OO History and Java

- 1960s: Simula, Norway
- 1970s: Alan Kay, Xerox PARC, Smalltalk
- 1980s: Stroustrup: C++,Eiffel, Objective-C, Actor, ObjectPascal, Common Lisp Object System (Clos), …
- 1986: first OOPSLA conference
- 2001: mainstream programming

 Copyright M.Utting/B.Pfahringer

Java History

- 1991: James Gosling @ Sun develops Oak for “embedded processors”, never really used, but small/portable/robust essential
- mid 1990: WWW revolution, rename into “Java”, market as web-language, especially for “applets”

Why Java?

- The Java programming language and environment is designed to solve a number of problems in modern programming practice.
- Java started as a part of a larger project to develop advanced software for consumer electronics: embedded processors need small, reliable, portable, distributed, real-time codes.
- Intended to use C++, but encountered a number of problems. Initially these were just compiler technology problems, but as time passed more problems emerged that were best solved by changing the language.
- Java Information: `http://java.sun.com/`

Client-side computing

Off-load some computation to the client to improve:
efficiency
security

Client-side computing issues

- Byte-code, interpreters, JITS
- Security
- Specialisation of User Interfaces

White paper description

- Java is “fully buzzword compliant”!
- Java: A simple, object-oriented, network-savvy, interpreted, robust, secure, architecture neutral, portable, high-performance, multithreaded, dynamic language!
- *["The Java Language: An Overview", from Sun]*

Archimedes Inc.

- *Archimedes Inc. is a fictitious software company that produces software to teach about basic physics. This software is designed to interact with the user, not only with text and illustrations like a textbook, but also a set of software lab benches on which experiments can be set up and their behavior simulated. The most basic experiment allows students to put together levers and pulleys and see how they act. The trials and tribulations of the Archimedes' designers is used here to provide examples of Java language concepts.*

Simple

- Remove: pointers, automatic coercions, multiple inheritance, operator overloading, preprocessor, goto, structures, global vars, etc.
- Add: automatic garbage collection, large library
- *The folks at Archimedes wanted to spend their time thinking about levers and pulleys, but instead spent a lot of time on mundane programming tasks. Their central expertise was teaching, not programming. One of the most complicated of these programming tasks was figuring out where memory was being wasted across their 20K lines of code.*

Object-Oriented
(more pure than C++)

- Everything is an object! All methods are virtual.
- *The folks at Archimedes had lots of things in their simulation, among them ropes and elastic bands. Their initial C version was a pretty big system, because they had to write separate software for describing ropes versus elastic bands. When they rewrote it in an object-oriented style, they found they could define one basic object that represented the common aspects of ropes and elastic bands, and then ropes and elastic bands were defined as variations (subclasses) of the basic type. When it came time to add chains, it was a snap because they could build on what had been written before, rather than writing a whole new object simulation.*

Network-Savvy

- Can access networks as easily as files.
- Many tools/libraries, run inside browsers.
- *The folks at Archimedes initially built their stuff for CD ROM. But they had some ideas for interactive learning games that they wanted to try out for their next product. For example, they wanted to allow students on different computers to cooperate in building a machine to be simulated. But all the networking systems they'd seen were complicated and required esoteric software specialists. So they gave up.*

Interpreted

- Compile for a “Virtual Machine” (VM) based on bytecodes (.class file)
- Have an interpreter (a simulator for the virtual machine).
- “Just-in-time” compiler: translate bytecodes into machine code just before execution (+ online profiling/optimising: “Hotspot”)

Robust

- A lot more compiler and runtime checks than C++.
  (eg. impossible to overwrite memory and corrupt data, exception handling, runtime checks of casts, …)
- *The folks at Archimedes had their application basically working in C pretty quickly. But their schedule kept slipping because of all the small bugs that kept slipping through. They had lots of trouble with memory corruption, versions out-of-sync and interface mismatches. What they gained because C let them pull strange tricks in their code, they paid for in quality assurance time. They also had to reissue their software after the first release because of all the bugs that slipped through.*

Secure

- Various levels: bytecodes are checked on load, “sandbox” model, “glassbox” model
- *Someone wrote an interesting "patch" to the PC version of the Archimedes system. They posted this patch to one of the major bulletin boards. Since it was easily available and added some interesting features to the system, lots of people downloaded it. It hadn't been checked out by the folks at Archimedes, but it seemed to work. Until the next April 1st, when thousands of folks discovered rude pictures popping up in their children's lessons. Needless to say, even though they were in no way responsible for the incident, the folks at Archimedes still had a lot of damage to control.*

Architecture-Neutral

- Java executables will run on any machine!
  (provided it has a Java bytecode interpreter)
- *Archimedes is a small company. They started out producing their software for the PC since that was the largest market. After a while, they were a large enough company that they could afford to do a port to the Macintosh, but it was a pretty big effort and didn't really pay off. They couldn't afford to port to the PowerPC Macintosh or MIPS NT machine. They couldn't "catch the new wave" as it was happening, and a competitor jumped in...*

Portable

- Standard (fixed) data sizes:
  - `byte = 8 bits  float = 32 bits`
  - `short = 16 bits  double = 64 bits`
  - `int = 32 bit  unicode characters`
  - `long = 64 bits  (handle Chinese, …)`
- Libraries: Java includes libraries for graphics, sound, etc., and these are implemented on all machines (UNIX, Windows 95, Mac…)
- This means all Java programs are portable!

High Performance

- Java runtimes interpret programs
  s l o w l y . . .
- However, it is possible to translate the Java bytecodes into native machine code just before a program is run.
- These "just-in-time compilation" runtimes can make Java nearly as fast as C++.

Multi-Threaded

- Several "threads" can run in parallel. Direct standard language support for multitasking.
- *Lots of things were going on at once in their simulations. Ropes were being pulled, wheels were turning, levers were rocking, and input from the user was being tracked. Because they had to write all this in a single threaded form, all the things that happen at the same time, even though they had nothing to do with each other, had to be manually intermixed. Using an "event loop" made things a little cleaner, but it was still a mess. The system became fragile and hard to understand.*

Dynamic

- Libraries are linked in later than in C++
  (Java: at runtime, C++: at compiletime)
- Installing a new version of a library automatically updates all programs!
- Even load classes (=code) while running!
- *To expand their revenue stream, the folks at Archimedes wanted to architect their product so that new aftermarket plug-in modules could be added to extend the system. This was possible on the PC, but just barely. They had to hire a couple of new programmers because it was so complicated. This also added problems when debugging.*