Java Tutorial - Spring 2000

Part 1: Introduction to Java Language and Applets

http:/ / www.npac.syr.edu/ projects/ tutorials/ Java/
Instructors: Geoffrey Fox , Nancy McCracken
Syracuse University
111 College Place
Syracuse
New York 13244-4100

Abstract of Java Tutorial

- Part 1:
  - Overview including History of Java Development
  - Overall Java Philosophy and Features including security etc.
- Part 2:
  - Java Programming Language
  - Object Oriented and Class Structure
  - Exceptions
- Part 3:
  - Applet Programming and Threads
  - Abstract Windowing Toolkit
- Part 4:
  - Networking and I/ O
  - Multithreading

What is Java in a NutShell?

- What is Java?
  - A simple, object oriented, distributed, interpreted, robust, safe, architecture neutral, portable, high performance, multithreaded, 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
  - Particularly good as a language for K-12 teaching
  - Even outside the web, e.g. in scientific computing, Java is as good and in some (modest) respects better than all other languages

Java is an important language in the world

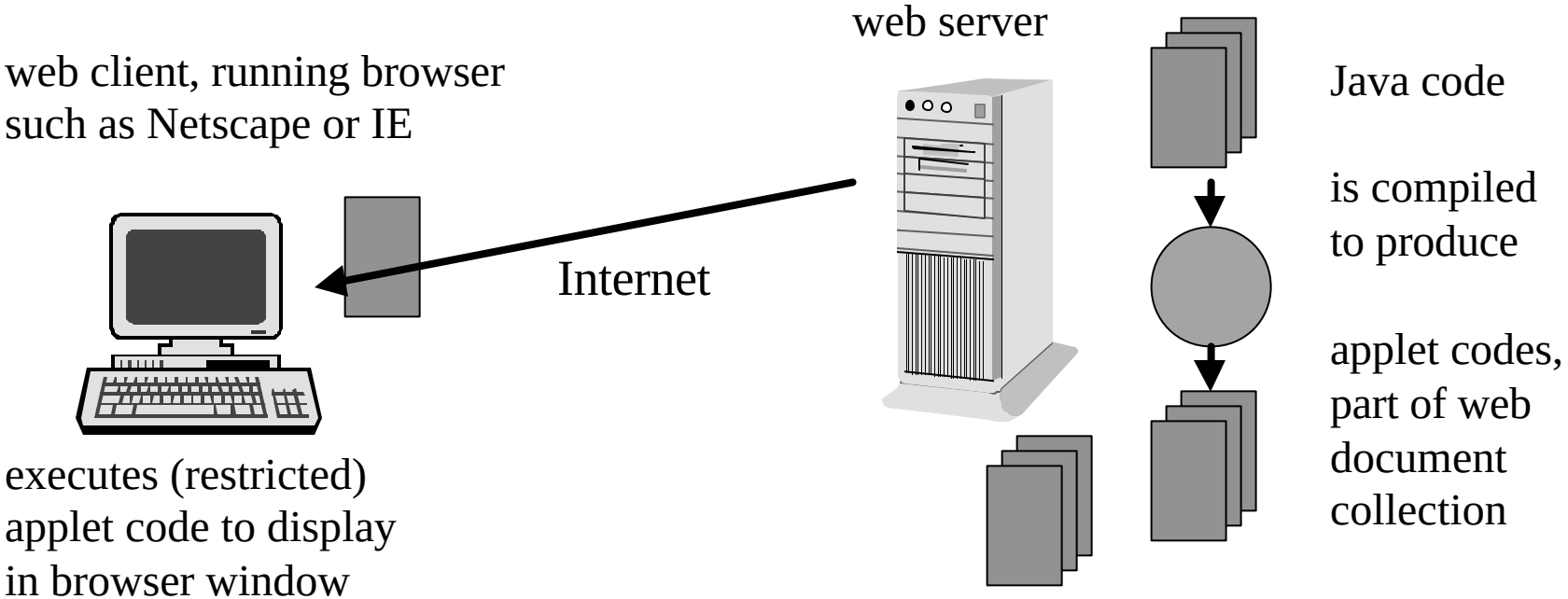
- The Java Language has several good design features
  - secure, safe (wrt bugs), object-oriented, familiar (to C C++ and even Fortran programmers)
- Java has a very good set of libraries covering everything from commerce, multimedia, images to math functions (under development at http:/ / math.nist.gov/ javanumerics)
- Java has best available electronic and paper training and support resources, growing labor force trained in Java
- Java is rapidly getting best integrated program development environments
- Java naturally integrated with network and universal machine supports powerful “write once-run anywhere” model

Java is also important in computer science

- Increasingly, Java is the language used for important computing paradigms that support applications: object-oriented computing, event-driven computing, distributed objects, linkage to databases, visual/ component computing, client/ servers, networking, multimedia computing . . .
- So Java is an important language to include at an advanced computer science level, along with other languages such as C++, that are useful to students to get jobs.
- But the good design features of Java also make it suitable for teaching basic computer science concepts: algorithms, data structures, software design, . . .
  - See the (old, but still relevant) discussion by Doug Lea at http:/ / gee.cs.oswego.edu/ dl/ html/ javaInCS.html

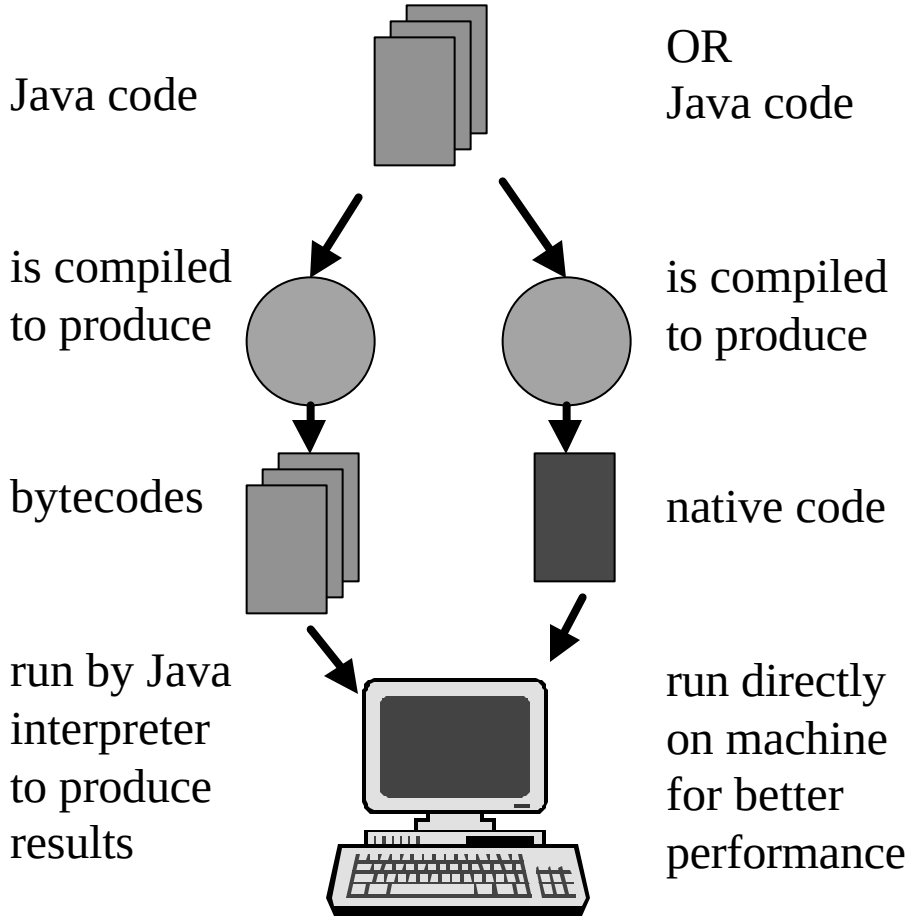
Architecture of Java Applets

- 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.



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 are required although Java applications may also use network connections for distributed computing.



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 excuted first.
- How to compile and run a Java application (via bytecodes):
  - 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.

The Simplest Java Application: Hello,World!

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

Each program is enclosed in a class definition.

main() is the first method that is run.

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

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

The notation class.method or package.class.method is how to refer to a public method (with some exceptions).

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 a main method but rather init, start, paint etc. for displaying on the browser window
- Applets are not trusted as a default, so they have several restrictions on running on the client machine
  - no printing or file I/ O
  - cannot connect through the network to any machine but its own server
  - any new windows created by the applet have a warning label

Preparing an Applet

- The applet should be run through javac compiler getting a .class file as before:

  javac MyApplet.java

- The resulting file MyApplet.class is then stored in the document collection of a web server (hence has a URL location).
- Also create an HTML file (say MyApplet.html) with an applet tag to MyApplet.class.
  - <APPLET
    codebase="http:/ / myserver.org/ mydirectory"
    code = "MyApplet.class" width=300 height=100>
- When the browser loads the .html file, it will also download the .class file and invoke the java interpreter to run the init, start, and paint methods.

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). One way to draw on the screen is to call the method drawString from the standard method paint.

The import statement (similar to an include) allows the use of methods from the Graphics class .

```
import java.awt.Graphics;

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

Puts this as a subclass of Applet.

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

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.

The browser will use a rectangle of width 150 pixels and height 25 pixels to display the applet within the other html.

More Details on Applet Tags - I

- Given the following HTML
  - <APPLET CODE="StockGraph.class"
  - CODEBASE="http:/ / www.javasoft.com/ applets/ "
  - WIDTH=200 HEIGHT=200>
  - </ APPLET>
- Runs the "StockGraph.class" executable as an 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), otherwise look for StockGraph.class on the given http hierarchy.
- May also supply an ARCHIVE attribute which specifies a ".jar" file to download (akin to tar on UNIX). The class named in the CODE attribute should be one of the classes in the jar file.
- Tag and attribute names are case insensitive.

More Details on Applet Tags - 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 such as Netscape 2.0 on Windows 3.1 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).
  - Align=top which aligns top of applet with top of tallest item in the line
  - Align=texttop which aligns top of applet with top of the tallest text in the line
- VSPACE and HSPACE specifies the vertical and horizontal spacing in pixels, around the applet space.

<param> Tags and Applets

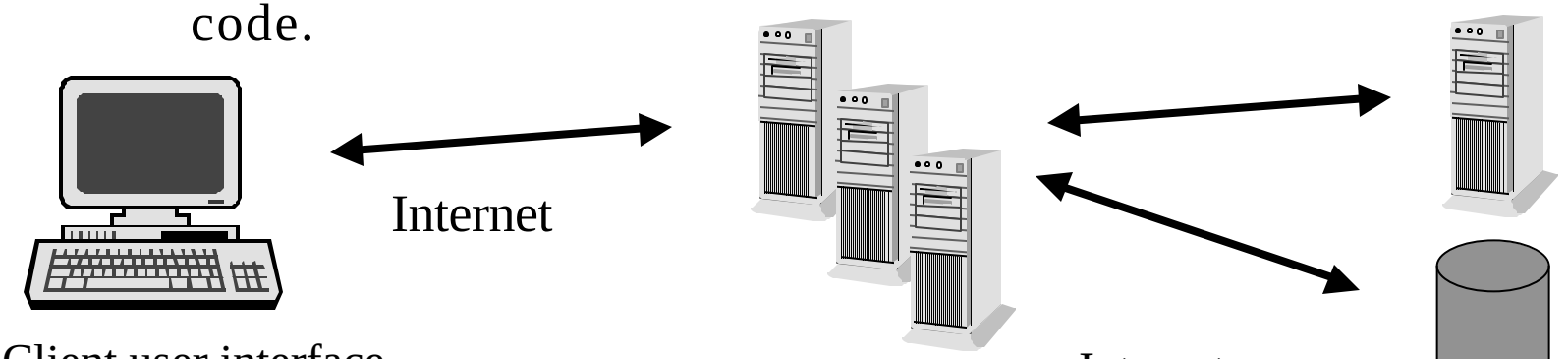
- The applet tag can be followed by parameters:
- <applet . . . >
- <param name=attributename1 value="attributevalue1" >
- .....
- <param name=attributenameN value="attributevalueN" >
- </ applet>
- The Java program accesses this information by
- String attribute;
- attribute = getParameter("attributename1");
- if(attribute == null)
- attribute = yourdefaultvalue;
- / / null is Java way of saying unset
- Typically this processing would be in init() method of Applet

Java vs. JavaScript

- Despite the word “java” in the name, JavaScript is a different language than Java, albeit with some similarities.
- A JavaScript program is written directly into the HTML page, and is executed by the JavaScript intrepeter, so also executes dynamic web page content in the browser window.
- JavaScript is special purpose - it is an object-based language that deals directly with browser entities such as windows, textfields, forms, frames and documents.
- JavaScript can respond to browser events (not as many as Java’s more complex capabilities with the user interface) such as mouse clicks and user-typed text.
- JavaScript is easy and fast to write, but not as powerful as Java.

Multi-tier Architecture

- Distributed applications on the web naturally have a multi-tier architecture.
- Java plays a role at all three levels:
  - Graphical User Interface and client side analysis systems, including visualization
  - Middle layer servers and software integration, including web servers, distributed object servers and other application servers.
  - Less important for backend client software, which may be legacy code.



Overview and History of Java Development

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.

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.

More Recent Java History

- 1996 JavaOne Developer's Conference (now held annually in June) focused on applets. Attendance of 5000 people.
- 1997 JavaOne focused on JavaBeans and new version of the language JDK1.1, both designed to put Java in line with other developments in component and secure distributed web object computing and use of databases.
- 1998 JavaOne featured attendance of 14,000+ and focused on Enterprise JavaBeans and other Enterprise classes, developing more on server-side applications.
- 1999 JavaOne had over 20,000 attendees. Continued development of JINI for dynamic networking, embedded devices such as Palm Pilots.

Overall Java Philosophy and Features

Some Key Java Features

- First we discuss original Java base language features as discussed in 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
- Finally we mention additional features added to Java in more recent versions.

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.

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 is 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.

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.

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.

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.

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.

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.

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

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).
- Java/ HotJava system implements ANDF concepts for the Java language.

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.

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!)

Sun's Comparison of Language Features

◆ ☆ Good □ Fair ○ Poor

	Java	Smalltalk	TCL	Perl	Shells	C	C++
Performance	☆	□	○	□	○	☆	☆
Simple	☆	☆	☆	□	□	□	○
Object-Oriented	☆	☆	○	☆	○	○	□
Robust	☆	☆	☆	☆	☆	○	○
Secure	☆	□	□	☆	□	○	○
Interpreted	☆	☆	☆	☆	☆	○	○
Dynamic	☆	☆	☆	☆	□	☆	○
Portable	☆	□	☆	☆	□	□	□
Neutral	☆	□	□	☆	□	○	○
Threads	☆	○	○	☆	○	○	○
GarbageCollection	☆	☆	○	○	○	○	○
Exceptions	☆	☆	○	☆	○	○	□

JDK 1.1

- 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.
- Netscape4.0 and Internet Explorer 4.0 do support JDK 1.1, on some platforms you must explicitly download the “AWT” version to get all of 1.1 support.
- Features: better event model in the AWT, Java Database Connectivity (JDBC), Remote Method Interface (RMI), internationalization, security certificates and encryption, and JavaBeans.
- http:/ / www.javasoft.com/ products/ jdk/ 1.1

JDK 1.2, newly renamed to Java 2 Platform

- This latest version of Java has additional classes.
- It is not currently supported in final release browsers, but a browser plug-in may be downloaded to support it.
- It features classes for the “Swing Set”, which is many new user interface elements with a more sophisticated look and feel, drag and drop, text editors, printing formatted text, 2D and 3D geometry, image processing and more multimedia classes.
- Adds API for CORBA.

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.

Java Books -- I

- Java in a Nutshell, by David Flanagan, is the language reference book in the familiar O'Reilly series. The 2nd edition of this book omits many examples from the first edition to make room for large section on JDK 1.1 - currently best book reference. Also Java Examples in a Nutshell.
- 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. The third edition has Swing Set and also the advanced API's.

Java Books -- II

- The Java Programming Language, by Ken Arnold and James Gosling, Addison-Wesley, May 1996, is the classic 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
- core Java, by Gary Cornell and Cay S. Horstmann, offers detailed coverage of the whole language and packages for advanced programmers. Also Volume 2 gives good coverage of advanced topics such as JDBC, RMI, JavaBeans and security. The 4th edition includes the Swing set and Java 2.

Java Development Environments

- These range from simple tools that give a windowing interface to the edit/ compile/ run or view cycle:
  - JavaEdit from Dick Chase on PC's
  - JADE (under development at NPAC for classes)
- to the elaborate commercial development environments that can also track projects and help generate code for user interface components
  - Microsoft Visual J++
  - Symantec Visual Café
  - Java Workshop from Sun
  - Borland Jbuilder
  - Kawa from Tek-Tools (does support Java2)
  - . . .

Resources for the Java Programming Language

- The original resource was The Java Language Specification by Sun Microsystems, Inc., March 1995 updated to October 1995 but superceded by Gosling and Arnold Book
- 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.