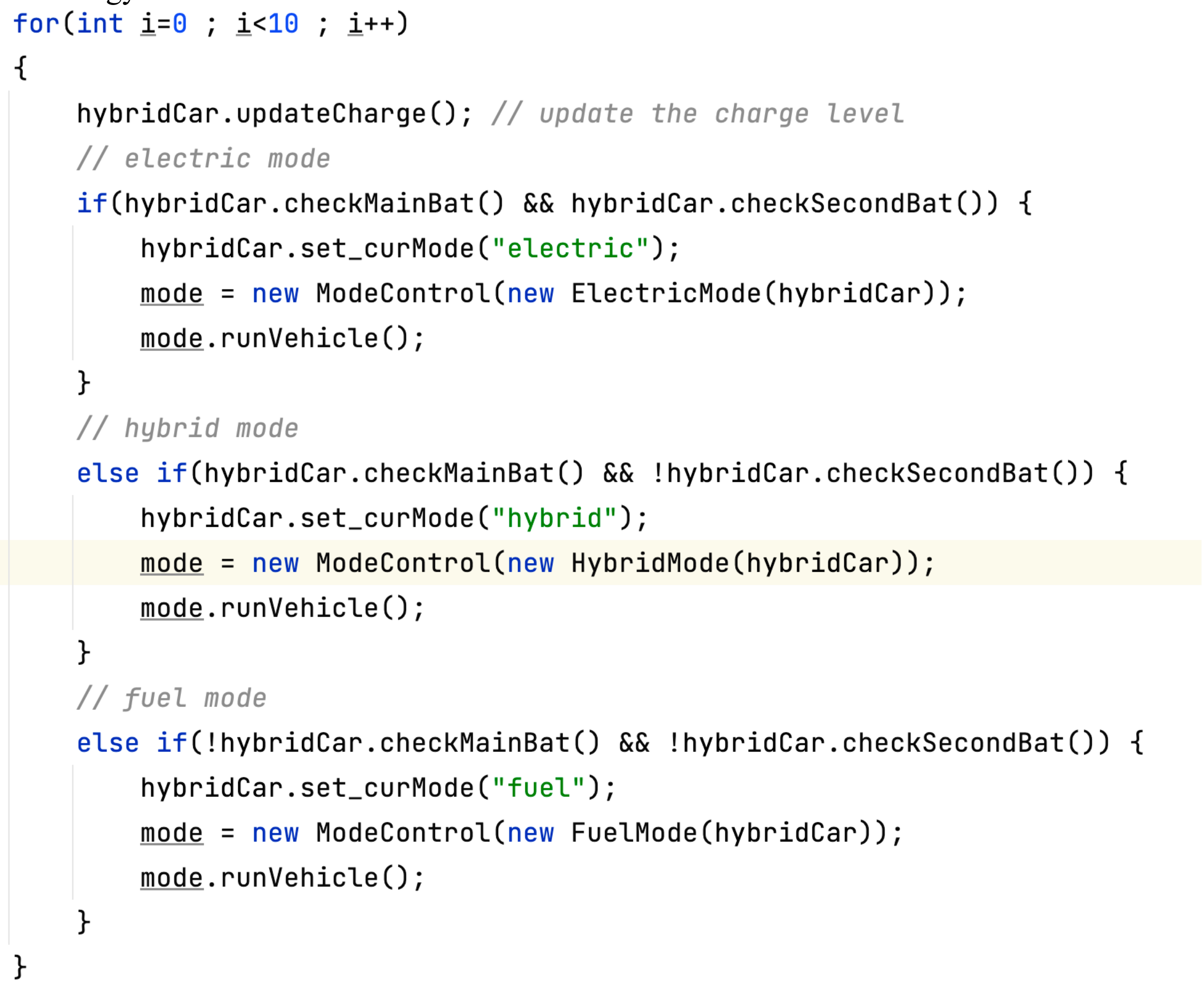
HW3 User's Manual

writer : Juyoung Ha (21700760)

1. Strategy Pattern

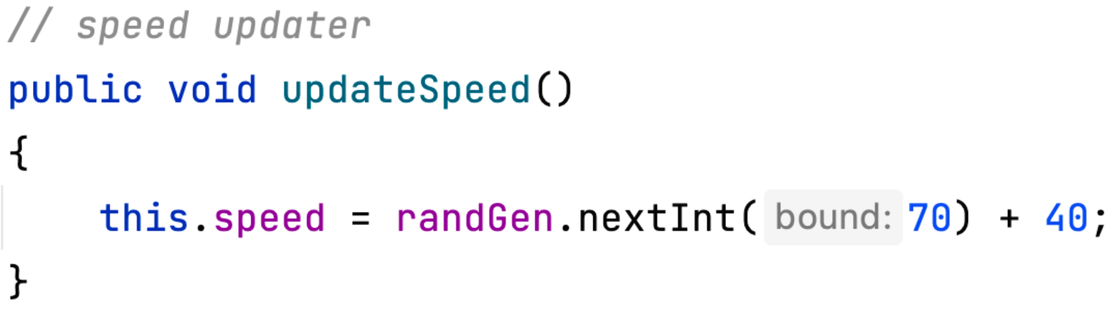


```
for(int i=0 ; i<10 ; i++)
{
    hybridCar.updateCharge(); // update the charge level
    // electric mode
    if(hybridCar.checkMainBat() && hybridCar.checkSecondBat()) {
        hybridCar.set_curMode("electric");
        mode = new ModeControl(new ElectricMode(hybridCar));
        mode.runVehicle();
    }
    // hybrid mode
    else if(hybridCar.checkMainBat() && !hybridCar.checkSecondBat()) {
        hybridCar.set_curMode("hybrid");
        mode = new ModeControl(new HybridMode(hybridCar));
        mode.runVehicle();
    }
    // fuel mode
    else if(!hybridCar.checkMainBat() && !hybridCar.checkSecondBat()) {
        hybridCar.set_curMode("fuel");
        mode = new ModeControl(new FuelMode(hybridCar));
        mode.runVehicle();
    }
}
```

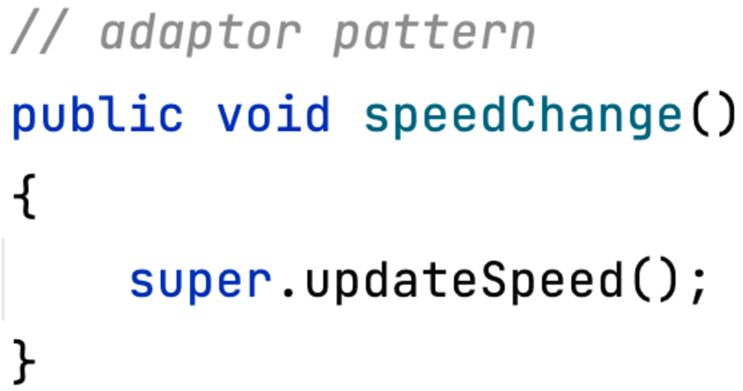
Above source code is the part of adaptorMain.java. As you can see, all car mode run by "runVehicle()" method, however, according to context, the runVehicle() method execute another algorithm. Each algorithm are implemented in FuelMode.java, HybridMode.java, and ElectricMode.java. By using DriveMode.java(interface) and ModeControl.java(class), the method runVehicle() execute another action(algorithm). This is good example of Strategy Pattern.

2. Adaptor Pattern

"AbstractHybridCar.java" have the method "updateSpeed" and the "HybridCar.java" have "speedChange.java" which implemented by updateSpeed method (in this case the class "AbstractHybridCar" is the parent class of "HybridCar" class).

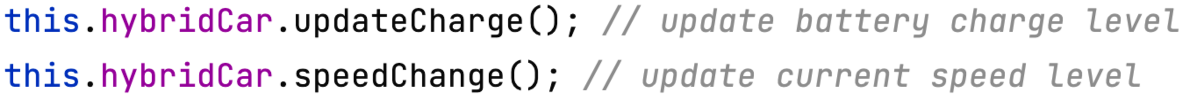


```
// speed updater
public void updateSpeed()
{
    this.speed = randGen.nextInt( bound: 70) + 40;
}
```



```
// adaptor pattern
public void speedChange()
{
    super.updateSpeed();
}
```

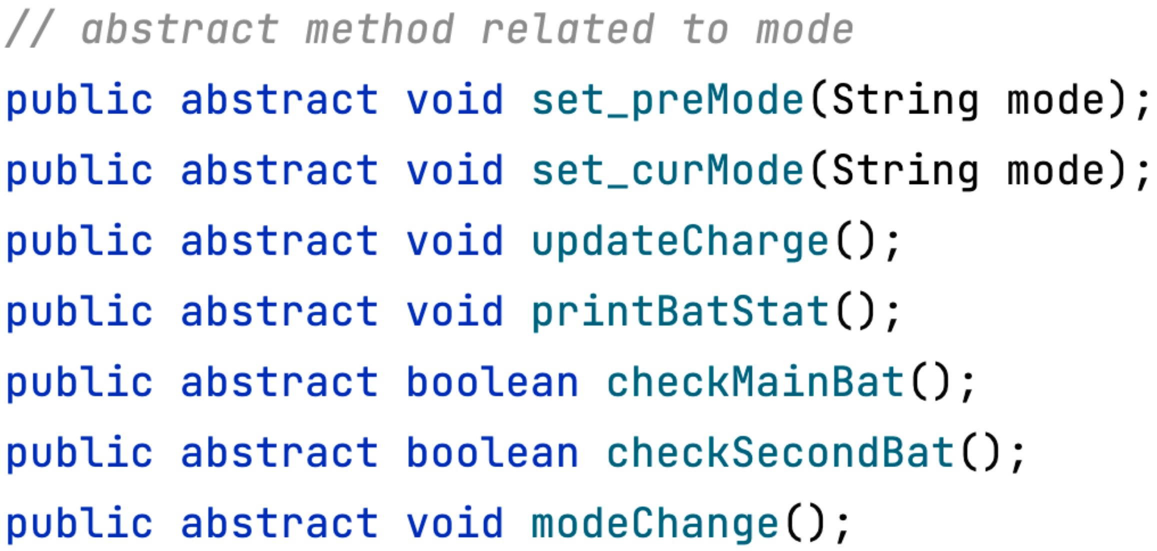
And the speedChange method are using user function, and the updateSpeed method are using to implement speedChange method.



```
this.hybridCar.updateCharge(); // update battery charge level
this.hybridCar.speedChange(); // update current speed level
```

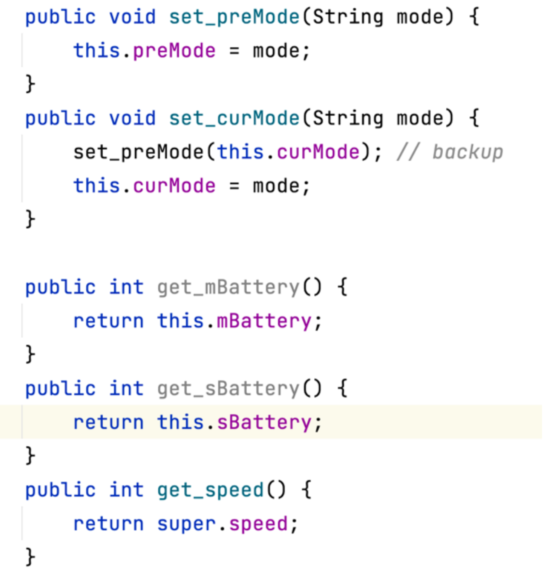
This is a example of adaptor pattern.

3. Template Method Pattern



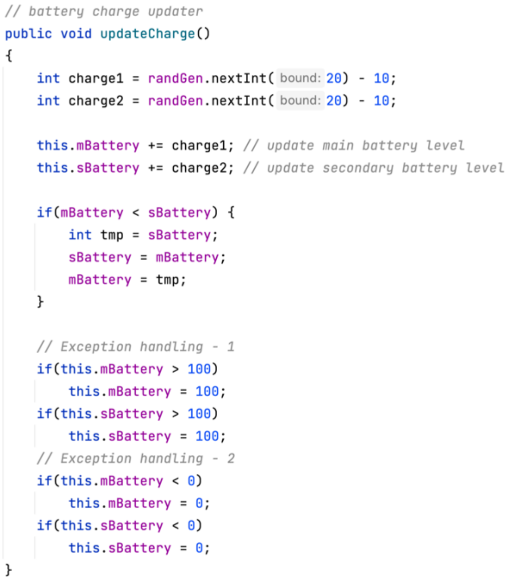
```java
// abstract method related to mode
public abstract void set_preMode(String mode);
public abstract void set_curMode(String mode);
public abstract void updateCharge();
public abstract void printBatStat();
public abstract boolean checkMainBat();
public abstract boolean checkSecondBat();
public abstract void modeChange();
```

"AbstractHybridCar.java" have abstract method and these abstract methods are implemented in HybridCar.java like below (HybridCar is child class of AbstractHybridCar). This is the example of "Template Method Pattern".



```java
public void set_preMode(String mode) {
    this.preMode = mode;
}
public void set_curMode(String mode) {
    set_preMode(this.curMode); // backup
    this.curMode = mode;
}

public int get_mBattery() {
    return this.mBattery;
}
public int get_sBattery() {
    return this.sBattery;
}
public int get_speed() {
    return super.speed;
}
```



```java
// battery charge updater
public void updateCharge()
{
    int charge1 = randGen.nextInt( bound: 20) - 10;
    int charge2 = randGen.nextInt( bound: 20) - 10;

    this.mBattery += charge1; // update main battery level
    this.sBattery += charge2; // update secondary battery level

    if(mBattery < sBattery) {
        int tmp = sBattery;
        sBattery = mBattery;
        mBattery = tmp;
    }

    // Exception handling - 1
    if(this.mBattery > 100)
        this.mBattery = 100;
    if(this.sBattery > 100)
        this.sBattery = 100;
    // Exception handling - 2
    if(this.mBattery < 0)
        this.mBattery = 0;
    if(this.sBattery < 0)
        this.sBattery = 0;
}
```