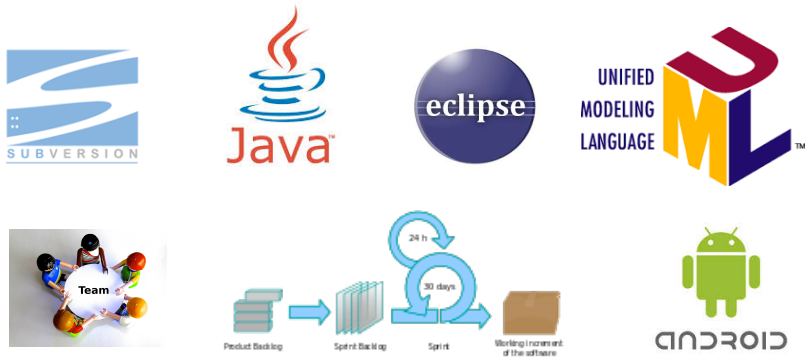
Course Wrap-up

CSC207 Winter 2015

Computer Science
UNIVERSITY OF TORONTO



10 weeks ago…

```
package basics;

public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello world!");
    }
}
```

Learning a new language

- A new memory model.
- Static typing rather than dynamic typing.
- Compiled rather than interpreted.
- Primitives vs. only objects.
- Everything belongs to a class.
- Generics.
- Interfaces and abstract classes.

You also did more of the learning yourself than before.

More languages

You'll see more languages and the underlying language design principles and approaches:

csc209: Software Tools and Systems Programming

csc324: Principles of Programming Languages

csc343: Introduction to Databases

csc309: Programming on the Web

Value:

- knowing which tool to apply in a given situation

- deeper appreciation of a language's design and how best to use it

Tools and Technologies

- A new technology: Android development
  - A new API
  - Extending Java knowledge
  - Emulator
- New software development tools:
  - A fully-featured IDE
  - Version control

Design Patterns

Language-independent.

Generic solutions to recurring design problems.

Solutions that follow the object-oriented design principles and best practices.

You'll see more design patterns in csc301: Introduction to Software Engineering.

Regular expressions

These have many practical applications:

Parsing input in a Java program (or another language).

Many Unix commands, such as `grep`, use regular expressions.

Regular expressions and related concepts are important in language design and compilers. csc488: Compilers and Interpreters

Learn more about the underlying theory in csc236: Introduction to the Theory of Computation and csc448: Formal Languages and Automata.

Software Development Process

Working in teams.

Iterative and incremental software development.

Changing requirements.

Elements of state of the art software development practices, such as Scrum and XP.

Software Development Process

You will learn much more through:

csc301: Introduction to Software Engineering

csc302: Engineering Large Software Systems

project courses (csc494/495 or csc490)

UCOSP: undergraduate capstone open-source project

PEY

Exam Prep

The exam

- 3 hours
- closed book, but there is an API at the end
  - it will be posted before the exam
- there is also some scrap paper at the end
- Javadoc and internal comments not required
- import statements not required
- helper methods always welcome
- assume all input is valid

Possible kinds of questions

Code writing

Tracing and understanding memory model

Multiple choice and short answer questions

Testing

Design patterns: recognize, explain, write code with

Regular expressions: write and read them

svn

Topics not on the exam

Android software development

Ideas for how to prepare

Solve old tests and exams.

Revisit the assignment and exercises. Learn from what you did previously; improve it.

Write code that tries to poke holes in your understanding of Java concepts (name lookup, exceptions, etc.).

Write regular expressions.

Write code that uses the design patterns we studied.

Write code on paper.

Course Evaluations

Very important! They're used by:

- Future students in choosing courses
- Faculty for improving the course
- University for evaluating faculty

Please complete them!

(Only ~6% of 12pm students and ~11% of 1pm students have completed the evals so far!)

Office Hours During Exam Period

Tentative schedule:

- Tuesday 7 April 1-2pm
- Thursday 16 April 1-2pm
- Thursday 23 April 1-2pm
- Tuesday 28 April 1-3pm

All the best!