Spring Quarter 2021
Home
Announcements
Syllabus
Zoom
Panopto Recordings
Assignments
Modules
Grades
Strategy Pattern
Due Apr 12 by 11:59pm Points 25 Submitting a file upload File Types zip
Read Chapter 1

Attachments:

01StrategyUMLSorter-1.pdf
Minimize File Preview

01StrategyPattensOutputGuitarHeroStudents.pdf
Minimize File Preview

Assignment 1: Strategy Pattern

Part 1. The upcoming Guitar Hero VIII: Legends of Rock needs a player configuration system.

Guitar Hero VIII lets you play three different characters (Slash, Jimi Hendrix, and Angus Young).
Each character can use one of three different guitars (Gibson SG, Fender Telecaster or Gibson Flying V) as well as perform a unique solo act (Put the Guitar on Fire, Jump off the Stage, Smash the Guitar).
Implement a player configuration system in Java using the Strategy design pattern.
You should have separate classes to represent each of the components specified.
Part 2. Include a UML class diagram (as a .pdf) that represents your class relationships as part of your submission.

You are free to use any editing tools you wish. A reasonably good UML drawing program is Draw.IO (There will be a demo of this in class later this week). TinyUML is also pretty good.
Eclipse also has a built in UML generator (hint this would be the easiest one).
You are welcome to use whatever you like *as long as it supports the basic UML symbols required for class diagrams -- we will go over these in class*.

While we will discuss them in class, an excellent tutorial on the basics of class diagrams lives here: http://www.agilemodeling.com/artifacts/classDiagram.htm (Links to an external site.)
Here's a class containing the Java main method to help you on your way. Note that the contents of this class are not exhaustive with respect to your assignment. More specifically, add items to the main method to show dynamic swapping of behaviors (which is a critical part of the Strategy pattern).

```
public class GuitarHero {
    public static void main(String[] args) {
        GameCharacter player1 = new GameCharacterSlash(); //note that constructor could be designed to accept initial behaviors

        // e.g. GameCharacter player1 = new GameCharacterSlash(new GibsonFlyingV(), new SetGuitarOnFire());
        GameCharacter player2 = new GameCharacterHendrix();
        player1.playGuitar();  //should print a message saying Slash is playing a Gibson Flying V
        player2.playGuitar(); //should print a message saying Jimi Hendrix is playing whatever you assigned in constructor
        player1.playSolo(); //should print a message saying Slash just set guitar on fire
        player2.playSolo(); //etc.

        //add code below to show the swapping of behaviors

        //e.g. player1.setSolo(new JumpOffStage());

        //then: player1.playSolo(); -- this should print a message that Slash jumped off stage
    }
}
```

Hint1: You should use the Java Duck example from Chapter 1 of HFDP.
Hint2: It helps if you draw a class diagram *before* you start coding.

Your submission must include .java files and the .pdf that contains the UML representation of your classes. In addition you must capture the output from running your program.

Label your .zip folder/file (course_assignment_firstname_lastname): cscd212as01Strategy_firstname_lastname.zip

More specifically turn in the following:

Source files: for Java,
More specifically you should be able to run java GuitarHero to compile and run your code from the directory created by unzipping the submission. This speeds up the process for the grader.

The file with main in it should be named GuitarHero.java.
Make sure you can compile and run from the top directory of the unzipped submission.
UML Class diagram (.pdf format required, name the file Strategy_UML)
Output capture from running your program. This can be a plain text file or a screen shot. Be sure and name the file Strategy_Output.