



core
WEB
programming

Introduction to Java

© 2001-2003 Marty Hall, Larry Brown <http://www.corewebprogramming.com>

Agenda

- **Unique Features of Java**
- **Java versions**
- **Installation and running Java programs**
- **Basic Hello World application**
- **Command line arguments**
- **Basic Hello WWW applet**

Java is Web-Enabled and Network Savvy

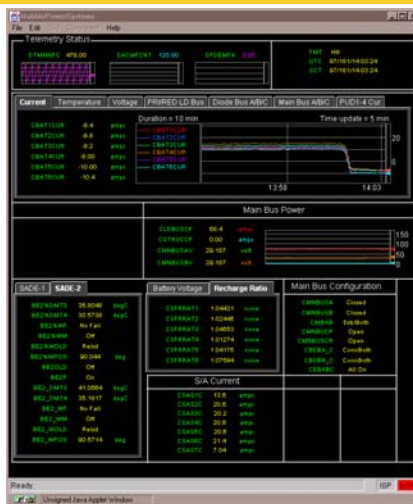
- **Safety in Java programs can be enforced**
 - Array bounds never violated; no address manipulation
 - Types enforced
- **The Web can deliver Software**
 - No more installation or updates; just a bookmark
- **Java's client/server library is easy to use**
 - Ordinary mortals can do network programming
- **Distributed Object Protocol and DBMS API**
 - RMI and JDBC

3

Introduction to Java

www.corewebprogramming.com

Hubble Space Telescope Monitoring



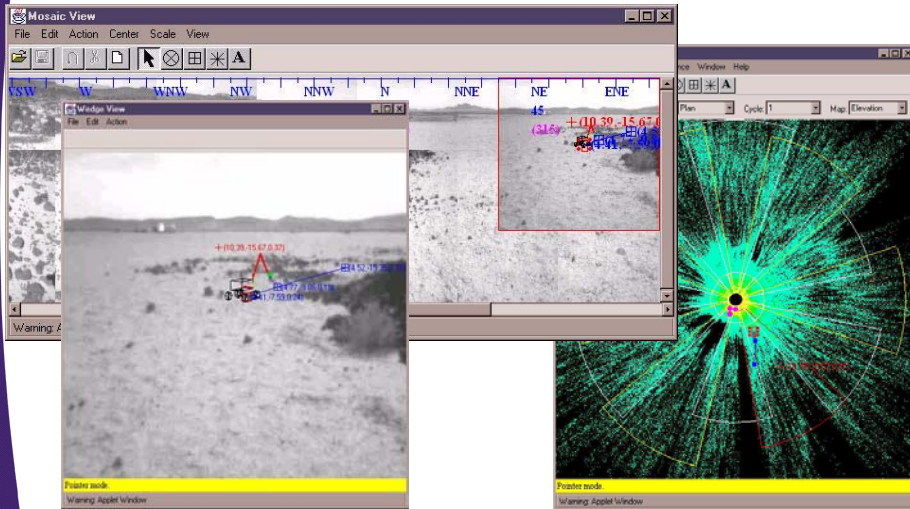
“NASA Goddard’s Most Successful Software Project Ever”

4

Introduction to Java

www.corewebprogramming.com

Mars Pathfinder Mission Simulator



Used for world-wide data viewing

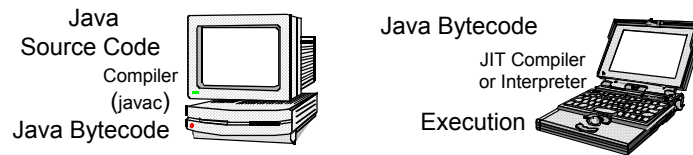
5

Introduction to Java

www.corewebprogramming.com

Java is Cross Platform

- **Compiles to machine-independent bytecode**



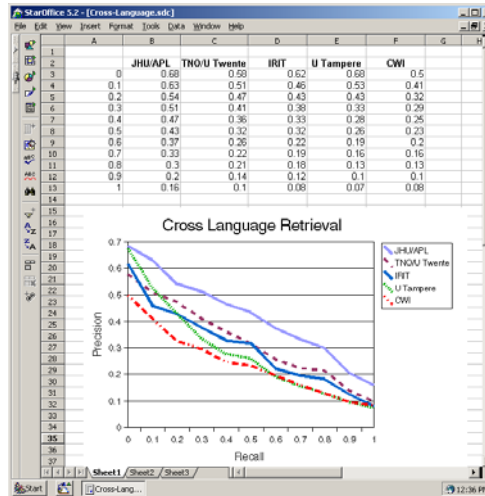
- **Windows, MacOS, OS/2, Solaris, ...**
- **Java has a portable graphics library**
- **Java avoids hard-to-port constructs**

6

Introduction to Java

www.corewebprogramming.com

StarOffice 5.2



Cross-platform office suite completely written in Java

www.corewebprogramming.com

7

Introduction to Java

Java is Simple

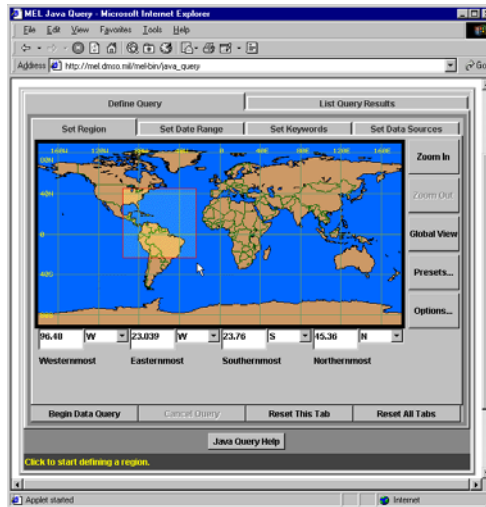
- **Java has automatic memory management**
 - No dangling pointers
 - No memory leaks
- **Java simplifies pointer handling**
 - No reference/dereference operations
- **No makefiles/No header files**
- **C++ syntax streamlined**

8

Introduction to Java

www.corewebprogramming.com

MEL - Master Environmental Library



Interactive geospatial data discovery and retrieval

www.corewebprogramming.com

9

Introduction to Java

Java is Object Oriented

- **All functions are associated with objects**
 - “Member functions” are only functions
 - Some describe it “object-obsessed”
- **Almost all datatypes are objects**
 - Files, arrays, strings, sockets, etc.
 - Still have “primitive” types for efficiency
 - byte, short, int, long, float, double, char, boolean
- **Object** is a common ancestor of all classes

10

Introduction to Java

www.corewebprogramming.com

Java is Rich with Powerful Standard Libraries

- **Threads (lightweight processes)**
- **Building and using data structures – Java Foundation Classes**
- **Parsing strings/streams**
 - JDK 1.4 supports Regular Expressions
- **Arbitrary precision integers and fixed-point arithmetic**
- **Serialization (saving object state to disk or sending via socket)**
- **Invoking remote objects – RMI**
- **Interfacing with relational databases – JDBC**
- **And many more ...**

Java Versions

- **Java 1.0 released in 1995**
- **Java 1.1 released in early 1997**
 - A new event-handling model based on listeners
 - Remote method invocation (RMI) and object serialization
 - Support for inner and anonymous classes
 - Arbitrary precision integers and floating-point numbers
 - Java DataBase Connectivity (JDBC) API for connecting relations databases
 - JavaBeans component architecture (Java's answer to ActiveX)
 - Digitally signed applets to extended security privileges without resorting to the "all or nothing" model of browser plug-ins or ActiveX

Java Versions, cont.

- **Java 2 Platform released in December 1998**
- **Standard Edition (JDK 1.2)**
 - Swing GUI components based on 100% Pure Java
 - Java 2D for professional, high-quality, two-dimensional graphics and imaging
 - The Collections Framework supporting advanced data structures like linked lists, trees, and sets
 - Audio enhancements to support .wav, .aiff, .au, .midi, and .rmf file formats
 - Printing of graphic objects
 - Java IDL API, which adds CORBA capability to Java

Java Versions, cont.

- **JDK 1.3 released in Spring of 2000**
 - Major Enhancements:
 - Java Naming and Directory Interface (JNDI)—a directory service for registering and looking up resources (objects)
 - RMI-IIOP—a protocol to communicate with distributed clients that are written in CORBA-compliant language
- **JDK 1.4 released in Spring 2002**
 - Major Enhancements
 - XML Processing
 - Logging API
 - Assertions
 - Next generation I/O library (java.nio)
 - SSL
 - JAAS – authentication and authorization API

Java 2 Platform, Enterprise Edition

- **Focused at e-commerce solutions**
 - Java Servlets and JavaServer Pages—Sun’s answer to Microsoft Active Server Pages and ColdFusion
 - Enterprise JavaBeans for bundling business logic in server-side components
 - JDBC data access for scrollable database queries (result sets)
 - JavaMail to send and receive mail with SMTP, POP3, or IMAP4 protocols
 - JAXP for parsing XML documents
 - Java Message Service for asynchronous communication between enterprise applications

Which Version Should You Use?

- **Applets**
 - Use JDK 1.1
 - Internet Explorer 4.0 and later and Netscape 4.06 through 4.72 support JDK 1.1. Netscape 6 and later support JDK 1.3.
 - Java Plug-In is required for later versions of Java
- **Applications**
 - For standard applications use JDK 1.4 (known as Java 2 SDK, Standard Edition, Version 1.4)
- **Best Approach**
 - Use JDK 1.4, but bookmark the JDK 1.1 API to check available methods when writing applets

Getting Started: Nuts and Bolts

1. Install Java

- JDK 1.4
 - <http://java.sun.com/j2se/1.4/>
- JDK 1.1
 - No longer supported by Sun
 - Compile to JDK 1.1 byte code using `-target` directive

2. Install a Java-Enabled Browser

- Netscape Navigator
 - <http://home.netscape.com/download/>
- Microsoft Internet Explorer
 - <http://www.microsoft.com/ie/download/>
- Sun's HotJava
 - <http://java.sun.com/products/hotjava/>

Getting Started: Nuts and Bolts, cont.

3. Bookmark or install the on-line Java API

- Java 2 SDK, Version 1.4 (JDK 1.4)
 - API Specification, <http://java.sun.com/j2se/1.4.2/docs/api/>
 - API Download, <http://java.sun.com/j2se/1.4.2/download.html#docs>
- Java 1.1(JDK 1.1)
 - API and Documentation, <http://java.sun.com/products/archive/jdk/1.1/index.html>

4. Create and run a Java program

- Create the file
- Compile it
- Run it

Getting Started: Details

1. Create the File

- Write and save a file (say `Test.java`) that defines public class `Test`
- File and class names are case sensitive and must match exactly

2. Compile the program

- Compile `Test.java` through

```
javac Test.java
```

- This step creates a file called `Test.class`
- If you get a “deprecation” warning, this means you are using a Java construct that has a newer alternative
- Use “`javac -deprecation Test.java`” for an explanation, then look the newer construct up in the on-line API

Getting Started: Details, cont.

3. Run the program

- For a stand-alone application, run it through

```
java Test
```

- Note that the command is `java`, not `javac`, and that you refer to `Test`, not `Test.class`
- For an applet that will run in a browser, run it by loading the WWW page that refers to it

Basic Hello World Application

- **“Application” is Java lingo for a stand-alone Java program**
 - Note that the **class name** and the **filename match**
 - A file can contain multiple classes, but **only one can be declared public**, and that one’s name must match the filename
- **File HelloWorld.java:**

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

Basic Hello World Application, cont.

- **Compiling:**
 - javac HelloWorld.java
- **Running:**
 - java HelloWorld
- **Output:**
 - Hello, world.

Command Line Arguments

- **Differences from C**

- In Java String is a real type
- Java arrays have an associated length
- The file name is not part of the command line arguments

- **File ShowArgs.java:**

```
public class ShowArgs {
    public static void main(String[] args) {
        for(int i=0; i<args.length; i++) {
            System.out.println("Arg " + i + " is " +
                args[i]);
        }
    }
}
```

Command Line Arguments, Results

- **Compiling and Running:**

```
> javac ShowArgs.java
> java ShowArgs fee fie foe fum
```

```
Arg 0 is fee
Arg 1 is fie
Arg 2 is foe
Arg 3 is fum
```

Basic Hello WWW Applet

- File HelloWWW.java:

```
import java.applet.Applet;
import java.awt.*;

public class HelloWWW extends Applet {
    public void init() {
        setBackground(Color.gray);
        setForeground(Color.white);
        setFont(new Font("SansSerif", Font.BOLD, 30));
    }

    public void paint(Graphics g) {
        g.drawString("Hello, World Wide Web.", 5, 35);
    }
}
```

Basic Hello WWW Applet, cont.

- File HelloWWW.html:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0
    Transitional//EN">
<HTML>
<HEAD>
    <TITLE>HelloWWW: Simple Applet Test.</TITLE>
</HEAD>

<BODY>
<H1>HelloWWW: Simple Applet Test.</H1>

<APPLET CODE="HelloWWW.class" WIDTH=400 HEIGHT=40>
    <B>Error! You must use a Java enabled browser.</B>
</APPLET>

</BODY>
</HTML>
```

Basic Hello WWW Applet, cont.

- **Compiling:**

```
javac -target 1.1 HelloWWW.java
```

- **Running:**

Load HelloWWW.html in a Java-enabled browser



Customizing Applets with PARAM

```
import java.applet.Applet;
import java.awt.*;

public class Message extends Applet {
    private int fontSize;
    private String message;

    public void init() {
        setBackground(Color.black);
        setForeground(Color.white);
        fontSize = getSize().height - 10;
        setFont(new Font("SansSerif", Font.BOLD, fontSize));
        // Read heading message from PARAM entry in HTML.
        message = getParameter("MESSAGE");
    }

    public void paint(Graphics g) {
        if (message != null)
            g.drawString(message, 5, fontSize+5);
    }
}
```

Customizing Applets with PARAM, cont.

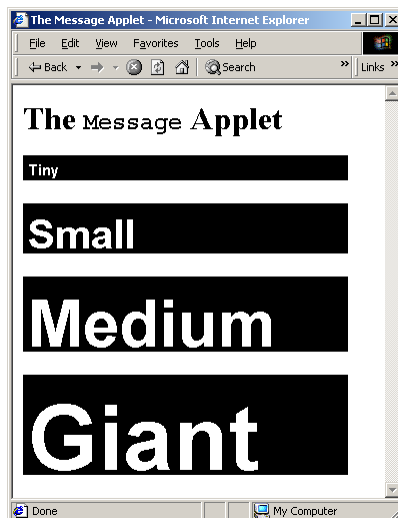
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<HEAD>
  <TITLE>The Message Applet</TITLE>
</HEAD>
<BODY BGCOLOR="WHITE">
<H1>The <CODE>Message</CODE> Applet</H1>
<P>
<APPLET CODE="Message.class" WIDTH=325 HEIGHT=25>
  <PARAM NAME="MESSAGE" VALUE="Tiny">
  <B>Sorry, these examples require Java</B>
</APPLET>
<P>
<APPLET CODE="Message.class" WIDTH=325 HEIGHT=50>
  <PARAM NAME="MESSAGE" VALUE="Small">
  <B>Sorry, these examples require Java</B>
</APPLET>
...
</BODY>
</HTML>
```

29

Introduction to Java

www.corewebprogramming.com

Customizing Applets with PARAM, Result



30

Introduction to Java

www.corewebprogramming.com

Summary

- Java is a complete language, supporting both standalone applications and Web development
- Java is compiled to bytecode and can be run on any platform that supports a Java Virtual Machine
- Java 2 Platform is bundled as a Standard Edition and Enterprise Edition
- Most browsers support only JDK 1.1
- Install Java Plug-In for later versions of Java



core
WEB
programming

Questions?