

CSSE 220

Software Engineering Techniques
Encapsulation
Coupling and Cohesion
Scoping

Please check out EncapsulationExamples from your SVN

The plan

- Software Engineering Techniques:
 - Pair programming
 - Version Control
- Learn 3 essential object oriented design terms:
 - Encapsulation (today's topic)
 - Coupling
 - Cohesion

What Is Pair Programming?

- Two programmers work side-by-side at a computer, continuously collaborating on the same design, algorithm, code, and/or test
- Enable the pair to produce higher quality code than that produced by the sum of their individual efforts

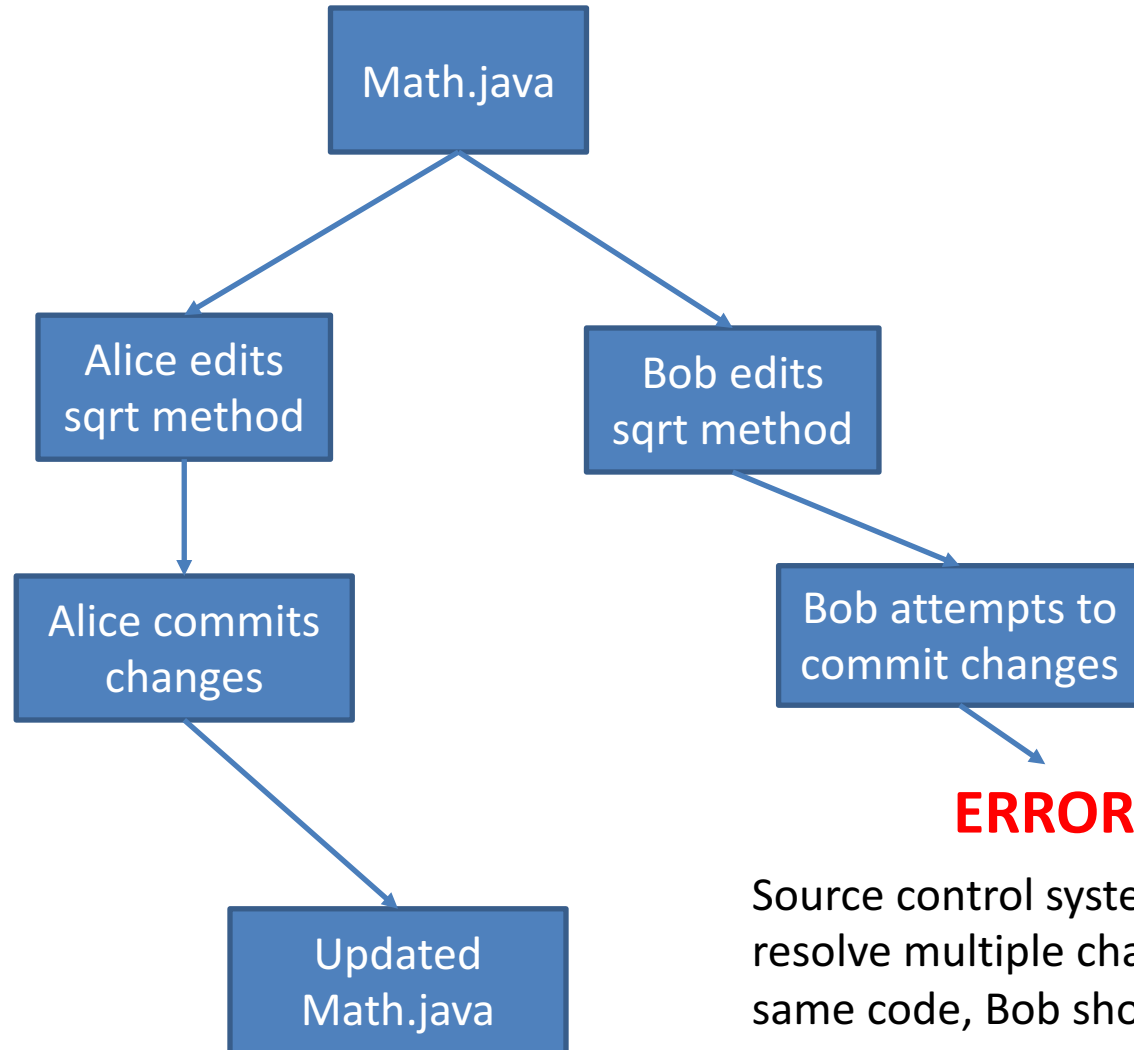


Pair Programming

- Working in pairs on a single computer
 - The *driver*, uses the keyboard, talks/thinks out-loud
 - The *navigator*, watches, thinks, comments, and takes notes
 - Person who really understands should start by navigating 😊
- For hard (or new) problems, this technique
 - Reduces number of errors
 - Saves time in the long run

SOFTWARE VERSIONS

When Two+ People Edit the Same Code

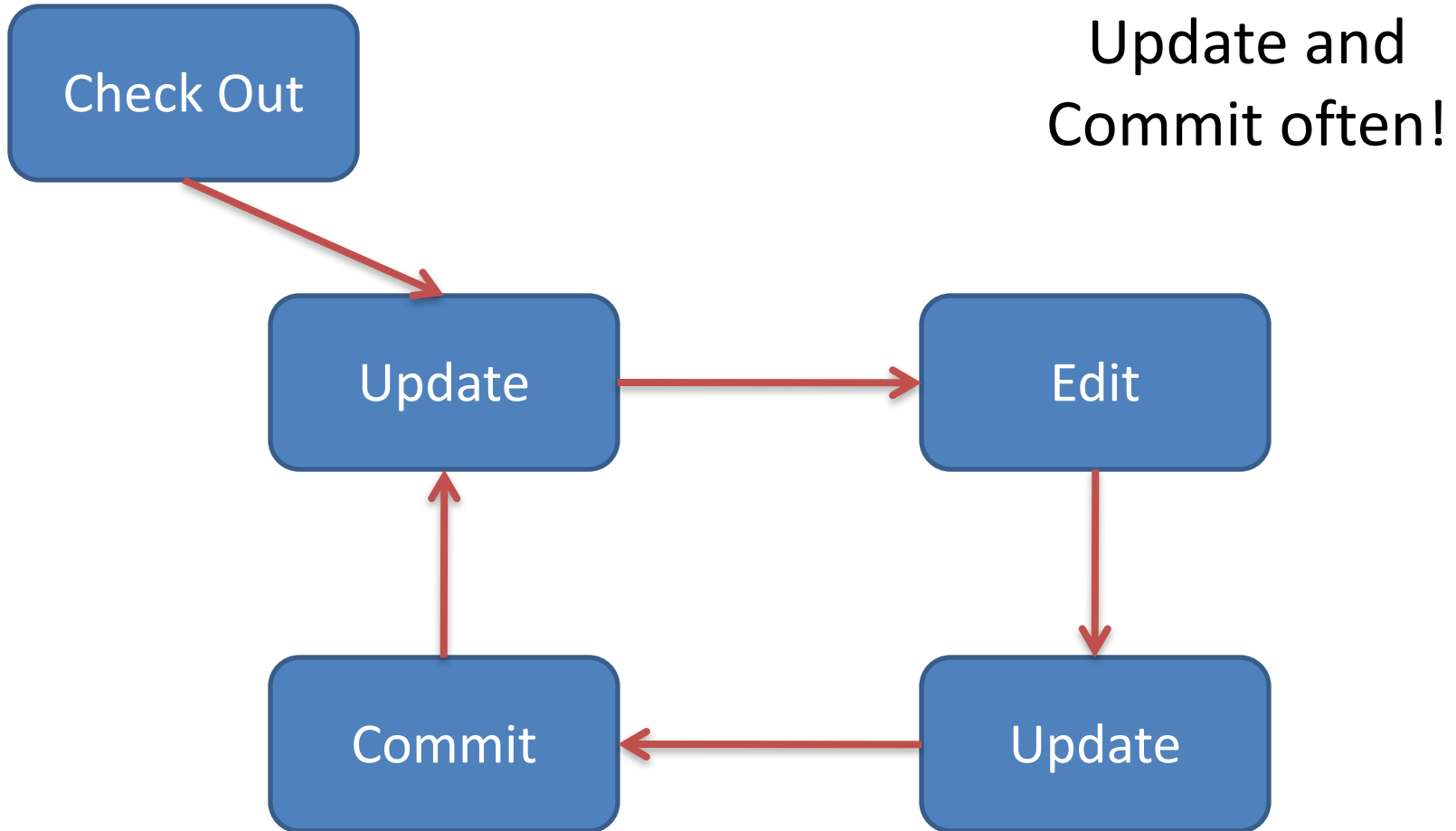


Source control system cannot resolve multiple changes on the same code, Bob should have updated and resolved conflicts before committing.

Team Version Control

- **Version control tracks multiple versions**
 - Enables old versions to be recovered
 - Allows multiple versions to exist simultaneously
- **Always:**
 - **Update before** working
 - **Update again** before committing
 - **Commit often** and with good messages
- **Communicate** with teammates so you don't edit the same code simultaneously
 - Pair programming ameliorates this issue 😊

Team Version Control



What if I get a conflict on update?

- If you did an update and now have File.java, File.java.mine, File.java.rN, and File.java.rM (where N and M are integers):
 - YOU HAVE A CONFLICT!
- Eclipse provides tools for resolving conflicts
- Follow the steps in this link to resolve a conflict:
 - <http://www.rose-hulman.edu/class/csse/csse221/current/Resources/ResolvingSubversionConflicts.htm>

Moving on....

- Learn 3 essential object oriented design terms:
 - **Encapsulation (today's topic)**
 - Coupling
 - Cohesion

What if there were no String class?

- Instead, what if we just passed around arrays of characters - `char[]`
- And every String function that exists now, would instead be a function that operated on arrays of characters
- E.g. `char[] substring(char[] input, int start, int end)`
- Would things be any different? Discuss this with the person next to you.

The Point of All Program Design

- Say someone has written a program that works and it has no bugs, but it is *poorly designed*. What does that mean? Why do we care?
- I think there are two things

Encapsulation

- Mike's definition "grouping some data and the operations that use that data into one thing (an object) and preventing that data from being changed except by using those operations"

Encapsulation

- Makes your program easier to understand by
 - Grouping related stuff together

Encapsulation

- Makes your program easier to understand by...
 - Saving you from having to think about how complicated things might be

An iceberg floating in the ocean. The small tip above the water represents the visible part of a program, while the much larger mass below the surface represents the hidden, complex implementation details.

Using put and get in HashMap

Implementing HashMap

Encapsulation

Makes your program easier to change by...

- Allowing you to change how your data is represented

City Temperature Activity

- I will split you into two groups
 - One group will solve the problem by creating a new class (see the **Class Section example** if you are unsure how to do that)
 - The other group will just write the code in main (see the **Letters Example** if you are unsure how to do that)
- If you finish early, try to solve it the other way too

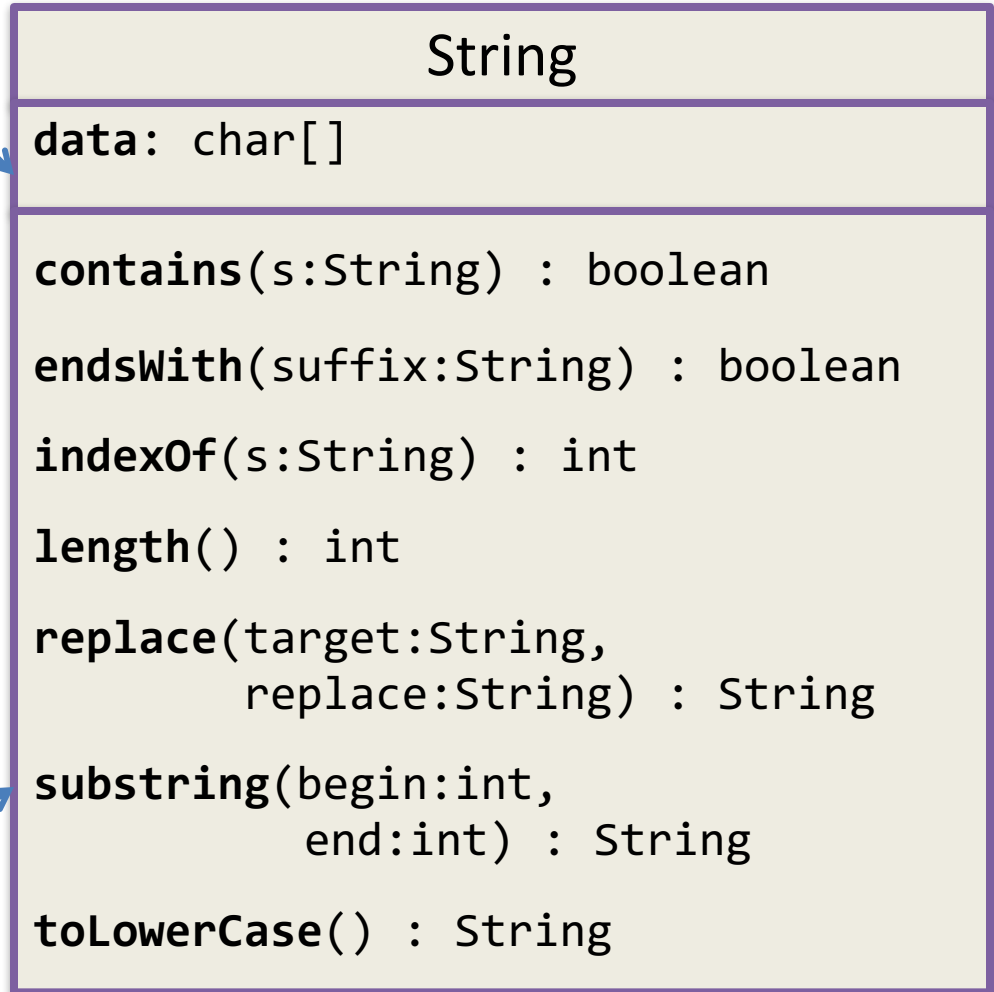
Encapsulation – a good thing?

- Note that we have the ability to change the representation of the CityTemperature class – but how important is that?
- Consider adding a bunch more statistics for each city (max, min, mode)
- Consider adding statistics overall (e.g. overall average)

Adding Types to The Diagram

- Shows the:
 - **Attributes**
(data, called **fields** in Java) and
 - **Operations**
(functions, called **methods** in Java)of the objects of a class
- Does *not* show the implementation
- Is *not* necessarily complete

Fields



Methods

TwoVsTwo

- Look at the code to understand the problem
- Try to solve it using classes and encapsulation
 - Decide what classes/methods you would use (I used two new classes and TwoVsTwo main)
- Draw UML for the classes/methods

My TwoVsTwo Solution

- Let's go through the code!

Crazy Eights

- Instructions are online
- This is to be done with a partner
 - These are assigned by the instructor – the assignment contains a link to the pairs and the repository to use
- If you have questions about the requirements, ask early!

Checkout CrazyEights Project

- Go to SVN repository view at bottom of workbench
 - Window → show view → Other → SVN → SVN Repositories
- Right click in SVN View, then choose New SVN Repository Location
 - [http://svn.csse.rose-hulman.edu/repos/csse220-201630-your_team_repository](http://svn.csse.rose-hulman.edu/repos/csse220-201630-<u>your_team_repository</u>)
 - Your team repository will be csse220-201630-crazyeightsXX where XX is the team number
 - On Moodle, click on “Crazy Eights Team Assignments” to see to what team you have been assigned

Work Time

- Work with your partner on the CrazyEights project
 - Get help as needed
 - *Follow the practices of pair programming!*
- *Don't do any of the work without your partner!*