CPSC 511
Programming Language
Fundamentals and Applications

Kris De Volder
kdvolder@cs.ubc.ca
http://www.cs.ubc.ca/~kdvolder/CPSC511/

0) Introduction and Course Overview

Course Outline

0) Introduction and Course Overview (week 1)

1) Core Semantics of Procedural Languages
   • procedures, environments, scope, continuations.
   • continuations on the web.

2) Objects and Aspects
   • fundamental concepts and semantics
   • aspects in the real world

3) Types and Checking
   • fundamental concepts
   • from theory to practice

4) TBD

Grading

Your course mark will be computed based on …

– A minimum of 4 short paper reviews (30%)
– Course project (60%)
   • A paper describing your course project, results and findings.
   • Presentation about your project given during lecture.
– Participation in/contribution to class discussion (10%)

Objectives for this Section

Explain Course Organization and Policies
grading, home work, project, paper reading and reviews

Introduce myself

Introduce Course Content and Style
• What is this course about?
• What type of course is this?
• Why is this course interesting?
• Why do we use Scheme?
Homework

Homeworks in this course are a teaching/learning tool
- NOT a grading tool
- To practice and learn the material
- To understand the code-models in a deep way

Do homeworks for yourself, not for me!
- **Homeworks don’t have to be handed in**
- Not required to solve all homeworks but recommended
- Different students, different skill sets and interests ⇒ benefit from different amount and kinds of practice
  ⇒ you are responsible use your own judgment on which homeworks you need / want to work on.

Homework: Please work with a friend!

Working together is encouraged!
- may help both understand things better!
- is more fun and more efficient!
- HW not for marks, only objective is learning.

Paper Reading & Reviews

Some classes we will
- discuss research papers related to the course material

You should
- read the papers ahead of class

As an incentive
- You will be asked to submit a short summary review **before** the class when we will discuss that paper.
- You do not have to submit a review for every paper we read, the course schedule will instruct you which papers to review/read. You are allowed to **skip at most one review**!
- Even if you do not submit a review, you are still expected to read the assigned papers.

Paper Review Format

Reviews
- short: 1 or 2 pages
- focus on key issues and questions:
  - what is the problem addressed?
  - what is the core of the proposed solution?
  - what are the main claims and novel contribution made by this paper compared to prior/related work.
  - how is the solution evaluated? how are the claims validated?

Sections of the review
- A few paragraphs, high-level summary of the paper (focus on key issues/questions from above).
- short list of points in favor or against this paper

Your review will be evaluated…
- based on how well it answers the above key issues and questions.
Project

Project ideas
- will be posted on the website but are only a guide
- you may make alternative proposals

Proposal
- By (Oct 15th) you submit a short abstract of what you are planning to do for a project.

Goal
- Apply and evaluate PL ideas, tools or techniques to solve a problem.
- Make presentation in class and write a report:
  - explain the context, related work in the area
  - how did you set up to evaluate the tools/language/…
  - what are the conclusions from this study (what did you learn, how well did the experiment work out, etc.)

Non goal:
- Does not have to be a publishable result, or novel contribution.

Term Paper (style option 1)

Written like a research paper
- Should address the following points clearly and directly:
  - what is the problem addressed?
  - what is the core of the proposed solution?
  - what are the main claims and novel contribution made by this paper compared to prior/related work.
  - how is the solution evaluated? how are the claims validated?

Note:
- It is unrealistic to expect a novel, publishable result from a course project. => This is not a requirement!
Simply describe to what extent your project is a recreation or application of existing work and provide adequate references and documentation in the related work section of your paper.
Your paper can describe a reapplication or recreation of existing research.

Term Paper (style option 2)

Written like an experience report
- Apply an existing technique, technology to implementing a system
- Paper should address
  - Related work
    - provide background on the problem/domain to which the techniques/technology are applied
    - provide background on the technology, techniques and tools you will attempt to use.
  - Explain why the technology / techniques are expected to help
    - what are the expected benefits
  - Explain how you set things up to apply the technology to try to solve your problem, implement your system.
  - Report on your experience in practice
    - How well did the technology live up to expectations
    - What advantages were observed in reality
    - What problems arose in the project, how did you address these, or how could the tools be improved to address them better in the future.
Introduce Myself

Kris De Volder
office: CICSR 309
e-mail: kvolder@cs.ubc.ca

RESEARCH
- Programming languages and Software Engineering
- Software Development Tools

Example research projects:
- JQuery: a source code browser based on queries on a source code database
- Design Rules: design a domain specific language to express "design rules" and check them against a code base.
- PointcutDoctor: IDE support for editing aspect-oriented programs.
- Control-flow breakpoints: IDE support for placing breakpoints with dynamic "control-flow-based" conditions.

What is this course about?

Objectives for this Section

- Explain Course Organization and Policies
  grading, home work, project, paper reading and reviews
- Introduce myself

Introduce Course Content and Style
- What is this course about?
- What type of course is this?
- Why is this course interesting?
- Why do we use Scheme?

Programming Languages
Fundamentals & Applications

Gain an understanding of fundamental concepts in PL implementation and semantics

How?
Through building high-level, executable models in Scheme.

See how these ideas are applied in more realistic settings.

How?
Reading research papers.
Do a course project using some more realistic tools.
Course Outline

0) Introduction and Course Overview (week 1)
1) Core Semantics of Procedural Languages
   - procedures, environments, scope, continuations.
   - continuations on the web.
2) Objects and Aspects
   - fundamental concepts and semantics
   - aspects in the real world
3) Types and Checking
   - fundamental concepts
   - from theory to practice: adding generics to Java
4) Monads and Embedded Languages
   - fundamental concepts, what are monads, monad comprehensions.
   - LNQ extension to C#.

Course Style... Why Scheme?

Scheme is an extremely simple language
- very short language specification (< 50 pages)
- very few “primitive” syntactic constructs
  - lambda, application, if, set!, define, variable-reference, literals,
- some “derived” syntactic constructs for convenience
  - let, let*, letrec, cond
- and a library of predefined procedures

But... Scheme is a very expressive language
- orthogonal language design.
- first class continuations!
- first class procedures
  => Simple yet VERY POWERFUL!

Why is This Course Interesting?

The core of Scheme is “lambda calculus”
=> Scheme is very close to the mathematical model used by programming language experts to define programming language semantics.
=> Our scheme interpreters are very close to formal models... but... they have an important advantage over them...
We can run them on a computer and see them work!
=> We can observe their properties and their behavior!
Now switch to Shriram’s presentation