Lesson 5.04 — Encapsulation

Overview

Objectives — *Students will be able to...*

- **Explain** what encapsulation/abstraction are, and why they are important programming strategies.

Assessments — *Students will...*

- **Teach** a mini-lesson on encapsulation/abstraction, using private fields, using class invariants, and changing internal implementations

Homework — *Students will...*

- **Complete** a quiz at the end of Day 2
- **Complete** chapter 8 self-check questions 17-21

Materials & Prep

- **Group copies** of WS 5.4
- **Assignments for 4 student groups**
- **4 classroom copies** of the textbook (or have students bring books to class)
- **Copies of the grading rubric** on the overhead or printed out (optional)

You will need to circle student assignments on point 2 of WS 5.4, so each group knows what topic they are expected to teach.

Pacing Guide: Day 1

Section	Total Time
Bell-work and attendance	5min
Introduction; reviewing the assignment	10min
Student preparation of lesson	30min
Student practice	10min

Pacing Guide: Day 2

Section	Total Time
Bell-work and attendance	5min
Student mini-lessons	35min
Quiz using student-generated questions	15min

Procedure

Your hook for today's lesson is to turn the reins over to students immediately. Have instructions printed out and sitting at teamwork stations (or on student desks). Encourage students to answer their own questions using the instruction sheet and textbook. Frequently asked questions and suggestions for student groups are included in Accommodations and Differentiation.

Bell-work and Attendance [5 minutes]

Introduction & Reviewing the Assignment [10 minutes]

1. Prepare students by telling them that the 4 topics covered today are the 4 subsections of 8.4, the reading that was due today.

2. Give students 15 minutes to prepare their presentation and generate quiz questions for tomorrow's class.
    a. Use a timer and announce time at 10- and 5- minutes remaining so students can pace themselves.
    b. Answers to commonly asked questions, and tips for the different groups may be found below:

ASSIGNMENT FAQ

Encapsulation & Abstraction Guidance for student teachers

1. If students are lacking direction, encourage students to refer to the textbook. A good example of why it's important to encapsulate code can be found in the "Did you know" blue box in section 8.4.
2. A good "Tricky Code Cheat Sheet" tip would be for students to form a rule their peers can memorize for when its' appropriate to encapsulate code.

Common Questions/Answers

1. **Why can't we just ignore code, or promise not to modify it instead of encapsulating it?**

   Most large programs are authored by many people over the course of months or years! Sometimes you don't need (or want) to know the details of another programming segment—if it ain't broke, don't fix it. To keep from accidentally changing code, it's best practice to "protect" it by encapsulation.

Private Fields Guidance for student teachers

1. If students are having trouble outlining their lesson, suggest that they start with the fourth complete version of code in the textbook. If they use the textbook example, make sure they point out which program is the client code.
2. A good "Tricky Code Cheat Sheet" tip would be the convention of fields at the top of the class, followed by constructors, followed by methods.

Common Questions/Answers

1. **What is the scope of private fields?**

   Private fields are visible to all of the code inside the Point class, including the instances of the class. Client code cannot directly refer to an object's fields if you've encapsulated them (marked them as private).

2. **But what if we still want client code to have to read some of the fields?**

   Write an accessor method (a "get" method). It returns a copy of the field's values to the client, so the client can see the values, but can't modify them.

Class Invariants Guidance for student teachers

1. Remind students to include the definition of a class invariant for students to record in their notebooks.
2. Since we did not cover the this keyword, you should point out to the group that the this keyword in the example allows the code to directly refer to the implicit parameter. Since this might be confusing, you should give this group extra assistance, or assign your most advanced group this lesson.
3. Using division/modulus 60 to convert seconds to minutes or minutes to hours as outlined in the book is a useful tip for the "Tricky Code Cheat Sheet."

Common Questions/Answers

1. **How is enforcing a class invariant different from using a class constant?**

   A class constant will always stay the same value; it does not ever change. A class invariant is a fact about some data that you as the programmer assert will always be true. For instance, in Pokemon, each stat must be less than 255—that is the max value you can have for any one stat. The value itself may go up or down depending on what happens to your Pokemon in the game, but it will always be less than 255.

2. **Can you use a mutator method to change the class invariants?**

   No, so pick and choose your invariants with caution!

3. Skip the exceptions example—exceptions are not on the AP exam and will not be tested!

Changing Internal Implementations Guidance for student teachers

1. If students need guidance on structuring their lesson, encourage them to coordinate with the group teaching class invariants, since they will be modifying the code taught by that group. They might also want to refer to the "Did You Know" blue box in the section.
2. A good tip for the Tricky Code Cheat Sheet could be the simpler time span code.

Common Questions/Answers

1. **How come changing the internal design does affect client code?**

   Since you encapsulated your class, the client code won't know that you changed the internal state. The only way your changing the class code can mess with a client program is if you change the constructors or method headers.

Student Preparation of Lesson [30 minutes]

1. Give each group 7 minutes to present their topic and 2 minutes for questions.
2. Encourage students to ask questions, and be sure to ask a question or two of each team (depending on how many teams you have).
3. Use the grading rubric as outlined here:

3 pts.	2 pts.	1 pts.	0 pts.
Presentation includes definitions and an example with proper syntax.	Presentation includes definitions or an example with proper syntax.	Presentation includes definitions or an example with proper syntax with few mistakes.	Presentation includes definitions or an example with proper syntax with many mistakes.
Presentation includes a non-example as helpful contrast.	Presentation includes a non-example that is marginally helpful.	Presentation includes a non-example that does not add to comprehension.	Presentation includes a non-example that adds confusion, or presentation does not include a non-example.
Presentation includes a helpful tip that is clearly explained and concisely stated.	Presentation includes a helpful tip that is clearly explained or concisely stated.	Presentation includes a helpful tip that is not clearly explained and may include a small error.	Presentation does not include a helpful tip or hint.

Accommodation and Differentiation

Circle around the room to help students through reading the text in the textbook. Make sure that each of your working teams is properly stratified (rather than using tiered grouping).

If students are speeding along, encourage students to write down questions to pose to other groups during mini-lessons. If a group finishes early, encourage them to rehearse lesson delivery.

Teacher Prior CS Knowledge

Java is inconsistent with its use of encapsulation which could cause confusion with beginning students. When getting the length of a String a, the access method `.length()` is used. However, when getting the length of an

array, the public, though final, instance variable `.length` is used.

Misconceptions

Students get the notion that because of data encapsulation, classes cannot be fields of another class. Even though the data in the class is protected from being modified directly by other classes, the class itself can be used as fields in other classes.

Videos

- BJP 8-4, *Encapsulation* http://media.pearsoncmg.com/aw/aw_reges_bjp_2/videoPlayer.php?id=c8-4
- CSE 142, *Encapsulation* (35:38–45:58) https://www.youtube.com/watch?v=V3Gs1Ug82_E&start=2138
- CSE 142, *"this" notation* (optional) (45:59–49:46) https://www.youtube.com/watch?v=V3Gs1Ug82_E&start=2759

Forum discussion

Lesson 5.04 Encapsulation (TEALS Discourse account required)