SE580: Lecture 1

Alan Jeffrey

Overview

Course summary

Timetable

Administrivia

Course overview

Homework

Course summary

This course covers issues in the design and specification of object-oriented programming languages. Sample topics include the use of patterns in program representation, static and dynamic semantics, subject reduction, subtyping, inheritance, polymorphism, genericity and concurrency.

Timetable (approximate!)

1. *Overview*: getting our bearings.
2. *Syntax*: specifying and implementing syntax.
3. *Dynamic semantics*: specifying and implementing an interpreter.
4. *Static semantics*: specifying and implementing a typechecker.
5. *Method call*: dynamic and static method dispatch.
6. *Midterm exam*.
7. *Object orientation*: subclassing and subtyping.
8. *Generics*: generic classes.
9. *The binary method problem*: a standard problem with OO languages.
10. *Java 1.5*: Scaling up to a real language.
11. *Final exam*

Administrivia: contact details

Lecturer: Alan Jeffrey

Email: ajeffrey@cs.depaul.edu

Office: CST 840

Phone: (312) 362 8322

Office hours: 4.00-5.30pm Tuesdays and Thursdays.

Administrivia: reading materials

Course home page: http://fpl.cs.depaul.edu/ajeffrey/se580/, contains lectures, homeworks, pointers to tools, source code...

No textbook: this course is based on a codebase and on-line materials, not a textbook.

Administrivia: prerequisites

SE450: Object-oriented Software Development

Administrivia: required software

A Java 1.3 or 1.4 compiler, e.g. Sun's JDK.

An editor for Java source, e.g. XEmacs.

An archiving program, e.g. WinZip.

Pointers to software are on the course home page.

Administrivia: homework

This course is based on the codebase for an interpreter.

The interpreter implements Hobbes, a simple class-based, multi-threaded, object-oriented language.

Each week, an incomplete version of the interpreter will be posted on the course home page. The homework is to complete the implementation.

Administrivia: assessment

Midterm exam (16 Oct 2003): 25%

Final exam (20 Nov 2003): 25%

Weekly homeworks (submitted using Courses OnLine, best 7 out of 8): 50%

All students must attend the mid-term and final exams

Late assignments will not be accepted without medical evidence.

Plagiarism or collusion is unacceptable, and will earn an F in the course.

Course overview

What is an OO programming language?

Why are OO programming languages interesting?

What is a programming language specification?

Why are programming language specifications interesting?

Interpreters

What is an interpreter?

Why are interpreters interesting?

Look at the Hobbes interpreter: what components make up an interpreter like this?

As well as the full-featured interpreter, there is a version missing some functionality.

Each week, your homework is to add the missing functionality.

Hobbes

Hobbes is the sample OO language we will be building an interpreter for.

Why a new language? Why not C++ or Java or C# or...

The Hobbes specification is available from the course home page. This specification will grow during the course!

Syntax

What is syntax? What is an abstract syntax tree (AST)?

Why is syntax interesting? Why do we build ASTs?

How do we specify syntax? Why?

The syntactic specification of Hobbes is available from the course home page.

At the moment, Hobbes only supports simple arithmetic: no classes, interfaces, objects, if statements... This will change!

Dynamic semantics

What is dynamic semantics? What is an interpreter?

Why is dynamic semantics interesting? Why do we build interpreters?

How do we specify dynamic semantics? Why?

The dynamic semantics specification of Hobbes is available from the course home page.

Static semantics

What is static semantics? What is an type checker?

Why is static semantics interesting? Why do we build type checkers?

How do we specify static semantics? Why?

The static semantics specification of Hobbes will be available from the course home page (once I've written it!).

Strong typing

What is a strongly typed language? Is C strongly typed? Is Java strongly typed?

Why are strongly typed languages interesting?

How can we formally verify that a language is strongly typed? Why?

Object orientation

What is an object oriented language? Is C object oriented? Is Java object oriented?

Why are OO languages interesting?

Why are OO languages difficult to specify?

Hobbes is OO (at least it will be once we're finished with it!).

After the midterm...

Generics

The binary method problem

Concurrency

Homework

Download the interpreter archive and unpack it.

Change directory to `interpreter`.

Run `appletviewer index.html`.

Run `java hobbes.test.Main tests/Test*.hob`.

Oh dear, lots of failed tests! The homeworks will be to get these tests to pass.

Electronically submit your edited `Val.java`.

Next week

Specification of syntax.

Read the syntactic specification of Hobbes.