Platform Independence

Java Virtual Machine (JVM)
- Precisely defined
- Interpreted

APIs for all Operating System Services
- File I/O
- Network I/O
- User Interface
- Multiple Threads

Safety
- Byte Code Verifier
- SecurityManager

Hiding the Network

The World Wide Web hides the network

- Click on phrase rather than type network address
- Search by words or phrases using Alta Vista, Lycos, etc

Java Dynamic Class Loading

Java Distributed Object Model

- Remote Method Invocation
- Bind to object via URL, invoke methods via remote interface
- Method invocations may return other remote objects
- A computation involving a whole Web of distributed objects
  - invoking each other's methods

the network is the computer

Promoting Better Programming

Platform Independence eliminates porting ?
Platform Independence eliminates continuing maintenance ?

Eliminates errors that plague C and C++
- Pointer errors are syntactically impossible
- No memory leaks with garbage collection
- Array bounds checked on all array accesses

Exception Handling

Embedded Documentation
- Utility to generate Web of documents

Missing Pieces

Efficient Compilers
- Now 1/30 speed of C++
- Just In Time compilers
- Java Chips

Reliable Security
- Bugs in Byte Code Verifier implementations

3-D Graphics
- Java Media 3D (compatible with VRML 2.0 ?)

Imperfect Platform Independence
- Especially in java.awt for Graphical User Interface

Conclusion

Telephone, Television, Radio are Ubiquitous Networks
- Many Vendors of User Appliances and Communications Links
- But completely compatible
- Because the Market demands compatibility

Same will happen with the Internet
- Web and Java are steps in this direction

Web and Java are in the best long term interests of Earth science
- We are showing our faith by rewriting VisAD in Java

Book and URL references

The Java Home Page
- http://java.sun.com
  - Developer's Info
  - Documentation
    - Java Tutorial

The Java Announcements Archive
- http://javarc.org

Java in a Nutshell published by O'Reilly & Assoc.

These slides hanging off my home page as a postscript file
- SSEC Home Page / Staff List / Bill Hibbard / Java course notes

The "Hello World" Application

Create a Java Source File: HelloWorldApp.java

```
/**
* The HelloWorldApp class implements an application that
* simply displays "Hello World!" to the standard output.
*/
class HelloWorldApp {
  public static void main(String[] args) {
    //Display the string.
    System.out.println("Hello World!");
  }
}
```

Compile the Source File

UNIX, DOS, Windows:

```
javac HelloWorldApp.java
```

MacOS:

Drag HelloWorldApp.java file icon onto Java Compiler icon.

Creates HelloWorldApp.class or prints error message

Run the Application

UNIX, DOS, Windows:

java HelloWorldApp

Names are case sensitive

MacOS:

Double-click the HelloWorldApp.class file icon.

Should see "Hello World!" on standard output

The "Hello World" Applet

Create a Java Source File: HelloWorld.java

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

public class HelloWorld extends Applet {
  public void paint(Graphics g) {
    g.drawString("Hello world!", 50, 25);
  }
}
```

Compile the Source File; Creates HelloWorld.class

Create an HTML File: Hello.html
In the same dir with HelloWorld.class

```
<HTML>
  <HEAD>
  <TITLE> A Simple Program </TITLE>
  </HEAD>

  <BODY>
    <APPLET CODE="HelloWorld.class" WIDTH=150 HEIGHT=25>
    </APPLET>
  </BODY>
</HTML>
```

appletviewer Hello.html
netscape Hello.html

Java versus C

Java is like C in low level syntax ...

- variables, arrays, expressions, loops, if-else, case

But ...

- no pointers
- no functions
- no structs, no unions
- everything is an object (except primitive types like int)

Variables may be declared at first use - for example:

```
for (int i=0; i<n; i++) { ...
```

No .h files

no separate declarations and definitions

No preprocessor (#include, #define, #ifdef, etc)

Every public class in a separate source file (.java)

'import' statement tells compiler where to get class definitions
and allows shorthand like URL in place of java.net.URL

'package' statement combines source files (e.g., java.net)

Object-Oriented programming

Object
- member variables (data - like a struct)
- member methods (functions that operate on data)

Class
- declare member variables and methods
- class versus instance variables and methods

Inheritance
- subclasses inherit (or override) variables and methods of parent
- in Java, a class has exactly one parent

Class Hierarchy
- the tree of classes
- in Java, 'Object' class is root

```
public class Area extends Object {
  int line, elem, nlines, nelems, lres, eres;
  byte[][] data;

  Area(int l, int e, int nl, int ne, int lr, int er) {
    super(); // implicit
    if (nl < 1 || ne < 1 || lr < 1 || er < 1) {
      throw new AreaException("get a clue, buddy");
    }
    line = l; elem = e;
    nlines = nl; nelems = ne;
    lres = lr; eres = er;
    data = new byte[nlines][nelems];
  }

  Area(int nl, int ne) {
    Area(1, 1, nl, ne, 1, 1);
  }

  // methods to read and write data:
  //   getPixel, putPixel, getSector, etc
}
```

Invoking Methods

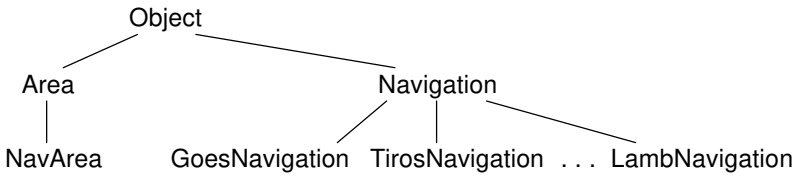
sometimes called "sending messages to objects"

```
Area vis_area = new Area(480, 640);

vis_area.putPixel(line, elem, pix);

byte pix = vis_area.getPixel(line, elem);

Area vis_area2 = vis_area.getSector(l, e, nl, ne, lr, er);
```

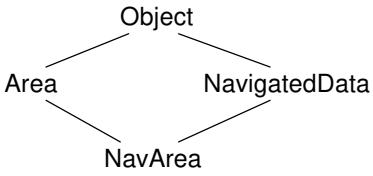
Inheritance

```
public class NavArea extends Area {
  Navigation navigation;
  Calibration calibration;

  NavArea(int l, int e, int nl, int ne, int lr, int er,
          Navigation nav, Calibration cal) {
    super(l, e, nl, ne, lr, er);
    navigation = nav;
    calibration = cal;
  }

  public byte getPixel(float lat, float lon) {
    int line = navigation.getLine(lat, lon)
    int elem = navigation.getElem(lat, lon)
    return getPixel(line, elem);
  }
}
```

Class Hierarchy



No Multiple Inheritance



In Java, this is not allowed. The NavArea class can have only one parent.

"McIDAS is the same 300 lines of code repeated 600 times"
- John Pyeatt

Object-Oriented programming is one way to fix this

```
public class Rectangle extends Object {
  int line, elem, nlines, nelems, lres, eres;

  Rectangle(int l, int e, int nl, int ne, int lr, int er) {
    line = l; elem = e;
    nlines = nl; nelems = ne;
    lres = lr; eres = er;
  }

  Rectangle subRectangle(int l, int e, int nl, int ne,
                         int lr, int er) {
    Rectangle result = CloneMetaData(l, e, nl, ne, lr, er);
    // resample from (line,elem, nlines, nelems, lres, eres)
    // to (l, e, nl, ne, lr, er) using calls to
    // this.getPixel  and  result.putPixel
  }
}
```

```
public class Area extends Rectangle {
  byte[][] data; // Area data
  // Area metadata

  Area(int l, int e, int nl, int ne, int lr, int er) {
    super(l, e, nl, ne, lr, er);
    data = new byte[nl][ne];
  }

  Area CloneMetaData(int l, int e, int nl, int ne,
                     int lr, int er) {
    Area result = new Area(l, e, nl, ne, lr, er);
    // copy area MetaData from 'this' to 'result'
    return result;
  }

  // access Area data - getPixel and putPixel
  // define and access Area metadata
}
```

```
public class Grid extends Rectangle {
  float[][] data; // Grid data
  // Grid metadata

  Grid(int l, int e, int nl, int ne, int lr, int er) {
    super(l, e, nl, ne, lr, er);
    data = new float[nl][ne];
  }

  Grid CloneMetaData(int l, int e, int nl, int ne,
                     int lr, int er) {
    Grid result = new Grid(l, e, nl, ne, lr, er);
    // copy grid MetaData from 'this' to 'result'
    return result;
  }

  // access grid data - getPixel and putPixel
  // define and access Grid metadata
}
```

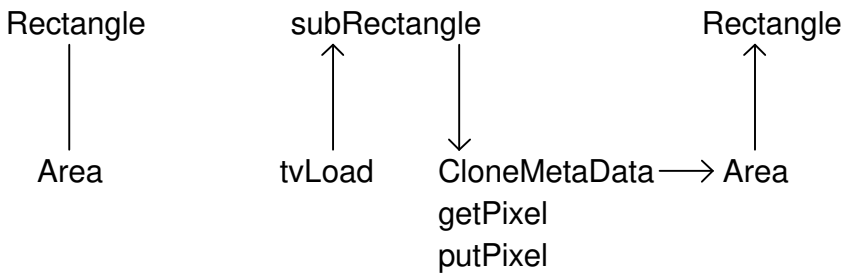
Code for subsectoring 2-D rectangles is repeated
all over McIDAS

Now just call subRectangle from any Area or Grid
subRectangle is a general method to make a subsector

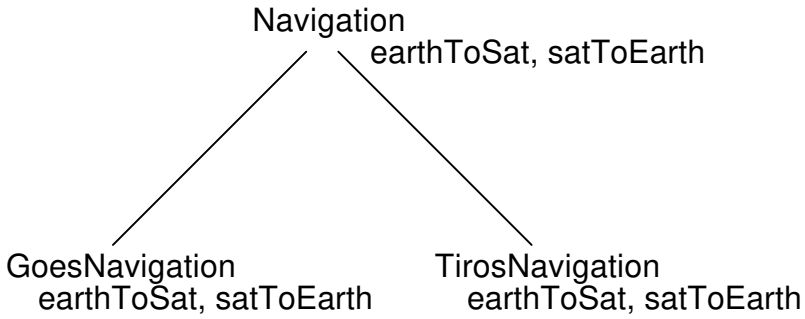
For example, to define an Area method for TV Load:

```
void tvLoad(Frame frame, int l, int e, int lr, int er) {
  Area tv_area = subRectangle(l, e, 480, 640, lr, er);
  // copy tv_area to frame
}
```

Method invocations up and down class tree



Subclasses can override methods of parents



Class Variables and Methods

```
public class UC extends Object {

  // 'static final' takes the place of #define in C
  static final int UCsize = 500;

  static int[] positiveUC = new int[UCsize];
  static int[] negativeUC = new int[UCsize];

  static {
    for (int i=0; i<UCsize; i++) {
      negativeUC[i] = +1;
      positiveUC[i] = -1;
    }
  }
  public static void putUC(int index; int value) {...}
  public static int getUC(int index) {...}
}

// never need objects of class UC, just do:
int j = UC.getUC(112);
```

Details: main, args, throws, length, I/O streams

```
class Count {
  public static void main(String[] args)
    throws java.io.IOException
  {
    int count = 0;
    int nargs = args.length;

    while (System.in.read() != -1)
      count++;
    System.out.println("Input has " + count + " chars.");
  }
}
```

[To run, type characters followed by CTRL/D]

Primitive Variable Types

byte	8-bit two's complement
short	16-bit two's complement
int	32-bit two's complement
long	64-bit two's complement
float	32-bit IEEE 754
double	64-bit IEEE 754
char	16-bit Unicode character
boolean	true or false

Reference Variable Types

Arrays
e.g., int[] data = new int[100];

Classes
e.g., Area vis_area = new Area(480, 640);

Interfaces
like classes, but only method signatures

No Pointers
Cannot cast references to integers
No pointer arithmetic
Cannot compute size of any object

All variables have default values

byte, short, int, long:	0
float, double:	0.0
boolean:	false
char:	\u0000
Reference types:	null

Can Initialize Variables on Declaration

```
int count = 0;
int[] table = {31, 28, 31, 30, 31, 30};
```

All variables are one of:

- member variable
- local variable
- method parameter
- exception handler parameter

primitive types are passed to methods by value
reference types are passed to methods by reference

```
// this is no good
public void earthToSat(int line, int elem,
                       float lat, float lon) { ...

// do this instead
public void earthToSat(ImageLoc line_elem,
                       EarthLoc lat_lon) { ...

public class EarthLoc {
  float lat, lon;
  // constructor and access methods
}

public class ImageLoc {
  int line, elem;
  // constructor and access methods
}
```

Arithmetic Operators

op1 + op2	Addition
op1 - op2	Subtraction
op1 * op2	Multiplication
op1 / op2	Division
op1 % op2	Remainder of op1 / op2
- op	Negation

The + also denotes string concatenation

```
System.out.println("Input has " + count + " chars.");
```

op ++	Increment op by 1; value before incrementing
++ op	Increment op by 1; value after incrementing
op --	Decrement op by 1; value before decrementing
-- op	Decrement op by 1; value after decrementing

Relational and Conditional Operators

op1 > op2	op1 is greater than op2
op1 >= op2	op1 is greater than or equal to op2
op1 < op2	op1 is less than to op2
op1 <= op2	op1 is less than or equal to op2
op1 == op2	op1 and op2 are equal
op1 != op2	op1 and op2 are not equal

object instanceof class

op1 && op2	op1 and op2 are both true
op1 ‖ op2	either op1 or op2 is true
! op	op is false

Bitwise Operators

op1 << op2	shift op1 left by op2
op1 >> op2	shift op1 right by op2 (extend sign)
op1 >>> op2	shift op1 right by op2 (0 fill)
op1 & op2	bitwise and
op1 | op2	bitwise or
op1 ^ op2	bitwise xor
~ op	bitwise complement

Assignment Operators

op1 += op2	op1 = op1 + op2
op1 -= op2	op1 = op1 - op2
op1 *= op2	op1 = op1 * op2
op1 /= op2	op1 = op1 / op2
op1 %= op2	op1 = op1 % op2
op1 &= op2	op1 = op1 & op2
op1 |= op2	op1 = op1 | op2
op1 ^= op2	op1 = op1 ^ op2
op1 <<= op2	op1 = op1 << op2
op1 >>= op2	op1 = op1 >> op2
op1 >>>= op2	op1 = op1 >>> op2

Operator Precedence in Java

postfix operators	[] . (params) expr++ expr--
unary operators	++expr --expr +expr -expr ~ !
creation or cast	new (type)expr
multiplicative	* / %
additive	+ -
shift	<< >> >>>
relational	< > <= >= instanceof
equality	== !=
bitwise AND	&
bitwise exclusive OR^	
bitwise inclusive OR	|
logical AND	&&
logical OR	||
conditional	? : (i.e. boolean_exp ? statement : statement)
assignment	= += -= *= /= %= ^= &= |= <<= >>= >>>=

Use parentheses if there is any doubt (by you or the next coder)

Control Flow Statements

if-else, switch-case	decision making
for, while, do-while	loop
try-catch-finally, throw	exception
break, continue, label:, return	miscellaneous

Although goto is a reserved word, currently the Java language does not support the goto statement. Use labeled breaks and continues instead.

If-else

```
int testscore;
char grade;

if (testscore >= 90) {
    grade = 'A';
} else if (testscore >= 80) {
    grade = 'B';
} else if (testscore >= 70) {
    grade = 'C';
} else if (testscore >= 60) {
    grade = 'D';
} else {
    grade = 'F';
}
```

Switch-case

```
int month;

switch (month) {
  case 1:  System.out.println("January"); break;
  case 2:  System.out.println("February"); break;
  case 3:  System.out.println("March"); break;
  case 4:  System.out.println("April"); break;
  case 5:  System.out.println("May"); break;
  case 6:  System.out.println("June"); break;
  case 7:  System.out.println("July"); break;
  case 8:  System.out.println("August"); break;
  case 9:  System.out.println("September"); break;
  case 10: System.out.println("October"); break;
  case 11: System.out.println("November"); break;
  case 12: System.out.println("December"); break;
  default: System.out.println("Invalid month"); break;
}
```

Loops

```
for (int i=0; i<n; i++) {
  outbuf[i] = inbuf[i];
}
```

```
while (System.in.read() != -1) {
  count++;
  System.out.println("Read character, Count = " + count);
}
```

```
int c;
InputStream in;
do {
  c = in.read();
} while (c != -1);
```

Exceptions

```
public class AreaException extends Exception {
  public AreaException() { super(); }
  public AreaException(String s) { super(s); }
}

. . .

Area ir_area;
try {
  ir_area = new Area(nl, ne);
}
catch (AreaException e) {
  ir_area = new Area(100, 100);
}
finally {
  // come here whatever happens
}
```

Return

```
public int countMissing() {
  int count;
  // compute count = # missing pixels
  return count;
}

public void psychiatrist() {
  // knows nothing, does nothing
  return;
}
```

Labelled break and continue

```
int c;
InputStream in;

line_loop: for (int line=0; line<nlines; line++) {
  for (int elem=0; elem<nelems; elem++) {
    c = in.read();
    if (c == END_OF_FILE) break line_loop;
    if (c == END_OF_LINE) continue line_loop;
    else data[line][elem] = c;
  }
}
```

Arrays

```
int[] arrayOfInts = new int[10]


String[] arrayOfStrings = new String[10];
for (int i=0; i<arrayOfStrings.length; i++) {
  arrayOfStrings[i] = new String("Hello " + i);
}


int[][] twodim = {{1, 2},
                  {3, 4, 5},
                  {6, 7, 8, 9}};
```

The String Class

"Input has " + count + " chars."

Every class has a toString method
    language does it for primitive types

Strings are immutable

String class includes equivalents to
    C string manipulation functions

Strings are objects, NOT arrays of chars

The StringBuffer Class

For Mutable Strings

```
class ReverseString {
  public static String reverseIt(String source) {
    int i, len = source.length();
    StringBuffer dest = new StringBuffer(len);

    for (i = (len - 1); i >= 0; i--) {
      dest.append(source.charAt(i));
    }
    return dest.toString();
  }
}
```

Abstract Classes and Abstract Methods

```
public abstract class Navigation extends Object {

  public Navigation() { super(); }
  public abstract int getLine(float lat, float lon);
  public abstract int getElem(float lat, float lon);
  public abstract float getLat(int line, int elem);
  public abstract float getLon(int line, int elem);

  public String toString() {
    return("none of your business");
  }
}
```

Abstract class may not be instantiated

i.e., cannot say "Navigation nav = new Navigation();"

Abstract methods must be overridden by non-abstract subclasses

Interfaces Like abstract classes - all methods must be abstract

```
/** This is an interface for any data object that
    approximates a function from R^n to R^m */
public interface RealFunction {
  /** evaluate function : R^n -> R^m */
  public abstract RealTuple evaluate(RealTuple value);
  /** invert function : R^m -> R^n */
  public abstract RealTuple invert(RealTuple value);
}

public class Rectangle extends Object
                       implements RealFunction {
  // R^2 -> R^1
}

public abstract class Navigation extends Object
                                 implements RealFunction {
  // R^2 -> R^2
}
```

Multiple inheritance OK for interfaces

Packages

```
package mcidas;

// define mcidas classes

public class Area extends Rectangle {
  // Area member variables and methods
}

// etc
```

CLASSPATH environment variable

If CLASSPATH = /home/classes
then mcidas classes in /home/classes/mcidas

Standard Java Packages

java.lang

Object, Class and Classloader

Boolean, Character, Double, Float, Integer, Long
wrappers for primtive types (Byte & Short in JDK 1.1)

String and StringBuffer

System, Runtime and Process

Thread and ThreadGroup, Runnable interface

Math (PI, E, abs, min, max, sqrt, sin, log, exp, etc)

Throwable, Exception, Error

java.io files, buffering, tokenizer

java.util Dictionary, Stack, Vector, Hashtable, Enumeration

java.net Socket, URL, InetAddress, etc

java.awt Button, Dialog, *Layout, Font, Color, TextField, etc
(AWT = Abstract Window Toolkit)

java.awt.image ColorModel, ImageConsumer, ImageProducer

java.awt.peer link to platform-dependent window system

java.applet Java program loadable by clicking on web link

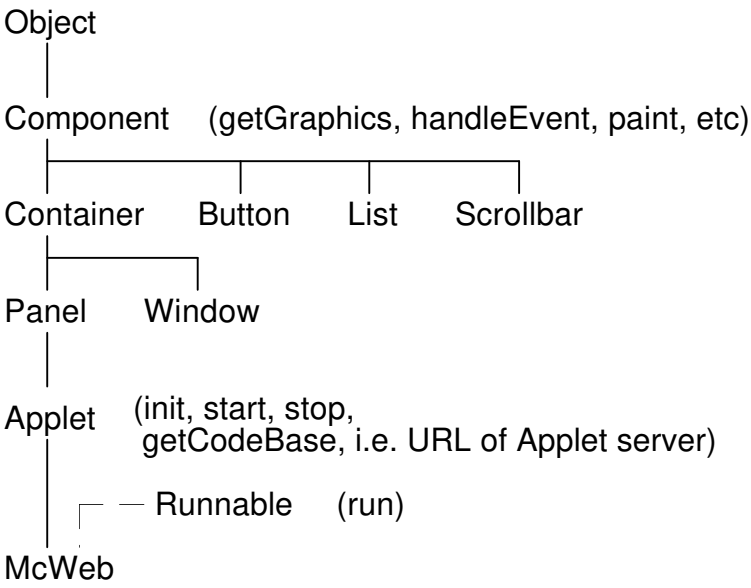
Threads

```
class TwoThreadsTest {
  public static void main (String[] args) {
    new SimpleThread("Jamaica").start();
    new SimpleThread("Fiji").start();
   }
}

class SimpleThread extends Thread {
  public SimpleThread(String str) {
    super(str);
  }

  public void run() {
    for (int i = 0; i < 10; i++) {
      System.out.println(i + " " + getName());
      try {
        sleep((int)(Math.random() * 1000));
      } catch (InterruptedException e) {}
    }
    System.out.println("DONE! " + getName());
  }
}
```

Synchronization

```
  // lock object (or array) during { ... }
  synchronized (object) { ... }

  // lock object during method
  public synchronized void clearArea() { ... }

  // lock class during method
  public static synchronized void getUC(int index) { ... }


public class EventHandler {
  public synchronized void run () {
    for (;;) {
      try this.wait(timout); catch (InterruptedException e);
      // process next event
    }
  }
}

  EventHandler evt;
  synchronized (evt) evt.notify();
```

Applets


Object
Component (getGraphics, handleEvent, paint, etc)
Container
Button
List
Scrollbar
Panel
Window
Applet (init, start, stop,
getCodeBase, i.e. URL of Applet server)
Runnable (run)
McWeb

Simplified Example of McWeb Applet

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

public class McWeb extends Applet implements Runnable {

  // McWeb applet data
  int      width, height; // applet's window size
  Button step, back, loop, exit;
  // etc
```

```
public void init()
{
  // get applet's window size
  width = size().width;
  height = size().height;

  // create GUI
  setLayout(new BorderLayout());
  Panel ledge = new Panel();
  ledge.setLayout(new FlowLayout());
  step = new Button("Step");
  ledge.add(step);
  back = new Button("Back");
  ledge.add(back);
  loop = new Button("Loop");
  ledge.add(loop);
  exit = new Button("Exit");
  ledge.add(exit);
  add("North", ledge);
  // etc
}
```

```
// handle user events
public boolean action(Event event, Object arg) {
  if (event.target == exit) {
    return true; // handle exit button click
  }
  else if (event.target == loop) {
    return true; // handle loop button click
  } // etc
  else return super.action(event, arg);
}

public boolean mouseMove(Event e, int x, int y)
{
  return true; // handle mouse move in image
}

public boolean mouseDown(Event e, int x, int y)
{
  return true; // handle mouse click in image
}
```

```
private Thread animator_thread = null;

public void start() {
  if (animator_thread == null) {
    // make an independent thread for this applet
    animator_thread = new Thread(this);
    animator_thread.start();
  }
}

public void stop() {
  if (animator_thread != null &&
      animator_thread.isAlive()) {
    animator_thread.stop();  // stop child if alive
  }
  // allow Thread object to be garbage collected
  animator_thread = null;
}
```

```
// invoked by call to animator_thread.start()
public void run()
{
  while (true) {
    // Component method - invokes update and paint
    repaint();

    // step animation, etc

    try {
      Thread.sleep(10); // in ms
    } catch (InterruptedException e) {
        break; // terminate while (and exit run)
    }
  }
}
```

```
  // NOTE paint and update logic to avoid flicker
  // see me or Java FAQ for messy details

  public void paint(Graphics g)
  {
    // draw some stuff
  }

  // override update to avoid flicker
  public final synchronized void update (Graphics g) {
    // flicker avoidance magic
  }

}
```

JDK (Java Developer's Kit)

appletviewer McWeb.html	run Applet
javac McWeb.java	compile
java visad.Main	run main() in class
javadoc visad	generate HTML docs
javah mcidas.McUC	generate .h and .c for native methods
javap visad.Field	disassembler
jdb visad.Main	debugger

Planned Java Enhancements

JDK 1.1

International, AWT, RMI, JDBC, etc

Enterprise APIs

JDBC (SQL), IDL (CORBA), RMI

Media APIs

2D, 3D, Animation, Telephony, Share, Framework

Security, Beans (components), Commerce (e$)

Just-In-Time compilers (JITs)

RMI (Remote Method Invocation)

Distributed Object Architecture

```
public interface AreaIFace implements Remote { ...
public class Area extends UnicastRemoteServer
                      implements AreaIFace { ...
```

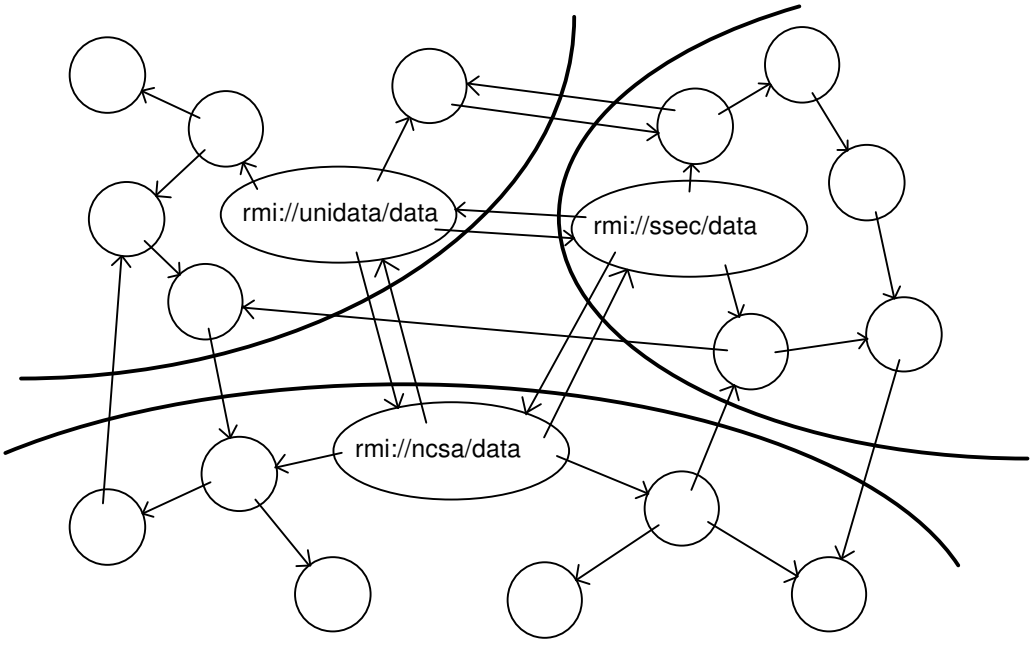
Server:

```
AreaIFace area = new Area();
java.rmi.Naming.bind(url, area)
```

Client:

```
AreaIFace area = (AreaIFace) java.rmi.Naming.lookup(url);
```

Object Serialization



The Network is the Computer

Collaborative User Interfaces

NCSA's Habanero and Java Media Share

```
public class McWeb extends Applet implements Wrapped { ...

  public void startInFrame(MirrorFrame f) { ...
  public void MarshallSelf(DataOutput out) {...
  public void UnmarshallSelf(DataInput in) {...

  public boolean handleEvent(Event e) {
    if (e.id == Event.ACTION_EVENT) return false;
    // handle other types of e.id
    Event ac = new Event(this, Event.ACTION_EVENT,
                    new McWebState(state1, ..., stateN));
    postEvent(ac);
  }

  // etc, etc, etc - a bit complex
```

See http://www.ncsa.uiuc.edu/SDG/Software/Habanero/

The Java Generic Library

http://www.objectspace.com/jgl

Containers
- Sequences, Maps, Sets, Queues, Stacks, etc

Algorithms
- searching, sorting, filtering, etc

Functions and Predicates

Iterators

Rewrite VisAD as a Java Class Library

Make portable across hardware platforms & operating systems

Was SGI only

Support for scientific programs written in Java

Was VisAD-unique language

Support a variety of discipline-specific User Interfaces

Was generic

Exploit Java's and Web's easy access to network information

Was remote procedure call to C or Fortran; was difficult

Data Model

Philosophy: data objects approximate mathematical objects

Objects partitioned by type

Mathematical type → Data type

- Real number → floating point, double, rational interval
- Vector → tuple
- Set → polypoint, polyline, polytriangle, etc
- Function → field (i.e., a finite sampling)

Metadata about how data approximate mathematical objects

- Missing data indicators
- Numerical precision (number of bits, lower & upper bounds)
- Sampling information (satellite navigation & calibration)

Data Examples

Image = 2-D field
    domain = samples of (latitude, longitude)
    range = (band1, band2, ..., bandN)

Time sequence of images = 1-D field
    domain = samples of time
    range = Image (i.e., the 2-D field defined above)

Set of observations = 1-D field
    domain = integers {1, 2, ..., M}
    range = (time, lat, lon, alt, temp, pres, ...)

Computational Semantics of Data

Real arithmetic →

- IEEE floating point standards
- Interval arithmetic
- Missing + 1.0 = Missing, etc
- X / 0.0 = Missing

Function operations →

- Implicit field resampling
- Implicit missing outside range of field domain
- Interpolation
- Numerical integration & differentiation

Interpret some operations with type mismatches

- field + float is interpreted as function + constant

Display Model

Display primitives

pixel or voxel coordinates (x, y, z), animation index

color components, transparency, etc

Mappings from data primitives to display primitives

for example, display time sequence of images by:

(lat, lon) → (x, y)

ir_radiance → color

time → animation index

Extensible set of display primitives and rendering algorithms

Direct manipulation (users change data via their depictions)

Class Library

Abstract classes for mathematical objects
- real number, vector, function
- abstract methods for mathematical operations

Extend to subclasses for various forms of approximation
- implement methods using approximation algorithms

Procedural interface to data hides storage details

Primitive types: real, text string
- user-defined names and units

Tuple: (type1, type2, ..., typeN)
Field: domain type → range type
- Sampling classes (of n-dimensional real spaces)

Primitive display type class

Rendering algorithm classes

Display class
- links to data objects (class)
- mappings from data primitive to display primitives (class)
  - include coordinate transforms (e.g., image on sphere)
- runnable

Synchronized access to data objects

Extensible
- data (e.g., spherical harmonics, metadata)
- sampling (e.g., satellite navigation)
- display primitives & rendering algorithms

Distributed Objects

Java Distributed Object Model

Java Remote Method Invocation system
- interfaces to remote objects
- bind via URLs
- methods may return remote objects

Distributed computations
- compare data objects scattered around network

Link display objects to remote data objects

User Interface

Support for multiple discipline-specific user interfaces
rather than a generic user interface

Vis5D (specific for 3-D atmosphere & ocean models)
new version based on Java / VisAD (long term plan)

Satellite image analysis and visualization

SpreadSheet paradigm
display object per cell

Java programming language for analysis

Virtual Reality (e.g., CAVE)

Collaboration

Use NCSA Habanero or Java Media Share

User interface replicted at multiple displays
    collect user interface events

Conclusion

Platform-independence

Flexible and extensible data and display models

Java programming support for scientists' algorithms

Support for multiple discipline-specific user interfaces

Exploit Internet via
- distributed computing
- collaborative user interfaces