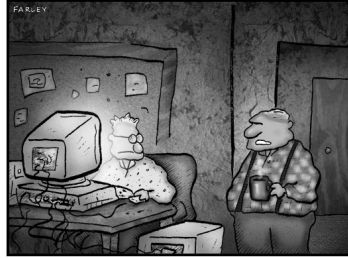


# Introduction to Java Programming

Introduction

## What is Java?

DOCTOR FUN



16 May 96  
Copyright © 1996 David F. Herby, a faculty member at Wake Forest University.  
http://www.wfu.edu/~dherby/doctorfun.html  
This cartoon is made available on the Internet for personal viewing only.  
Opinions expressed herein are solely those of the author.

## History of Java



- Originally developed for consumer electronics by Sun Microsystems
- Designed by James Gosling
- Official announced in 1995
- It's still under development
  - Version 1.0.2
  - Version 1.1.7
  - Version 1.2.8 (aka Java 2)
  - Version 1.3 (still Java 2 version 1.3)
  - Version 1.4.1 (Java 2)
  - Version 1.5 (Tiger - coming soon)

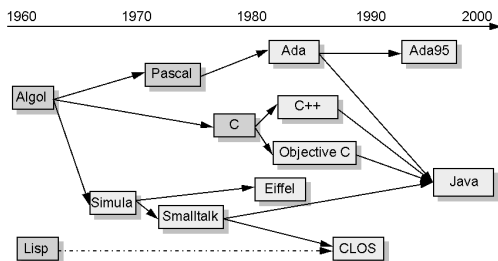


## Some History

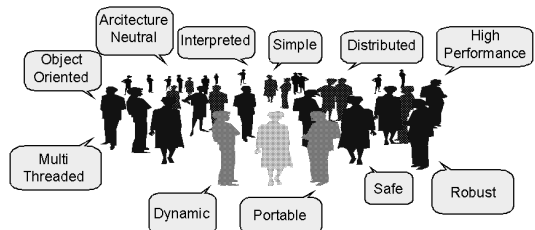
Designed originally for consumer electronics

- Project Green
- Oak
- Consumer Electronics
- Mosaic (Hot Java)
- Web
- 1995 First Alpha Release of Java

## History of Java



## Features of Java

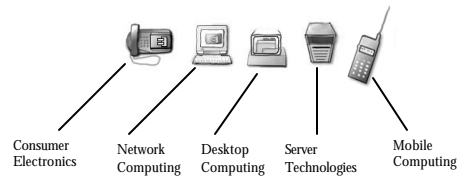


## Features of Java

### Java benefits

- **Simple:** remove complex, error-prone features from predecessor languages
- **Familiar:** base syntax and semantics on C and C++
- **Object Oriented:** provide object-orientation as the language foundation
- **Architecture Neutral:** compiles to a virtual machine
- **Portable:** standard data types
- **Robust:** memory management, strict typing
- **Interpreted:** provide basis for quick prototyping by eliminating static linking and reducing code dependencies which required recompilation in C++
- **Dynamic:** can add new pieces of an application and link into a running system!
- **Secure:** code loaded "over the network" can be checked to make sure it is safe
- **High Performance:** can do "just in time" compilation to speed up operation
- **Threaded:** allows the program to spawn threads of execution to work on separate tasks concurrently

## Java Runs Everywhere ??



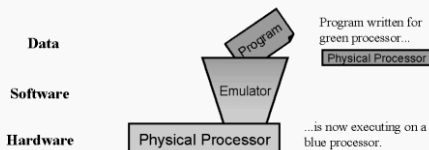
## Architecture: JVM

### Emulation

"Imitation."

The direct execution of machine language programs written for one processor by a different processor.

Emulation is accomplished using software, either microcode within a processor or a program.



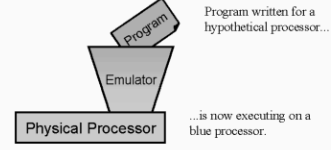
## Architecture: JVM

### Virtual machine

A hypothetical machine with all of the characteristics of a physical processor.

Implemented using machine emulation.

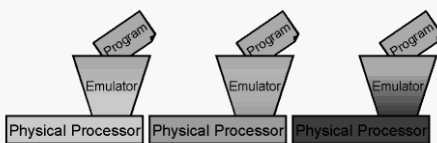
A hypothetical machine is a machine that is not realized in hardware.



## Architecture: JVM

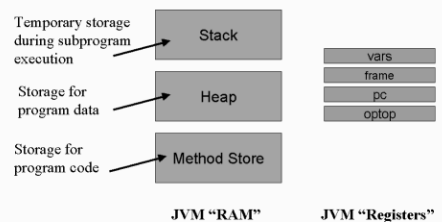
### Hardware platform independence

An advantage of the virtual machine concept is that the **same program** can be run directly on **different processors** provided an emulator exists! The program has hardware platform independence.



## Architecture: JVM

### Java Virtual Machine (JVM)

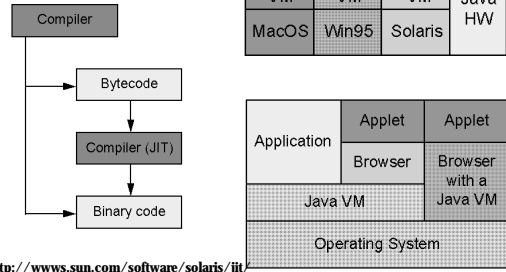


## Architecture: JVM

### Java programs

- Java programs are written in the Java language and contained in Java source code files with a **.java** extension.
- Java source code files are translated into JVM machine instructions called *bytecodes*. The files containing bytecodes are named using a **.class** extension.
- **.class** files contain additional information about the **interface** to the code within the file, not just the executable instructions!
- When a Java program is executed, the **.class** file is loaded into the JVM and executed.

## Java Model

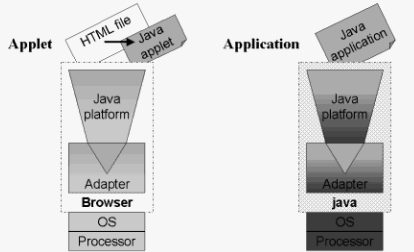


<http://www.sun.com/software/solaris/jit/>

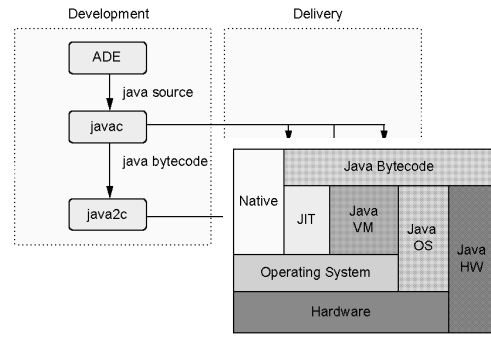
## Java Model

### Where do Java programs run?

- Java programs can run wherever a Java platform is implemented:

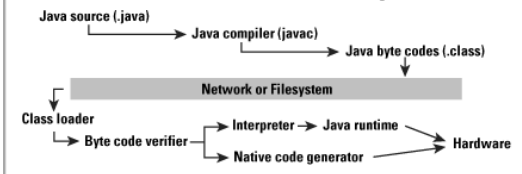


## Java Model



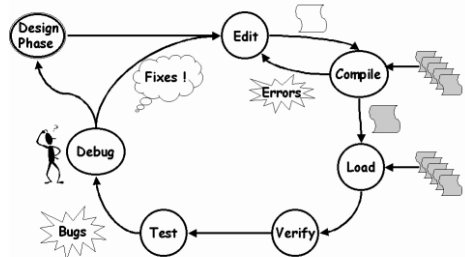
## Java Model

### How Java ensures security



## Coding cycle

### Implementing in Java



## Java program type

### Classifications of Java programs

- **Where do they run?**

- Applet*

- Runs inside a browser or in the appletviewer.
    - Requires an HTML file with an <applet> tag.

- Application*

- Runs as a standalone program on a Java platform.
    - Requires a "main" method as a starting point.

- **How do they interact with the user?**

- Graphical*

- Applets **always** provide a graphical user interface.
    - Applications **may** provide a graphical user interface.

- Text-based*

## Hello World (application)

```
/**
 * Simple "Hello World" text based application
 * @author Brad Rippe
 * @version 1.0
 */
import java.io.*;

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

## Watch out...

### Application alert!

- **Name of public class is the same as the name of the source file**

- public class HelloWorld (... is in HelloWorld.java file)

- **The JDK bin directory must be in the path**

- print your path using the path command under DOS and verify that it has the jdk1.2.2\bin directory

- **Applications must have public static void main( String[] args )**

- {...}
  - which is the starting point for the application

## Hello World (applet)

```
/** "Hello World" simple applet
 */
import java.applet.*;
import java.awt.Graphics;

public class HelloWorld extends Applet {
    public void paint (Graphics g){
        g.drawString("Hello World" , 25, 25);
    }
}
```

## HTML file for applet

```
<html>
<body>
<applet code="HelloWorld.class" width=275
        height=55>

</applet>
</body>
</html>
```

## Watch out...

### Applet alert!

- **Applet is always run through a browser or the appletviewer!**

- Cannot run an applet using java.
  - Always run the applet by *viewing* the HTML file.

- **Applet must always have an associated HTML file**

- HTML file must contain an <applet> tag where the code attribute contains the name of the .class file

- **Applets must extend the Applet class**

- public class HelloWorldApplet extends Applet {

- **Name of applet class is the same as the name of the source file**

- public class HelloWorldApplet extends Applet (... is in HelloWorld.java file)

## Tools Time!

- Download, install the compiler
- Creating Hello World application
- Creating Hello World applet

## Overtime

## Keywords & Identifiers

- Keywords are words that have specific meaning in Java (ex. `class`, `extends`, etc.)
- An identifier is a name given to a variable, class or method
  - Can contain a-z, A-Z, 0-9, `_`, `$`, but can not start with 0-9
  - Must not be a keyword
  - Case sensitive
  - Of any length
- All variables must be declared before use

## Data Types

- Primitive data type
  - `boolean` (1 bit) `true` or `false`
  - `byte` (8 bits) -128 .. +127
  - `char` (16 bits)
  - `short` (16 bits) -32,768 .. +32,767
  - `int` (32 bits) -2,147,483,648 .. +2,147,483,647
  - `long` (64 bits)
  - `float` (32 bits)     `double` (64 bits)
- Reference data type
  - Classes (will learn later)

## Arithmetic (Fig 1.9 - 1.11)

- Assignment:         `=`
- Addition:           `a + b`
- Subtraction:       `c - 10`
- Multiplication: `c * 0`
- Division:           `45 / 5`
- No exponent operator
- Modulus:           `8 % 3`

## Arithmetic Precedence

- Priority 1:           `()` - negation
- Priority 2:         `*`    `/`    `%`
- Priority 3:         `+`            `-`

## Conditional Statement: if()

- Syntax:
  - `if (data1 cond.operator data2) {`
    - `statement1;`
    - `statement2;`
    - ...
  - `}`

## Conditional Operators

- Equality `a == b`
- Inequality `a != b`
- Greater than `a > b`
- Less than `a < b`
- Greater than or equal `a >= b`
- Lesser than or equal `a <= b`
  - Notice: works with primitive data type

## Event Driven Programming (Just enough to do assignment 1)

- What is Event-Driven programming?
- Getting started with event-driven programming in Java:
  - Applet starts with `init()`, `start()` and `paint()` methods
  - `action()` trap users response to the object

## Additional Resources

- <http://developer.java.sun.com/developer/onlineTraining/new2java/programming/learn/>
- <http://java.sun.com>
- <http://www.javaworld.com>
- <http://java.sun.com/docs/books/tutorial/java/TOC.html>
- [brippe@fullcoll.edu](mailto:brippe@fullcoll.edu) (Subject CIS 226)