Java Programming Language

Part I

November 16th, 1997

Overview

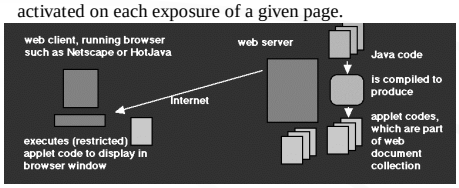
- History and alpha and later versions
- Comparison of Java and JavaScript
- And in Remaining Lectures:
  - Overall Java Philosophy and Features, including security etc.
  - Java Programming Language
  - Object-Oriented and Class Structure
  - Exceptions
  - Applet Programming and Threads
  - Abstract Window Toolkit
  - Networking and I/O
  - Future and HPCC Implications

What is Java in a NutShell?

- What is Java?
  - A simple, object-oriented, distributed, interpreted, robust, safe, architecture-neutral, portable, high-performance, multi-threaded, dynamic language.
- Java is interesting because
  - It is both a general purpose object-oriented language along the lines of C++ and it is particularly designed to interface with Web pages and to enable distributed applications over the Internet.
    - The Web is becoming the dominant software development arena; this will drive Java as the best supported, most widely taught language
    - Even outside the Web, e.g. in scientific computing, Java is as good and in some (modest) respects better than all other languages

Architecture of Java Applets

- Browsers (HotJava, Netscape 2.0/3.0/4.0, Microsoft IE ...) supporting Java allow arbitrarily sophisticated, dynamic, multimedia applications inserts called *Applets*, written in Java, to be embedded in the regular HTML pages and activated on each exposure of a given page.



Overview -- What are Java applets in detail?

- Applet constructs are implemented in terms of a special HTML tag:
  - <APPLET

    codebase="URL directory path"
    code="Java class file name"
    width=".."
    height=".."

    >

    where the URL and class file name points to a chunk of server side software that is to be downloaded and executed at the client side on each presentation of a page containing this applet which executes in window specified in size by width and height in pixels.

Running a Java Applet

Steps to running a Java Applet:

1. Write an HTML file that refers to the applet using the APPLET tag, described in later pages.
2. If necessary, write the Java code and compile into class files
3. Visit the HTML file with a Web browser or with appletviewer.
   - Using a Browser

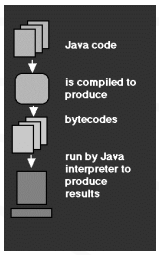
     With a Java-enabled *Web browser*, such as Netscape 2.0/3.0 or HotJava, you run an applet by "*surfing*" to a web page containing the APPLET tag.
   - Using appletviewer

     When doing development, use *appletviewer* to run applets by specifying the HTML file on the command line:

     appletviewer stock.html

Architecture of Java Applications

- Java applications are compiled and run on a machine just like any other general programming language such as C/C++.
- No Web server or network is required although Java applications may also use network connections for distributed computing.



7

Java Applications in a Nutshell

- All Java programs are written into a file with a *".java"* extension. Applications are *.java* files with a main() method which is executed first.
- How to compile and run a Java application:
  - Run the compiler on a *.java* file:
    javac MyProgram.java
    producing a file "MyProgram.class" of Java bytecodes.
  - Run the interpreter on a *.class file*:
    java MyProgram
    which executes the bytecodes.
- The resources javac and java are part of JDK and are not in Netscape and so are not necessarily available on the same machine as your web server.

8

The Simplest Java Application: Hello, World!

- Since Java is object-oriented, programs are organised into modules called *classes*, which may have data in *variables* and functions called *methods*.

Each program is enclosed in a class definition.

main() is the first routine that is run.

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

The notation class.method or package.class.method is how to refer to a public method defined within the class.The methods from some classes are automatically available to be called.

Syntax is similar to C - braces for blocks, semicolon after each statement. One difference: upper and lower case matter!

9

Java Applets

- *Java applets* are classes written in Java which are intended not to run as stand-alone programs (as applications do) but as subprograms of a browser which is already managing a window.
- Applets should NOT have *main()* method but rather *init(), start(), paint()* etc. for displaying on the browser window.
- The applet should be run through *javac* compiler getting a *.class* file as before:
  javac MyApplet.java
- Also create an HTML file (say MyApplet.html) with an applet tag to MyApplet.class.

10

How to Run an Applet?

- Two ways to run an applet:

  1) If you have JDK on one's machine, one can run the applet with appletviewer:

  appletviewer MyApplet.html

  2) Alternatively, run Netscape 2.0/3.0 or Internet Explorer essentially anywhere, point the browser at MyApplet.html, and applet is downloaded from the web server and run by Java interpreter built into the browser. This requires that the .html and .class files are located in the document space of the web server. (This way we can compile on places with JDK installed but run almost anywhere!)

11

Applet Tag: Calling Applets from HTML - I

- Given the following HTML runs the "StockGraph.class" executable as an applet:

```
<APPLET
    CODE="StockGraph.class"
    CODEBASE="http://www.javasoft.com/applets/"
    WIDTH=200
    HEIGHT=200
>
</APPLET>
```

  - WIDTH and HEIGHT are attributes that are passed along to the applet.
  - If the optional CODEBASE attribute is provided, then load the executable image from the directory specified by CODEBASE. Without the CODEBASE attribute, will look for StockGraph.class in the local server's hierarchy (relative to where the HTML was loaded) With the CODEBASE attribute will look for StockGraph.class on the given http hierarchy.
- Tag and attribute names ARE case insensitive.

12

Applet Tag: Calling Applets from HTML – II

```
<APPLET CODE="StockGraph.class"
        WIDTH=200 HEIGHT=200
        ALT="-- StockGraph Not Supported --"
        NAME=SUNW
       ALIGN=top VSPACE=5 HSPACE=5>
          Put a bunch of text here to be displayed by browsers that do not support Java
</APPLET>
```

- ALT specifies text to displayed if the browser understands the applet tag, but if unable to run applets.
- NAME specifies the name of this instance of the applet; This will make it possible for applets on the same page to find and communicate with each other.
- ALIGN specifies the alignment of the applet. The possible values are the same as those available in the IMG tag (top, middle, bottom, texttop, absmiddle, baseline, absbottom, left, right).
- VSPACE and HSPACE specifies the vertical and horizontal spacing in pixels, around the applet space.

13

<param> Tags and Applets

- The applet tag can be followed by parameters:

```
<applet . . . >
    <param name=attributename1 value="attributevalue1" >
    .......
    <param name=attributenameN value="attributevalueN" >
</applet>
```

- The Java program accesses this information by the following code segment (typically in init() method of the Applet):

```
String attribute;
attribute = getParameter("attributename1");
if  ( attribute == null )
    attribute = yourdefaultvalue;
// null is Java way of saying unset
```

14

The Simplest Java Applet: Hello, World!

- Java applets are part of the class hierarchy that can call methods to display on a screen (within the browser window). This example defines the public method *paint* in this class and calls a method *drawString* defined in the class Graphics.

The import statement (similar to an include) makes all the method names in the Graphics class in the awt package available.

```
import java.awt.Graphics;

public class HelloWorldApplet extends java.applet.Applet
{
   public void paint(Graphics g)
   { g.drawString("Hello, world!", 5, 25);
   }
}
```

Puts this as a subclass of applet.

drawString is one of many graphics methods that define what is to be displayed.

The paint method displays a graphics object on the screen - one of the standard methods that takes the place of main for applets.

15

Displaying Your Applet From a Web Page

- You should name the file with your applet name, HelloWorldApplet.java, run the compiler (*javac*), getting a bytecode file HelloWorldApplet.class, which you put in a web directory.

```
<html> <head>
<title> Simple Hello Page </title>
</head>
<body>
My Java applet says:
<applet code="HelloWorldApplet.class" width=150
height = 25>
</applet>
</body> </html>
```

Name of your applet class.

Browser uses a rectangle of width 150 pixels and height 25 pixels to display the applet.

16

Overview and History of Java Development

17

History of Java Language and Team

- Starts in 1991 by Project Green --- a group in Sun that detaches from the main campus as a semi-autonomous task force focused on operating software for consumer electronic devices such as smart set-top boxes
- Gosling (creator of Sun NeWS which had major conceptual impact both on current Java and Telescript models) realizes that C++ is not adequate and initiates development of a new language Oak, later renamed as Java.
- A PDA (Personal Digital Assistant -- codename *7) based on Oak/Java ready in 1993. Green Team incorporates as FirstPerson, Inc.
- *7 proposal to Time-Warner rejected in 1993. 3DO deal falls through in 1994. FirstPerson, Inc. dissolves.
- Small group (~30 people, now Java Team) continues development and decides to adapt Oak as a Web technology.

18

History of Java Language and Team until Dec. 95

- An experimental web browser written in Java, called WebRunner and later renamed as HotJava, ready in 1994.
- Alpha release of Java and browser HotJava April '95.
- Netscape licences Java in May '95 and builds Java into Netscape 2.0 -- This confuses ownership and open-ness of Java Beta JDK (Java Development Kit) published in summer/fall '95. It is better software but lower functionality than Alpha.
- First alpha Java books appear in fall '95 such as a popular overview by SAMS and technical book "Java!" by Tim Ritchey, edited by New Riders.
- Dec 4 1995 Business Week cover story on "Software Revolution --- The Web Changes Everything" exposes Java as a breakthrough force in the expanding Web/Internet.
- Also points out that "Java as a business" is yet to be defined.
- In next week, SGI, IBM, Adobe, Macromedia, and finally Microsoft adopt/license Java. Java goes into open standards process and is adopted by Web community.

19

JDK 1.0 -- The Java Development Kit!

- The beta version of Java, Version 1.0 of JDK released January 96 by JavaSoft.
- JDK 1.0 becomes the Internet standard and so compatible with ongoing Java implementations by all licensees, most notably Netscape and Microsoft. Beta/1.0 JDK includes:
  - Java compiler (.java to .class) for Sun Solaris and Windows NT/95 avaliable as .class itself (javac written in Java) but no source
  - Java source for the foundation classes (modified and incompatible with alpha) appletviewer to run/preview applets
  - Source code for the interpreter

20

JDK 1.1 -- Java Grows!

- A substantial new version of Java released March 97 by JavaSoft.
  - This release includes many developments both by Sun and by partner companies such as IBM. There are minimal changes to the language - primarily development of new classes to support enterprise computing.
- Currently (Nov 97) Netscape4.0, Internet Explorer 4.0, and HotJava do support JDK 1.1 including the Java DataBase Connection (JDBC)
  - http://www.javasoft.com/products/jdk/1.1

21

Java Web Servers

- Originally, the Java Interpreter was incorporated into browsers such as those from Netscape and Microsoft, but the Web server remained a standard one. Now Web servers are being developed in Java itself. This leads to more natural integration of the use of Java applets on the Web browsers and Java applications running on the Web server machine.
- Java WebServer from Sun - http://jserv.javasoft.com/ - This web server is developed following new Java Server API, which allows for security and encryption servers. WebServer will run Java applications, called servlets, on the server side similarly to CGI scripts.
- Jigsaw from the W3 Consortium -http://www.w3.org/pub/WWW/Jigsaw - This web server had the most services but is currently being rewritten so that the model for running Java on the server side will be JavaBeans.

22

Java vs. JavaScript

23

Comparison of Java and JavaScript – I

- Netscape renames Livescript as Javascript and this is an interesting variant of Java which is *fully interpreted* (code can be included directly in HTML file)-- use for overall customization of client
- Use Java for detailed programming and JavaScript for overall integration of client interface and system
- JavaScript: Interpreted by client and NOT compiled
- Java: Compiled on Server before execution on client
- Note both are reasonably "pure" C/C++ like languages and do NOT have useful sh/awk text and system enhancements of Perl(5)
- JavaScript: Object based -- no classes or inheritance -- built in extensible objects
- Java: Object-oriented. Programs consist of object classes with inheritance

24

Comparison of Java and JavaScript – II

- JavaScript: Integrated with HTML as embedded ASCII but of course HTML looks rather irrelevant at times!
- Java: Applets distinct from HTML but invoked from HTML Pages
- JavaScript: do not declare variables' datatypes -- Loose typing
- Java: MUST declare variables' datatypes -- Strong typing
- JavaScript -- Dynamic Binding -- object references computed at runtime
- Java -- Static Binding -- object references must exist at compile time
- Java and JavaScript are secure and cannot write to disk
- JavaScript has most user interface features of Java (such as buttons and frames), except not mouse interactions like "dragging".

25

Overall Java Philosophy and Features

26

Some Key Java Features

- Document: *The Java: A White Paper by Sun Microsystems* -- October 1995 draft by James Gosling and Henry McGilton -- enumerates the original design of Java:
  - Simple and Familiar
  - Object-oriented
  - Architecture-neutral
  - Portable
  - Somewhat Interpreted
  - Distributed
  - Robust
  - Secure
  - High performance
  - Multi Threaded
  - Dynamic

27

Java Features -- It's Simple and Familiar!

- Java omits several rarely used, poorly understood and confusing features of C++ including operator overloading, multiple inheritance, pointers, and automatic type coercions.
- It adds automatic garbage collection which makes dynamic programming easier in Java than in C or C++. No more mallocs!
- It also adds 'Interface' construct, similar to Objective C concept, which often compensates for the lack of multiple inheritance by allowing method calling syntax to be "inherited".
- The resulting language is familiar as it looks like C++ but is simpler and hence easier to program in.
- It also results in a much smaller kernel which is suitable for planned Java ports to consumer electronic devices. Base (alpha) interpreter is ~40Kb, libraries and threads add additional 175Kb.

28

Java Features -- It's Object-Oriented

- Java model can be viewed as a C++ subset, with some dynamic elements inherited from Objective-C (method overloading, garbage collection).
- Structures, Unions, and Functions are absorbed into data and methods of Java classes -- Java is Simple!
- The strength of Java object-oriented model is not in sophistication but in simplicity and the extensive class library associated with the system (some 250 public classes were released in both alpha and beta).
- Java class plays also a role of a communication atom in the Web embedding model.
  - Applet classes identify themselves by names in the HTML applet tag.
  - Applet downloads other classes, present in the applet source.

  (Hence, the Java class names play the role of addressing mode for the distributed Java code database.)

29

Java Features -- It's Architecture-Neutral

- C/C++ programming in a heterogeneous network environment requires use and compatibility across several vendor platforms and the corresponding compilers. This problem is solved in Java by designing platform-independent binary representation called Java bytecode (or opcode).
- Java compiler (written in Java and platform-independent) reads Java source and generates Java bytecode. These bytecodes are shipped to client machines upon browser requests.
- Each client machine must run Java interpreter which performs runtime execution of Java bytecodes. Java interpreter is written in POSIX compliant ANSI C and needs to be ported to and conventionally compiled (once) on each individual platform.
- Once the interpreter is ported, application developers don't need to worry at all about platform specificity and differences between native compilers.

30

31

Java Features -- It's Portable

- Java language offers a uniform abstract (virtual) machine model which is identical for all platforms.
- SUN owns the Java Virtual Machine (see online report) -- it is universal while classes can be added by any user
- Unlike in C/C++ where various integers match the architecture of a physical machine at hand, Java byte, char short, int and long are always of the same size, equal to 8, 16, 16(unicode), 32 and 64 bits, respectively.
- No header files, preprocessors,#define etc.
- Floating point is always IEEE 754
- Differences between vendor specific windowing environments (X Windows, MS Windows, Macintosh) are removed in terms of the Abstract Windowing Toolkit (AWT) metaphor.
- AWT is given by ~60 Java classes (alpha) which offer a universal GUI programming model, portable between UNIX, PC and Mac, and translated automatically to native windowing systems on individual platforms by Java interpreters.

32

Java Features -- It's Somewhat Interpreted

- Java represents a compromise between fully compiled (like C/C++) and fully interpreted (like Smalltalk or Perl) models.
- Java "compiler" produces a binary bytecode output which is portable and much smaller than the real binary for a specific machine (Typical bytecode size is of order of the original source code, within a factor of 2).
- Java "interpreter" executes this bytecode and is therefore less dynamic than e.g. Perl interpreter (which performs an equivalent bytecode construction internally and on-the-fly when reading the program source).
- In general, the compilation process is: a) time consuming and b) platform specific. Hence, interpreters are built and used to facilitate a) rapid prototyping and/or b) portability. Java model is focused on platform independence but the development throughput is also reasonable since the Java compiler is fast and generates compact bytecode output.

33

Java Features -- It's Distributed (and can support parallel computing)

- Popular TCP/IP based protocols such as FTP or HTTP are supported in terms of network protocol classes.
- This implements Java plus message passing and immediately supports various forms of distributed processing.
- New protocols, such as PVM and MPI, can be added and dynamically installed. Parallel computing can be built on top of these base classes.
- Distributed computing model of Java is mainly client-server, with Java compiler preparing the opcodes at the server side, and Java interpreter executing it at the client side.

34

Java Features -- It's Robust

- Java enforces compiler-time type checking and eliminates this way some error prone constructs of C/C++.
- Pointer arithmetic is fully eliminated which allows e.g. for runtime checking of array subscripts and enforces security of the Java model.
- Explicit declarations are always required, i.e. C-style implicit declarations are abandoned. This allows the Java complier to perform early error detection.
- Rapid prototyping in Java is less natural than in JavaScript,Lisp, Tcl, Smalltalk or Perl, but the software quality assurance of Java is higher than in these more dynamic and 'forgiving' languages.

35

Java Features -- It's (Hopefully) Secure

- Java bytecodes are shipped across the network and executed on client machines. Security is therefore a critical issue and strongly enforced in Java.
- Java contains its own networking classes which are designed to be secure
- Modifications of the C++ model such as eliminating pointer arithmetic and coercion were dictated mainly by the security requirements.
- Most viruses are based on acquiring access to private/protected sectors of computer memory which is impossible in Java.
- Java opcodes are executed at the client side by Java interpreter which operates exclusively on the virtual memory. Hence, unless there are security bugs in the Java interpreter itself, the model is safe and users cannot create security holes by incorrectly or maliciously written applets.
- The bytecodes sent across network are verified at the client which prevents evil/corrupted classes from causing problems

36

Java Features -- High Performance

- Java interpreter performs on-the-fly runtime execution of the Java bytecodes which results typically in a satisfactory performance.
- NOT true in initial software which is often 100 times slower than C
- performance is improved in new "just-in-time" interpreters, which saves code for repeated sections to provide compiled code efficiency after first execution
- Support for generating native machine code out of Java bytecodes, viewed as intermediate compiler form, is also provided and useful for performance demanding applications.
- The performance of the machine code, generated from Java bytecodes, is comparable to that offered by typical C/C++ compilers on the same platform.
- Several of these concepts are in fact similar as in the OSF/ANDF project. Using ANDF terminology, we would call Java compiler a 'producer', and the machine code generator discussed here, an 'installer'. Default Java working mode doesn't use installers but directly interprets the intermediate form (this mode is supported in ANDF by GAI -- Generalized ANDF Interpreter).

37

Java Features -- It's Multithreaded

- Java model offers preemptive multithreading, implemented in terms of the Thread class. Thread methods offer a set of synchronization primitives based on monitor and conditional variable paradigm by C.A.R. Hoare. Java threads inherit some features from the pioneering Cedar/Mesa System by Xerox Park that gave birth to Macintosh and object-oriented programming.
- A typical use of Java multithreading in applet programming is to have several independent but related simulations (e.g. various sorting algorithms), running concurrently in an applet window. Multithreading is also used internally by the browser to handle multiple document dynamics.
- Another interesting application domain are multi-HotJava environments to come such as collaboratory or gaming.
- Java threads don't have built-in point-to-point communication primitives. Various thread communication environments can be provided by coupling the thread and network protocol objects.

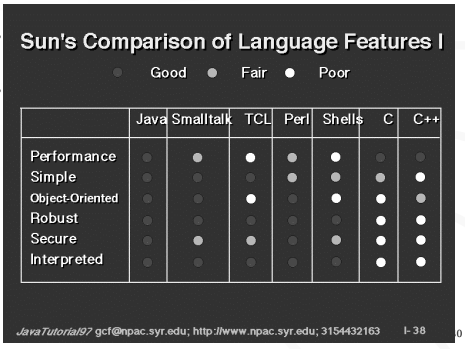
38

Java Features -- It's Dynamic

- Java model is more dynamic than C++ and closer to Smalltalk or Perl.
- Subclasses don't need to to be recompiled after superclass implementation is updated.
- C++ has "fragile superclass" problem where must recompile children if change anything (method/instance variable) in a superclass or referenced class -- Java resolves references at runtime and avoids this.
- Classes have runtime representation (implemented in terms of the Class class) which allows one to look up type of a given object instance at runtime (in C cannot know if pointer is to integer or browser!)

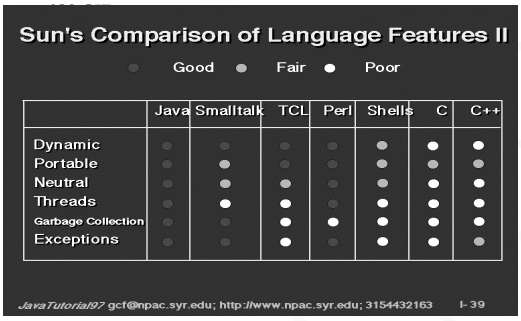
39

Sun's Comparison of Language Features I



40

Sun's Comparison of Language Features II



41

Java Books – I

- Teach Yourself Java in 21 Days, 2nd ed., by Laura Lemay and Charles L. Perkins, Sams.net Publishing, is a "how-to" book at the intermediate programming level, greatly expanded from the original edition.
- Java in a Nutshell, by David Flanagan, is the language reference book in the familiar O'Reilly series. The 2nd edition of this book is now out - it omits many examples from the first edition to make room for large section on JDK 1.1 - currently best book reference. Both 1st and 2nd edition are quite useful.
- Java, How to Program, by Deitel and Deitel, Prentice-Hall, starts with beginning programming concepts and progresses rapidly through Java language. It has the most programming exercises and also has companion teaching multimedia books.

42

Java Books – II

- The Java Programming Language, by Ken Arnold and James Gosling, Addison-Wesley, May 1996, has lots of details on the language basics for intermediate and advanced programmers. It covers threads and i/o packages, but not applets or windowing packages. All serious computer scientists should read to understand fundamentals
- Java Primer Plus, supercharging Web applications with the Java programming language, by Paul M. Tyma, Gabriel Torok, and Troy Downing, Sams.net, doesn't assume a lot of programming background, has chatty explanations and still covers lots of programming detail.

43

Java Beta Books – III

- There are now many books in the Java Series from SunSoft Press, Prentice-Hall. Here are the first five:
  - Instant Java*, by John A. Pew, contains multimedia and animation applets for HTML authors. This is not a programming book.
  - Java by Example, by Jerry R. Jackson and Alan L. McClellan, covers all key features with examples, but not as much detail about the language.
  - Just Java, by Peter van der Linden, for intermediate programmers, gives good explanations of key features without going into detail.
  - Core Java, by Gary Cornell and Cay S. Horstmann, offers detailed coverage of the language and packages for advanced programmers.
  - Graphic Java, by Gary McClellan, gives more details on windowing and user interface and includes new classes (not in originial Java release) for such things as "rubberbanding".

44

Resources for the Java Programming Language

- The original resource was the The Java Language Specification by Sun Microsystems, Inc., March 1995 updated to October 1995 but superseded by Gosling and Arnold Book
- Addison Wesley has several other fundamental Java books on Application Programming Interface and Language Specification and Virtual Machine (by end of summer 1996)
- http://www.javasoft.com web site has plenty of references including
- Tutorial: http://www.javasoft.com/books/Series/Tutorial/index.html Books:http://www.javasoft.com/java.sun.com/aboutJavaSoft/book-news.html Collection of Applets: http://www.gamelan.com
- Most of the books cited earlier have CDROM's with examples and the JDK.

45