COMP1409: Introduction to Software Development I

Mike Mulder (mmulder10@bcit.ca)

Week 4

Agenda

- Quiz
  - Quiz 3
  - Review Answers
- Logistics, Lab 3 Solution Review and Assignment 1
- Review
- Lesson 4
  - Variables
  - Constants
  - Multiple Constructors
- Lab 4

Quiz 3

Closed book, laptop, phone, etc.

You have a maximum of 15 minutes to complete

Raise your hand when you are done, and I will retrieve your paper

We will review the answers afterwards

Course Topics – Week 4 Today

Week	Date	Topics	Comments
1	Sept. 8	Bluej Definitions: Classes, objects, data types, state, methods Naming conventions A simple java class	
2	Sept. 15	Comments Visibility modifiers Methods: return types and parameters if statements null Constructors Parameters Assignment statement Logical operators Relational operators	Quiz 1
3	Sept. 22	Accessors Mutators Console output String concatenation public static void main()	Quiz 2
4	Sept. 29	Logical operators Local variables Static variables and methods Constants	Quiz 3

Course Topics

Week	Date	Topics	Comments
5	Oct. 6	Thanksgiving – No Class	
6	Oct. 13	Arithmetic operators Overloading Switch/case	Quiz 4
7	Oct. 20	Composition (object interaction and external method calls)	Quiz 5, Assignment 1 Due
8	Oct. 27	References; identity versus equality null (again) this Debugging techniques	Quiz 6
NOTE: Course Withdrawal Deadline Please inform your instructor that you are dropping this course. You must also fill out and submit the **'REQUEST TO WITHDRAW FROM A PART-TIME STUDIES COURSE'** before session 8 or else you will receive a failing grade on your academic record**.**			
9	Nov. 3	Arrays while loops	Quiz 7
10	Nov. 10	Remembrance Day – No Class	
11	Nov. 17	More arrays for loops	Quiz 8, Assignment 2 Due
12	Nov. 24	ArrayList class Enhanced for (foreach) loop	Quiz 9

Course Topics

Week	Date	Topics	Comments
13	Dec. 1	Iterators	Quiz 10
Course Evaluation: To be conducted online during Session 11 prior to the class break.			
14	Dec. 8	Final Exam	Assignment 3 Due

Logistics

Lab 4B due on Friday of next week (Oct. 5)

No class on Oct. 6 (Thanksgiving)

Assignment 1 Due Oct. 19

Lab 3B

Solutions were generally very well done for a first complete Java class!

Upcoming labs will continue to build upon Java classes and new Java syntax.

Make sure you ask questions if you need clarification or get stuck.

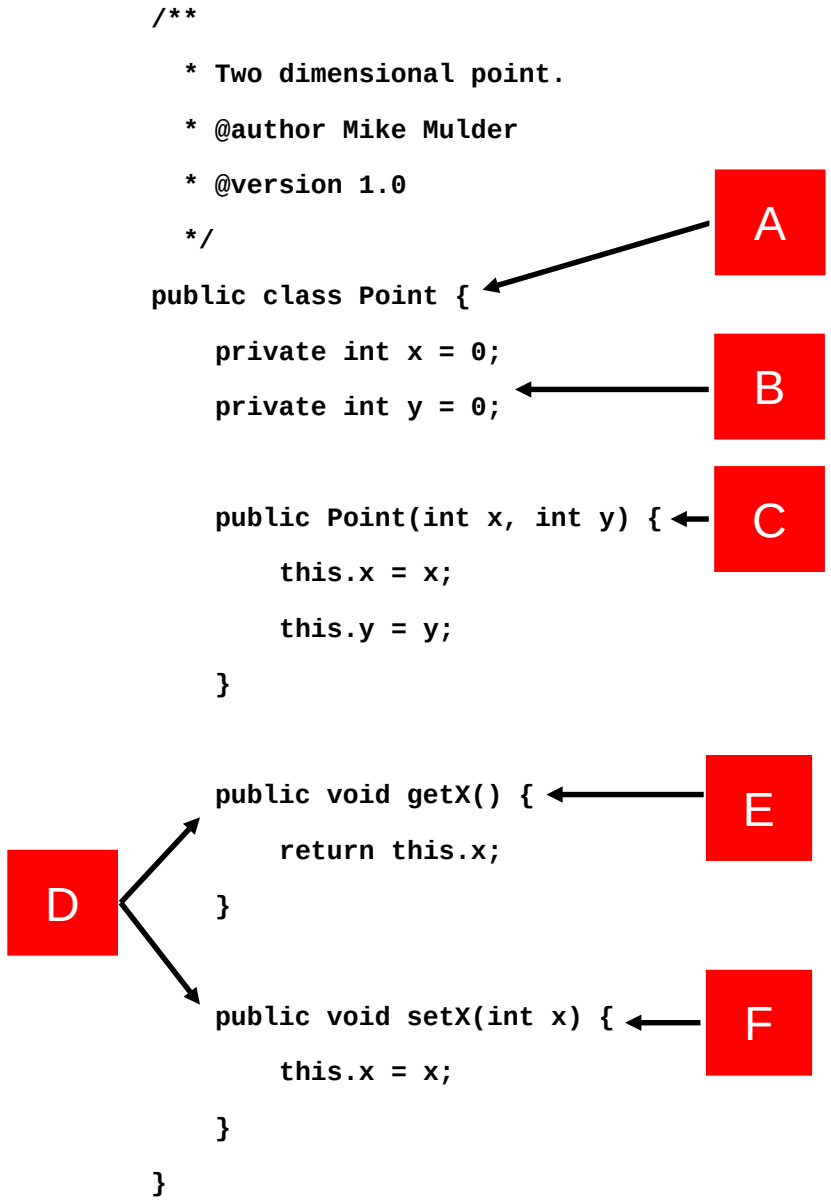
Lab 4b Questions

- Available Monday, Tuesday and Wednesday evenings for questions by e-mail

- E-mail: mmulder10@bcit.ca

- Note: Please don’t hesitate to ask questions if you unclear or stuck. In real life, you will often have to clarify requirements or rely on your co-workers to help you troubleshoot problems.

Anatomy of a Java Class



A – Class Name (and Visibility)

B – Instance Variables (including visibility and type)

C – Constructor (including visibility and parameters)

D – Methods (Today's Class)

E – Accessor/Getter/Get Method

F – Mutator/Setter/Set Method

Some Review Questions

- What is Abstraction? What is Encapsulation?

- What are the inputs of a method called? The output?

Some Review Questions

- What does void mean in the context of a method?

- What does null mean?

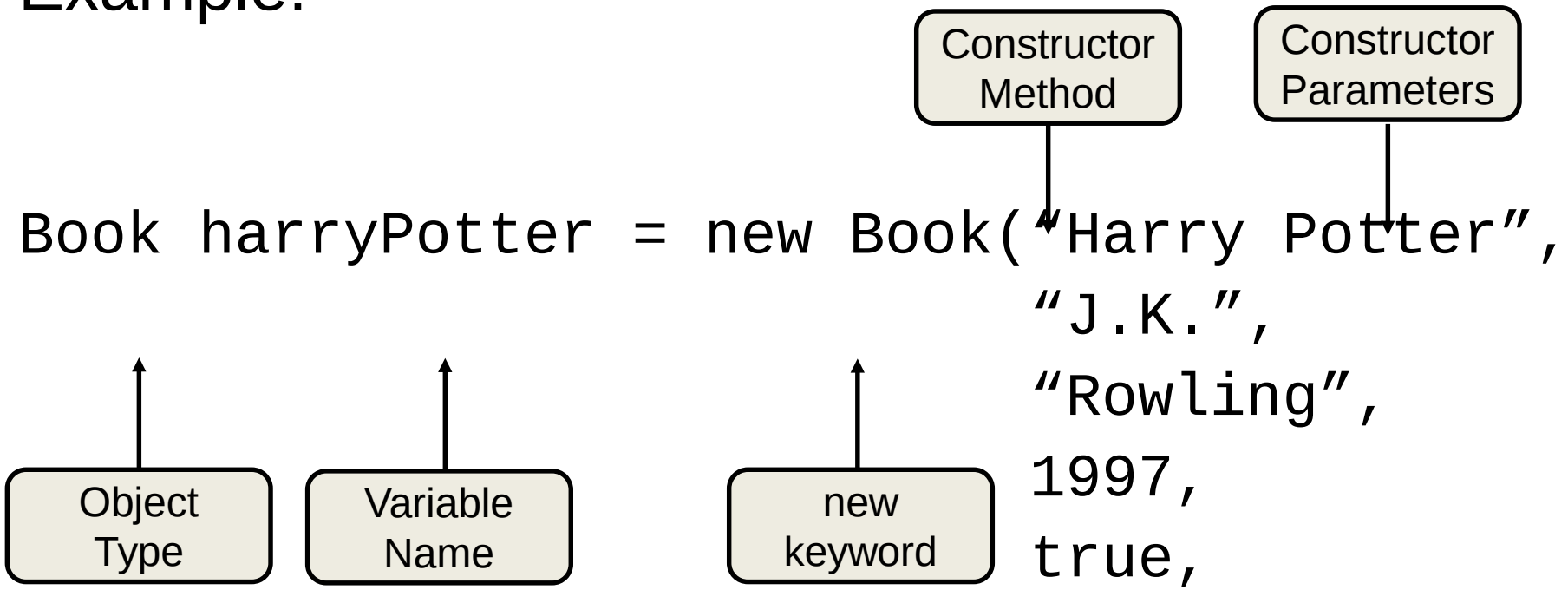
Some Review Questions

- How do we print output in Java?

- What is the operator to concatenate Strings?

Creating an Instance of a Class (Review)

Example:



Learning Outcomes: Lesson 4

- Local variables
- static variables
- static methods
- Constants (“final”)
- Multiple Constructors

Local variables

- Local variables are defined inside a **method**
- Instance variables are defined inside a **class**

- Local variables are created when the method is called
  and are destroyed when the method ends its run

- Instance variables are created when an object is created
  and are destroyed when the object is destroyed

Local variables

- Local variables are only accessible from within the method
  - local scope
  - no visibility modifier at all
- Instance variables are accessible anywhere in its class (including inside every method)
  - class scope
  - private visibility modifier

- A parameter is a type of local variable
- Local variables are NOT set to any default values by Java; you cannot use these until/unless YOU have set their values

Local variables vs. instance variables

- Scope
- Lifetime

	Instance Variable	Local Variable
Scope	Class	Method
Lifetime	As long as the Object exists	Each time the method is called

```
public class Point {

    private double x = 0.0;
    private double y = 0.0;

    public Point(double x, double y) {
        this.x = x;
        this.y = y;
    }

    public double distance() {
        double distance =
            sqrt((this.x * this.x) +
                 (this.y * this.y));

        return distance;
    }
}
```

Instance Variables

Local Variable

static variables

- static variable **values** belong to the **class**
  - Not to each object
  - Because of this, each object uses the one and only single instance
- You can access static variables without object instantiation
- Declared with the keyword **static**
  - **public static int population**

- No object required (e.g. public static void main())

static methods

- static methods, like static variables, belong to the **class**
- static methods can use only class variables aka static variables
  - But cannot access instance variables
- Can be called without object instantiation
- Also declared with **static** keyword

Constants: final

- Constants are data that are only set once
  - Unlike variables, constants never change values
- Declared with the **final** keyword at the class level
- Used to represent *things that don't change*
  - The value of pi
  - Bound checking values

- Cannot create mutator methods for final data

Symbolic Constants – static final

static finals are named in all CAPITALS with _ to separate words

- Example PI instead of pi
- **final** int NUMBER_OF_SECONDS_PER_MINUTE = 60;

static variables and methods

What are they used for?

- static variables
  - Constants (i.e., values that don't change).

- static methods
  - Utility methods that do not require object instance variables.
  - Sometimes we have classes that just provide utilities methods. Examples:
    - Complex math operations
    - String validation opertions

static variables and methods - examples

static variables

```
public static final double MIN_EARTH_MASSES = 0.0
public static final double MAX_EARTH_MASSES = 100.0
```

static methods

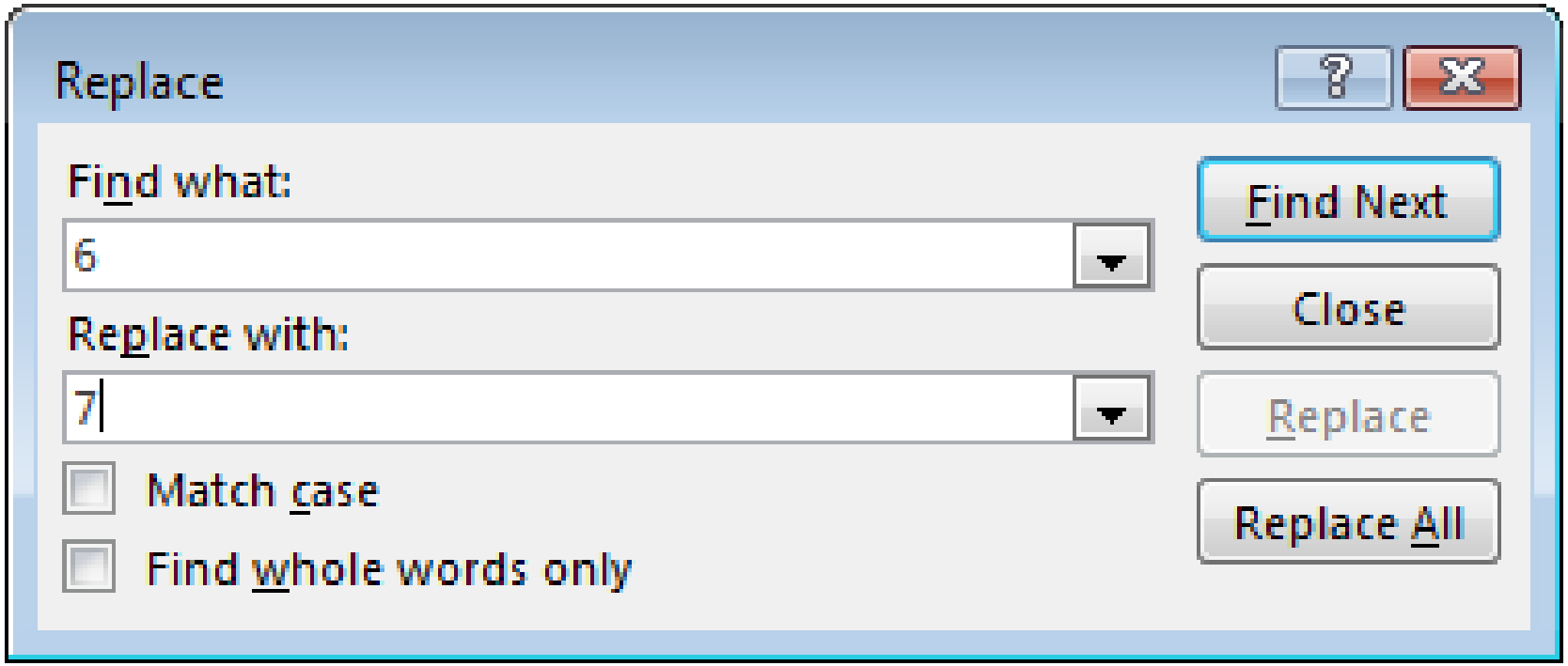
```
public static double calculateVolume(double x, double y, double z) {
    return (x * y * z);
}

public static void main(String[] args) {
    // Do stuff
}
```

Constants vs. Magic Numbers

- Tom has 6 brothers and sisters. His 6 brothers and sisters live at home. Tom wears size 6 shoes. So he has size 6 socks too. His 6 siblings have bigger feet. Every night Tom has 6 beers. His 6 drinks make him sleepy.

- Change to 7 beers.



Constants

- public static final int NUMBER_OF_NIGHTLY_BEERS = 6;
- public static final int NUMBER_OF_SIBLINGS = 6;
- public static final int SHOE_SIZE = 6;
- public static final String NAME = “John”;
- ~~public static final int SIX = 6;~~

- NAME has NUMBER_OF_SIBLINGS brothers and sisters. His NUMBER_OF_SIBLINGS brothers and sisters live at home. NAME wears size SHOE_SIZE shoes. So he has size SHOE_SIZE socks too. His NUMBER_OF_SIBLINGS siblings have bigger feet. Every night NAME has NUMBER_OF_NIGHTLY_BEERS beers. His NUMBER_OF_NIGHTLY_BEERS drinks make him sleepy.

- Change to 7 beers.

- public static final int NUMBER_OF_NIGHTLY_BEERS = 7; // that’s all

- Moral of the story: put no numbers in your code, println’s, or comments.

Summary – Static and Final

Constants (use in place of Magic Numbers)

```
public static final int NUMBER_OF_BEERS = 6;
```

(Note: public if accessible outside of the class)

Class Variables (the same for all instances of a class, but changeable)

```
private static int numberOfBeers = 6;
```

Final Instance Variables (set once in the constructor, then unchangeable)

```
private final int numberOfBeers;
```

Method Signatures (Last Class)

- A method signature includes:

  modifier return-type name(parameters)

- Examples:

```
public  boolean     isStudentEnrolled(String studentID)
private void        doSomething(double a, int b, char c)
public  double      getAverage(double num1, double num2, double num3)
private char        doSomethingElse()
```

Multiple Constructors

A class can have multiple Constructor methods
They each must have a unique signature

Constructor Signature:

modifier return-type ClassName(parameters)

Examples:

```
public Student(String firstName, String lastName, String preferredName);
public Student(String firstName, String lastName);
```

Multiple Constructors - Example

```
public class Student {
    private String firstName;
    private String lastName;
    private String preferredName;

    public Student(String firstName,
                   String lastName,
                   String preferredName) {

        this.firstName = firstName;
        this.lastName = lastName;
        this.preferredName = preferredName;
    }

    public Student(String firstName, String lastName) {
        this.firstName = firstName;
        this.lastName = lastName;
        this.preferredName = firstName;  // or can leave it undefined (i.e., null or “”)
    }
}
```

Multiple Constructors – Why?

- Constructor is **overloaded** (i.e., has multiple versions)
- Typically to account for conditions where differing amount of data is available to construct an object
- Unused instance variables are set to default values.

For example:

- Person – sometimes may not have a Preferred Name that is different than their First Name
- Preferred name could left as null, set to an empty string or set to the same value as First Name

Review Questions…

What is the scope and lifetime of a local variable?

What is the scope and lifetime of an instance variable?

Review Questions…

When would we use static variables?

When would we use static methods?

Review Questions…

What is a magic number?

When would we have multiple constructors for a class?