Media Computation Workshop Day 2

Mark Guzdial
College of Computing
Georgia Institute of Technology
guzdial@cc.gatech.edu
http://www.cc.gatech.edu/~mark.guzdial
http://www.georgiacomputes.org

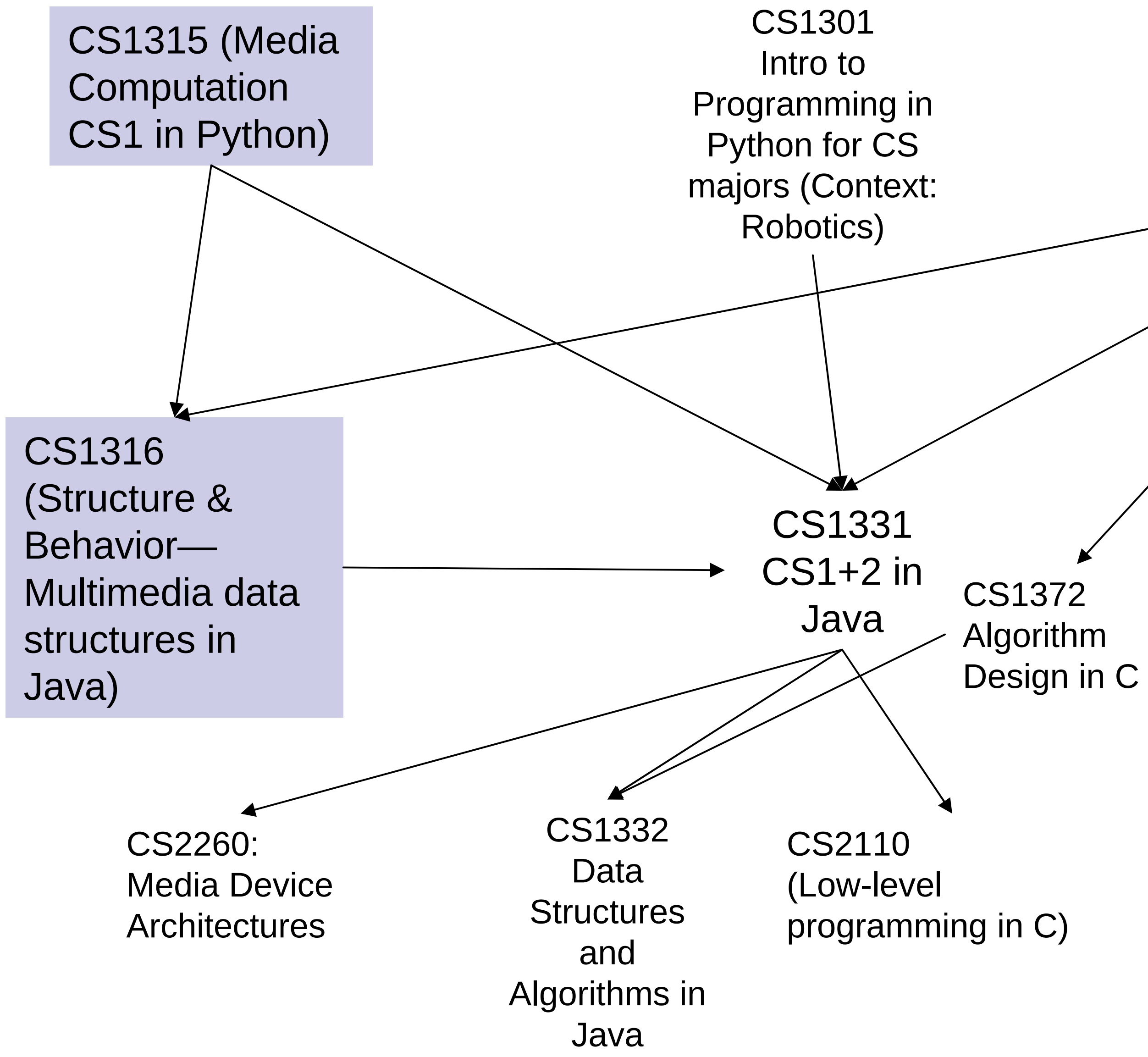
Workshop Plan-Day 2

- 9-9:45 am: Overview of results of Media Computation.
  - Why a contextualized computing education approach
  - Support available for teachers for adopting, adapting, and assessing.
  - 10:00-10:15: Break
- 10:15-12:00: Pictures and sounds in Java: Overview
- 12:00-1:00: Lunch
- 1:00-2:30: Movies in Media Computation
  - 2:30-2:45: Break
- 2:45-3:15: Discussion. *How might you use these kinds of assignments in your classes?*
- 3:15-4:30: Tackling a homework assignment in Media Computation. *Making a movie*.

Why a Contextualized Approach

- What Georgia Tech is Teaching
- What our research results show

What Georgia Tech Teaches


CS1315 (Media Computation CS1 in Python)
CS1301 Intro to Programming in Python for CS majors (Context: Robotics)
CS1371 (Computing for Engineering in MATLAB (*only*))
CS1316 (Structure & Behavior—Multimedia data structures in Java)
CS1331 CS1+2 in Java
CS1372 Algorithm Design in C
CS2260: Media Device Architectures
CS1332 Data Structures and Algorithms in Java
CS2110 (Low-level programming in C)

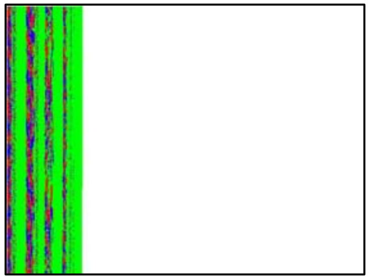
Institute for Computing Education (ICE@GT) Summer Workshops for High School Teachers: Media Computation CS1 in Java

A Context for CS1 for CS majors: Robotics

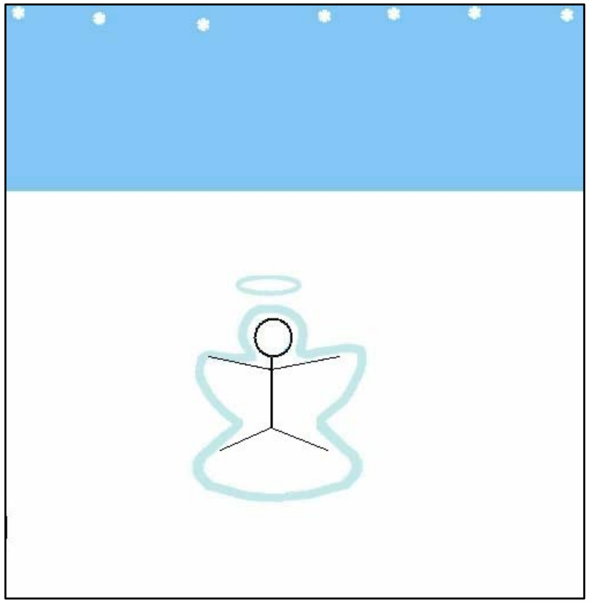
- Microsoft Research has funded the Institute for Personal Robotics in Education
  - Tucker Balch, Directing
    Monica Sweat, Developing GT's CS1
  - Joint between Bryn Mawr and Georgia Tech
  - http://www.roboteducation.org
- Goal is to develop a CS1 and CS2 with robotics as the context.
  - HW2: Recursively follow a light
  - CS2: Add a camera

Media Computation: Teaching in a Relevant Context

- Presenting CS topics with media projects and examples
  - Iteration as creating negative and grayscale images
  - Indexing in a range as removing redeye
  - Algorithms for blending both images and sounds
  - Linked lists as song fragments woven to make music
  - Information encodings as sound visualizations



Examples of Student Work



Soup-
Audio
Collage

Canon-
LinkedList of
(MIDI) Music

Use in Undergraduate, High School, and Teacher Workshops

- Introductory media computing in Python
  - For both non-majors and CS majors
- Introduction to object-oriented programming in Java
- Introduction to data structures in Java
  - Used at Georgia Tech, Linfield College, and Gainesville College
- Introduction to programming and CS AP for high school teachers

Schools Using Media Computation

- ***Java***
- University of Mass. at Boston
- Denison University
- Duke University
- Northwestern College, Iowa
- University of Wisconsin, Oshkosh
- University of Northern Iowa:
- St. Thomas University, Florida
- New Mexico Highlands University
- Norfolk State University
- Hickory High School APCS
- John Carroll University)
- San Jose State University
- Public high schools in DeKalb County, Fulton County, and Columbus County in Georgia
- West Virginia Institute of Technology:
- Rollins College
- University of Western Ontario:
- Columbia College:
- SUNY-Albany

- ***Python***
- Gainesville College (North Georgia)
- Univ. of Calif. Santa Cruz
- UIC (Univ. of Illinois at Chicago)
- Australian National University
- Plymouth State University
- Rivier College
- Kalamazoo College
- Union College,
- Ryerson University, Toronto
- University of Texas at El Paso
- University of California at Riverside
- University of California, Santa Clara
- Blue Ridge Community College
- University of San Francisco
- Muhlenberg College
- University of Alaska, Anchorage
- Southern Catholic College
- Clemson University
- Brown University
- Centre College:
- Ohio State
- Wilfrid Laurier University, Waterloo Ontario, Canada
- Smith College Virginia Tech /
- Kennesaw State University
- Brock University, St. Catharines, Ontario, Canada
- University of Aarhus

Support for Teachers

- http://coweb.cc.gatech.edu/mediacomp-teach
- Mediacomp-teach@cc.gatech.edu mailing list

Results

- Average CS1 success pre-MediaComp:
  Average 72.8%
  In MediaComp (51% female):
  Average 84% (as high as 90%)
  - Similar results at Gainesville, U.Ill-Chicago, ANU
  - Specific majors more dramatic:
    Management majors's success rate 49% => 88%
- Students are excited, and becoming CS majors, CS minors, CS teachers, and Computational Media majors (over 100, 25% female)
- Students from MediaComp Data structures continue into traditional CS and have equivalent success rates.

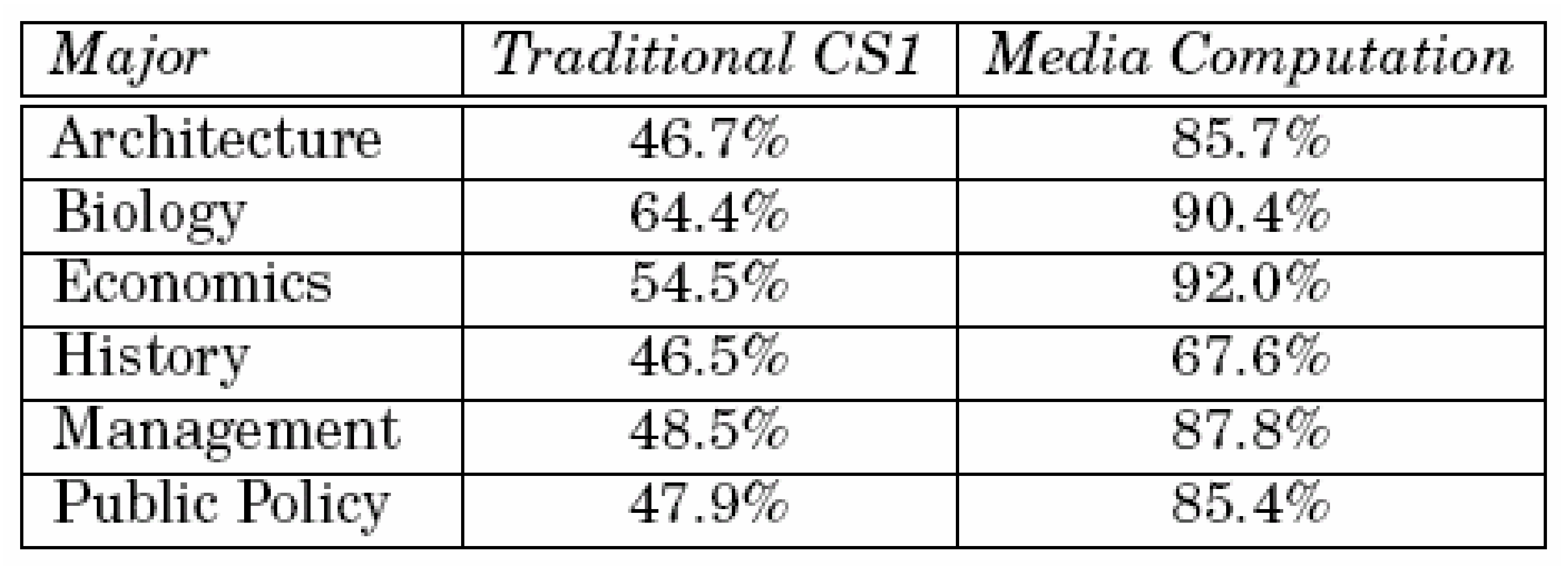
Student voices

- Intro CS student (female): “I just wish I had more time to play around with that and make neat effects. But JES [IDE for class] will be on my computer forever, so… that’s the nice thing about this class is that you could go as deep into the homework as you wanted. So, I’d turn it in and then me and my roommate would do more after to see what we could do with it.”
- High School teacher: “This was the best (non-college credit) workshop I have ever taken.”
- Students in multimedia data structures: “Data structures is an important step. Use of media! It makes it fun.”

What works, Where it doesn’t

- What works
  - Open-ended assignments to allow for student creativity and expression
    - Especially with their *own* media
  - Collaborative space for students to publicly share their media artifacts with others.
  - Transfer from Python =>Java
- Where it doesn’t
  - MediaComp isn’t “just the slow path”
  - Where approaches or languages are mandated

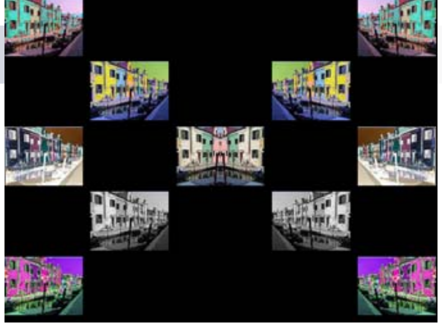
Success Rates for Specific Majors



Major	*Traditional CS1*	*Media Computation*
Architecture	46.7%	85.7%
Biology	64.4%	90.4%
Economics	54.5%	92.0%
History	46.5%	67.6%
Management	48.5%	87.8%
Public Policy	47.9%	85.4%

Success rates in traditional CS1 for students in various majors average Fall ’99 to Fall ’02, compared to Spring ’03 to Fall ’05 in Media Computation.

Example Collage Code



```
def hw3():
  venice1=makePicture(getMediaPath("venice.jpg"))
  print venice1
  venice2=makePicture(getMediaPath("venice.jpg"))
  print venice2
  venice3=makePicture(getMediaPath("venice.jpg"))
  print venice3
  venice4=makePicture(getMediaPath("venice.jpg"))
  print venice4
  venice5=makePicture(getMediaPath("venice.jpg"))
  print venice5
  venice6=makePicture(getMediaPath("venice.jpg"))
  print venice6
  canvas=makeEmptyPicture(640,480)
  print canvas
  IncreaseRed(venice1)
  targetX=1
  for sourceX in range(1,getWidth(venice1),3):
    targetY=1
    for sourceY in range(1, getHeight(venice1),3):
      px=getPixel (venice1, sourceX, sourceY)
      cx=getPixel (canvas, targetX, targetY)
      setColor(cx,getColor(px))
      targetY=targetY +1
    targetX=targetX +1
  IncreaseBlue(venice2)
  targetX=128
  for sourceX in range(1, getWidth(venice2),3):
    targetY=96
    for sourceY in range(1, getHeight(venice2),3):
      px=getPixel (venice2, sourceX, sourceY)
      cx=getPixel (canvas, targetX, targetY)
      setColor(cx,getColor(px))
      targetY=targetY +1
    targetX=targetX +1
```

```
negative(venice3)
  targetX=1
  for sourceX in range(1, getWidth(venice3),3):
    targetY=192
    for sourceY in range(1, getHeight(venice3),3):
      px=getPixel (venice3, sourceX, sourceY)
      cx=getPixel (canvas, targetX, targetY)
      setColor(cx,getColor(px))
      targetY=targetY +1
    targetX=targetX +1
  greyScaleNew(venice4)
  targetX=128
  for sourceX in range(1, getWidth(venice4),3):
    targetY=288
    for sourceY in range(1, getHeight(venice4),3):
      px=getPixel (venice4, sourceX, sourceY)
      cx=getPixel (canvas, targetX, targetY)
      setColor(cx,getColor(px))
      targetY=targetY +1
    targetX=targetX +1
  IncreaseGreen(venice5)
  targetX=1
  for sourceX in range(1, getWidth(venice5),3):
    targetY=384
    for sourceY in range(1, getHeight(venice5),3):
      px=getPixel (venice5, sourceX, sourceY)
      cx=getPixel (canvas, targetX, targetY)
      setColor(cx,getColor(px))
      targetY=targetY +1
    targetX=targetX +1
```

```
targetX=256
  for sourceX in range(1, getWidth(venice6),3):
    targetY=192
    for sourceY in range(1, getHeight(venice6),3):
      px=getPixel (venice6, sourceX, sourceY)
      cx=getPixel (canvas, targetX, targetY)
      setColor(cx,getColor(px))
      targetY=targetY +1
    targetX=targetX +1

  mirrorVertical(canvas)
  show(canvas)
  return(canvas)

def IncreaseRed (venice1):
  for pixel in getPixels(venice1):
     myred = getRed(pixel)
     setRed (pixel, myred * 1.5)

def IncreaseBlue(venice2):
  for pixel in getPixels(venice2):
    myblue = getBlue(pixel)
    setBlue (pixel, myblue * 1.5)

def IncreaseGreen(venice5):
  for pixel in getPixels(venice5):
    mygreen = getGreen(pixel)
    setGreen (pixel, mygreen * 1.5)

def greyScaleNew(venice4):
  for px in getPixels(venice4):
    newRed = getRed(px) * 0.299
    newGreen = getGreen(px) * 0.587
    newBlue = getBlue(px) * 0.114
    luminance = newRed+newGreen+newBlue
    setColor(px,makeColor(luminance,luminance,luminance))

…
```

Follow-up Survey: *Did it have a lasting impact?*

- In Spring 2004, conducted an email survey with students from Spring 2003 (*n*=120) and Fall 2003 (*n*=303) students.
- 59 responses
  - 11 (19%) had written a Python program on their own since the class had ended.
  - 27% had edited media that they hadn’t previously.

“Did the class change how you interact with computers?”

- 20% said no.
- 80% said yes, but it was also more about changing how they *thought* about computers.
  - “Definitely makes me think of what is going on behind the scenes of such programs like Photoshop and Illustrator.”
  - 'I understand technological concepts more easily now; I am more willing and able to experience new things with computers now’
  - 'I have learned more about the big picture behind computer science and programming. This has helped me to figure out how to use programs that I've never used before, troubleshoot problems on my own computer, use programs that I was already familiar with in a more sophisticated way, and given me more confidence to try to problem solve, explore, and fix my computer.’

Latest Findings: MediaComp CS2

- Is context still useful in a second course?
  - Work with Lana Yarosh
- 11% agreed with “Working with media is a waste of time that could be used to learn the material in greater depth.”
- A majority of the class (70%) agreed or strongly agreed that working with media makes the class more interesting.
- 67% of the students agreed or strongly agreed that they were really excited by at least one class project and 66% reported doing extra work on projects to make the outcome look “cool.”
- Critical finding: Students saw the narrative—more on that tomorrow.

New Degree: BS in Computational Media

- Joint with School of Literature, Communications, and Culture
- Requirements:
  - All the same General Education as CS, including Calculus, Discrete Math, Statistics.
  - ½ of required courses in Computer Science.
    - *Same* courses as our CS majors.
  - ½ in Liberal Arts
    - Performance art, film studies, media theory, etc.
- Results
  - 58 majors in first year, 24% female.
  - Nearly 200 majors today, still about ¼ female.
  - Fall 2006 class 70% larger than Fall 2005, Predicting *100%* increase in Fall 2007!

Threads™

- Carrying Context to the Degree Level
- A new conceptualization for an undergraduate degree.
- We defined 8 “Threads” which together define Computing:
  - Computing and Media, People, Platforms, Computational Modeling, Information Internetworking, Intelligence, Embodiment, and Foundations
  - The courses within each are “core” CS (some), advanced/new CS, and some non-CS (e.g., Psychology in People Thread)

Threads™ as a Degree

- Our BS in Computer Science is defined as *any two Threads*.
  - 28 possible paths to a degree now.
- Advantages:
  - Clearly says what CS is. It *isn't* just programming.
  - Clear differentiator for students.
    - It's not just India's or China's CS
  - Explains *why* students take any particular course

Bill Gates in TIME 12 Feb 2007

- Q: Education is a big focus for you. So, is there better learning through technology?

- A: It's important to be humble when we talk about education, because TV was going to change education and videotape was going to change it and computer-aided instruction was going to change it. But until the Internet exploded 10 years ago, technology really hadn't made a dent in education at all. ***Learning is mostly about creating a context for motivation. It's about why should you learn things.*** Technology plays a role, but it's not a panacea.

Generalizing the Solution

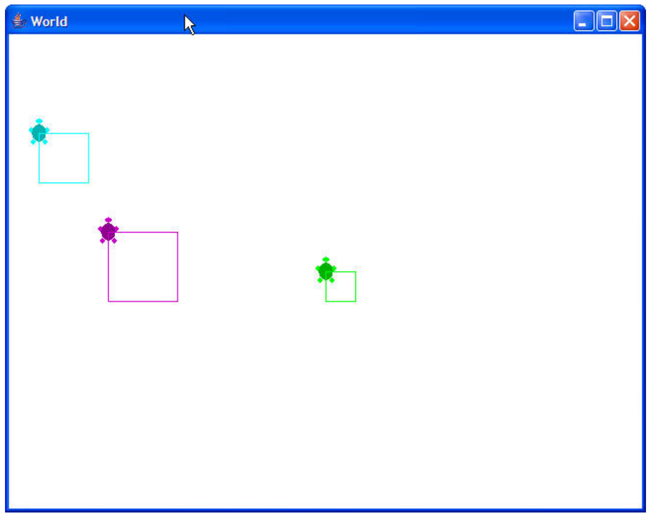
- These are Georgia Tech’s solutions.
  They are instantiations of a general solution.
- *Contextualize and specialize*
  - We can’t compete globally in terms of commodity skills.
    - And our students find those jobs boring, anyway.
  - We have to show:
    - Alternative options
    - Ways to specialize
    - Context to motivate and demonstrate broader outcomes.

Funding Sources

- National Science Foundation
- Microsoft Research
- Georgia Tech's College of Computing
- Georgia’s Department of Education
- GVU Center,
- Al West Fund
- President's Undergraduate Research Award
- Toyota Foundation

Java Media Computation

- We use DrJava
  - Interactive Java programming environment
- Syllabus
  - Introducing objects in a media computation context
- Image manipulation
- Sound manipulation
- Movie manipulation



Where to get DrJava

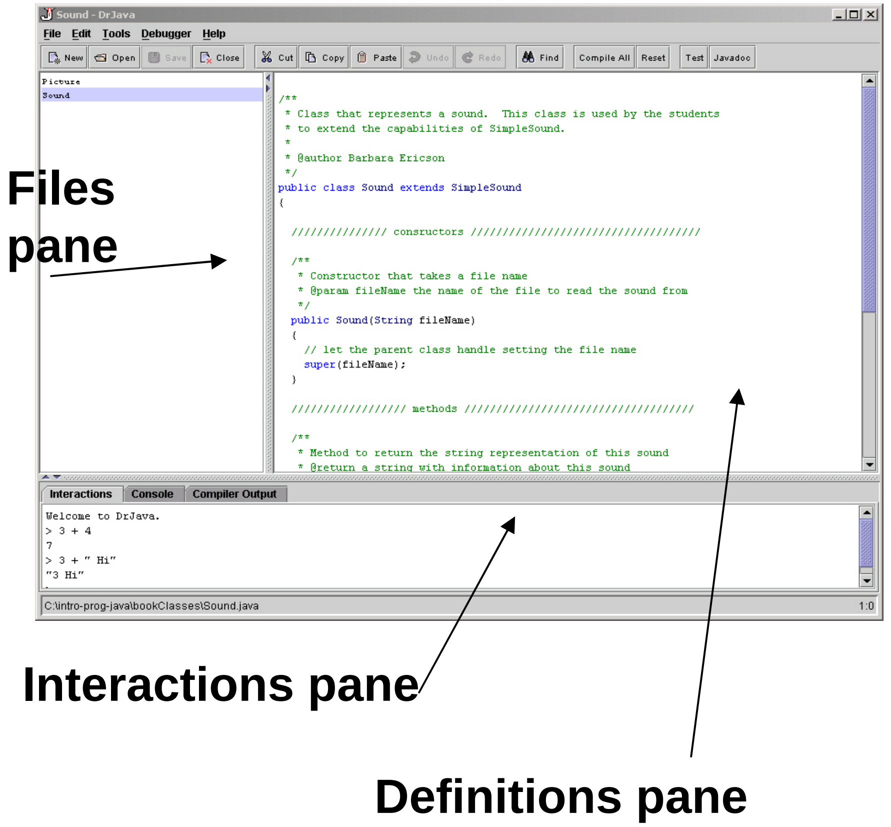
- DrJava is a free development environment for Java aimed at students
  - From Rice University
- It can be downloaded from
  - http://www.drjava.org/
- It requires Java 1,3, 1.4, or 1.5
  - It recommends using 1.4.2
  - Download Java first from java.sun.com

How to Start DrJava

- Click on the DrJava icon on your desktop
- Wait while DrJava loads
  - It sill display the splash screen while loading
- Once you see the full environment
  - you may begin

DrJava Features

- Works with multiple files
  - Files pane
- Color coded editor
  - Definitions pane
- Interpretation of Java code
  - Interactions pane
- Integrated debugger



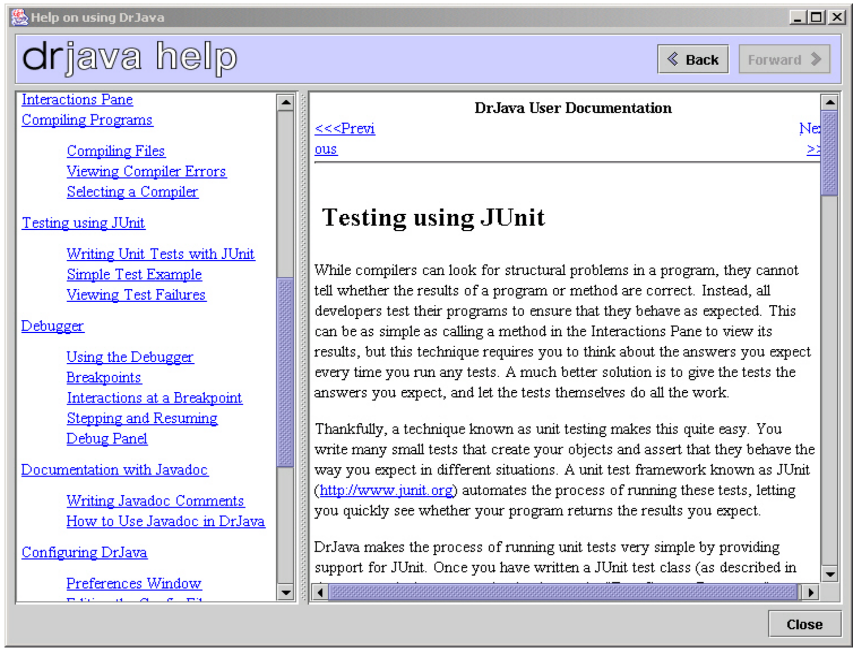
```
/**
 * Class that represents a sound.  This class is used by the students
 * to extend the capabilities of SimpleSound.
 *
 * @author Barbara Ericson
 */
public class Sound extends SimpleSound
{

  /////////////// consructors ////////////////////////////////////

  /**
   * Constructor that takes a file name
   * @param fileName the name of the file to read the sound from
   */
  public Sound(String fileName)
  {
    // let the parent class handle setting the file name
    super(fileName);
  }

  ///////////////// methods ///////////////////////////////////

  /**
   * Method to return the string representation of this sound
   * @return a string with information about this sound
```

```
Welcome to DrJava.
> 3 + 4
7
> 3 + " Hi"
"3 Hi"
```

C:\intro-prog-java\bookClasses\Sound.java 1:0

Help Window

- Click on Help in the menu and then again on Help to bring up the help window
  - Or use F1
- The table of contents is on the left
  - Click on a topic to see the help on the right



DrJava User Documentation

<<<Previ ous Ne >>

Testing using JUnit

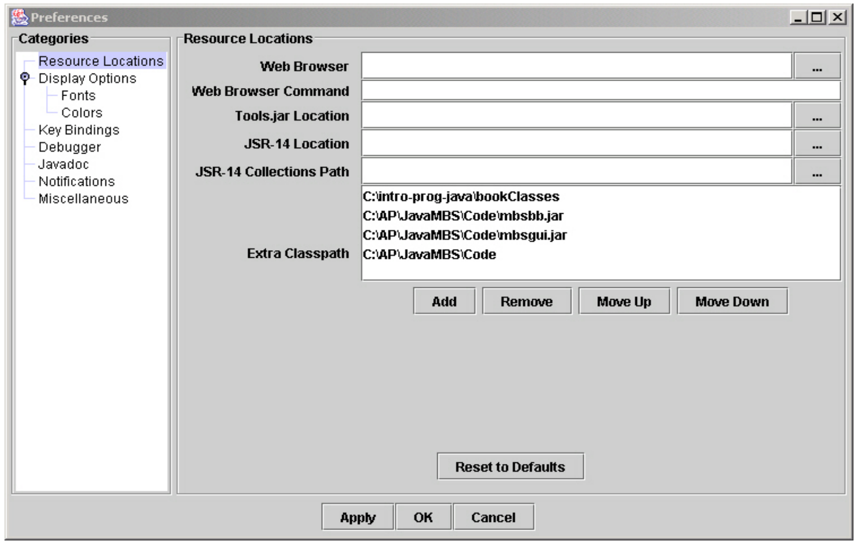
While compilers can look for structural problems in a program, they cannot tell whether the results of a program or method are correct. Instead, all developers test their programs to ensure that they behave as expected. This can be as simple as calling a method in the Interactions Pane to view its results, but this technique requires you to think about the answers you expect every time you run any tests. A much better solution is to give the tests the answers you expect, and let the tests themselves do all the work.

Thankfully, a technique known as unit testing makes this quite easy. You write many small tests that create your objects and assert that they behave the way you expect in different situations. A unit test framework known as JUnit (http://www.junit.org) automates the process of running these tests, letting you quickly see whether your program returns the results you expect.

DrJava makes the process of running unit tests very simple by providing support for JUnit. Once you have written a JUnit test class (as described in

How to add to the classpath

- Bring up the preferences window
  - Click on Edit and then Preferences
  - Click the add button
    - And select all jar files that you want to add to the classpath
    - Also add directories that have classes that you wish to add to the classpath
    - When you are done click on “Ok”



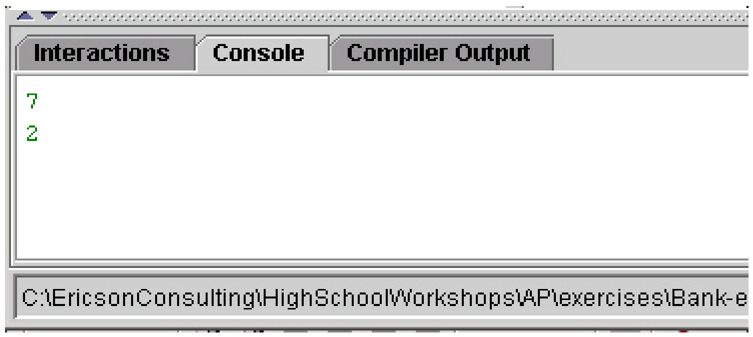
Extra Classpath
C:\intro-prog-java\bookClasses
C:\AP\JavaMBS\Code\mbsbb.jar
C:\AP\JavaMBS\Code\mbsgui.jar
C:\AP\JavaMBS\Code

How to use the interactions pane

- If you don't end a statement with a semicolon ';' it will be interpreted and the result printed
- If you end a statement with a semicolon it will be interpreted but the result won't be printed
  - Use System.out.println(expression) to print
- To enter multiple line statements use Shift + Enter

How to use the console pane

- If you click on the console tab
  - You will see the console pane
- It shows just the items that you have printed to the console
  - Using System.out.println or
  - System.out.print

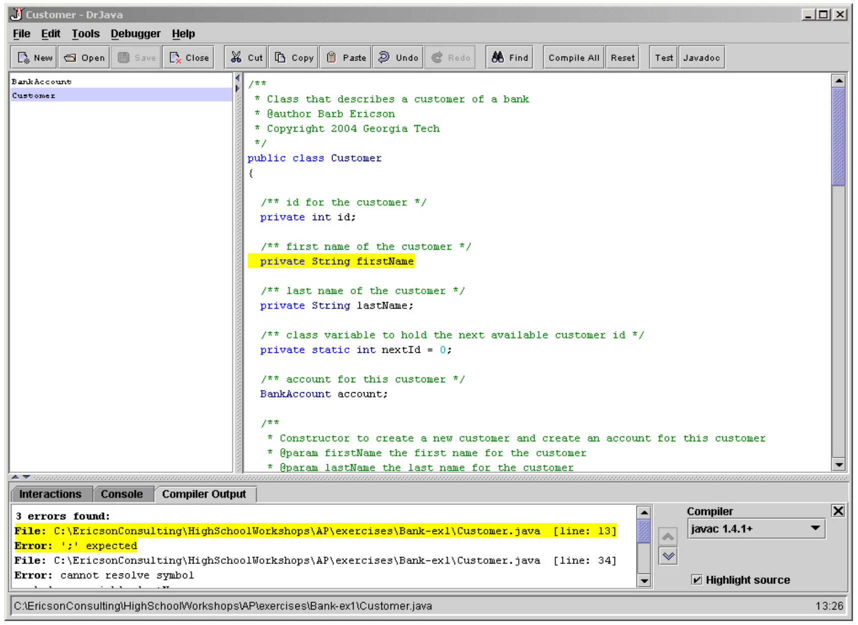


```
7
2
```

C:\EricsonConsulting\HighSchoolWorkshops\AP\exercises\Bank-e

Compiler output pane

- Compiler errors are shown in the compiler output pane
  - The first error will be highlighted and the line of code that caused the problem will be highlighted
  - You can also click on an error to go to the line that contains the error



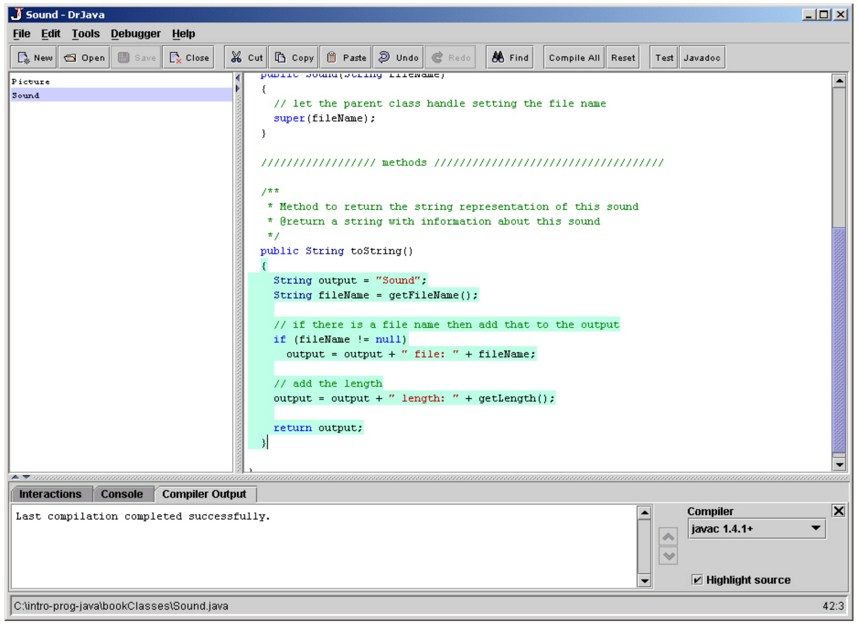
```
/**
 * Class that describes a customer of a bank
 * @author Barb Ericson
 * Copyright 2004 Georgia Tech
 */
public class Customer
{

  /** id for the customer */
  private int id;

  /** first name of the customer */
  private String firstName

  /** last name of the customer */
  private String lastName;

  /** class variable to hold the next available customer id */
  private static int nextId = 0;

  /** account for this customer */
  BankAccount account;

  /**
   * Constructor to create a new customer and create an account for this customer
   * @param firstName the first name for the customer
   * @param lastName the last name for the customer
```

```
3 errors found:
File: C:\EricsonConsulting\HighSchoolWorkshops\AP\exercises\Bank-ex1\Customer.java  [line: 13]
Error: ';' expected
File: C:\EricsonConsulting\HighSchoolWorkshops\AP\exercises\Bank-ex1\Customer.java  [line: 34]
Error: cannot resolve symbol
```

```
C:\EricsonConsulting\HighSchoolWorkshops\AP\exercises\Bank-ex1\Customer.java     13:26
```

How to use the definitions pane

- Click in the definitions pane to add code to a file
  - It will automatically indent for you when you hit enter
    - Or use tab
  - It will highlight all code in a block when you click to the right of a closing parenthesis
  - You can indent selected lines using the Edit menu
  - You can comment out and uncomment lines in the Edit menu


Sound - DrJava
File Edit Tools Debugger Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile All Reset Test Javadoc
Picture
Sound
Interactions Console Compiler Output
Compiler
javac 1.4.1+
Highlight source

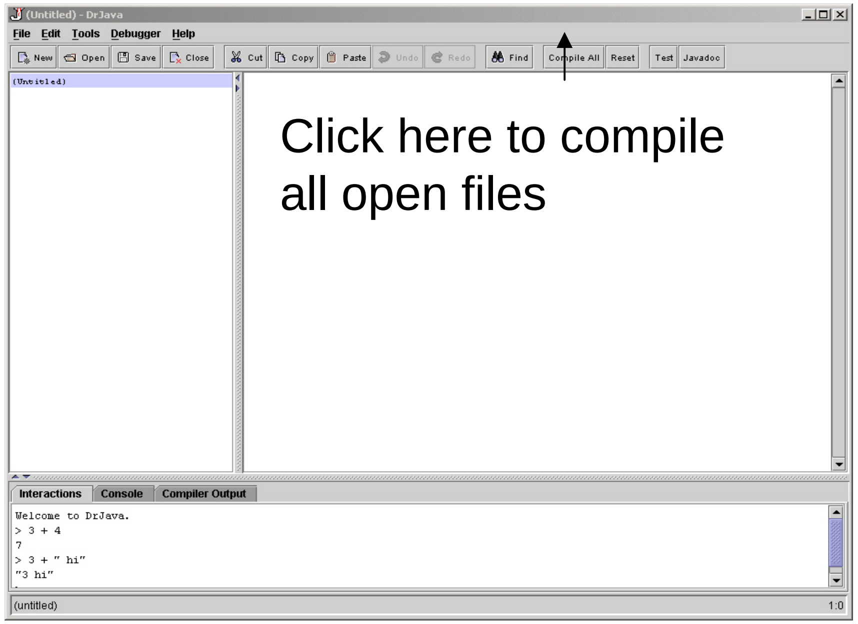
```
  public Sound(String fileName)
  {
    // let the parent class handle setting the file name
    super(fileName);
  }

  ////////////////// methods ////////////////////////////////////

  /**
   * Method to return the string representation of this sound
   * @return a string with information about this sound
   */
  public String toString()
  {
    String output = "Sound";
    String fileName = getFileName();

    // if there is a file name then add that to the output
    if (fileName != null)
      output = output + " file: " + fileName;

    // add the length
    output = output + " length: " + getLength();

    return output;
  }
```

```
Last compilation completed successfully.
```

C:\intro-prog-java\bookClasses\Sound.java 42:3

How to compile programs

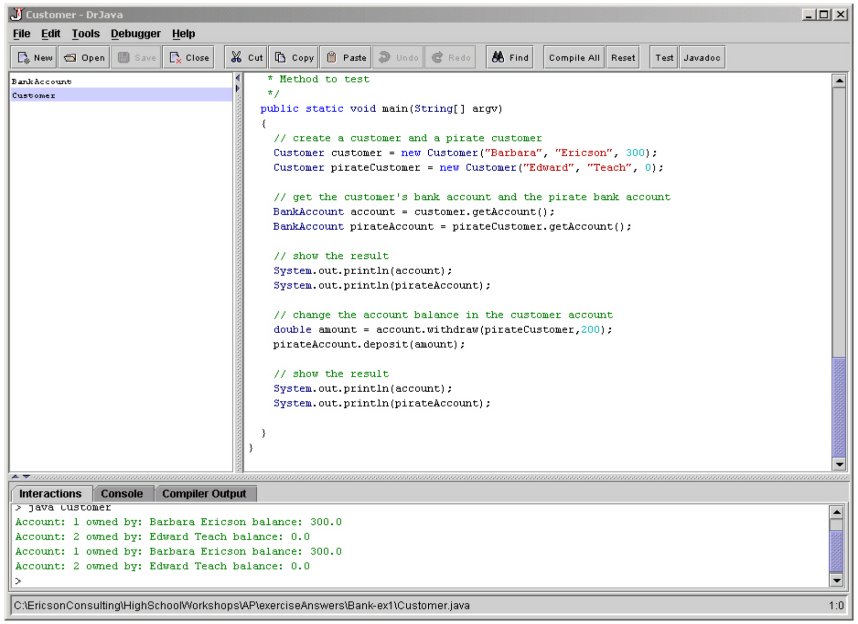
- Click on Compile All to compile all files in the files pane
  - Or use Tools->Compile All Documents
- You can compile just the shown file by using
  - Tools->Compile Current Document



```
Welcome to DrJava.
> 3 + 4
7
> 3 + " hi"
"3 hi"
```

How to execute programs

- Make sure that the file that you want to run the main method from is selected in the files pane
- Click on Tools->Run Document's Main Method
- Output will be shown in the interactions and console panes


Customer - DrJava
File Edit Tools Debugger Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile All Reset Test Javadoc
BankAccount
Customer
Interactions Console Compiler Output

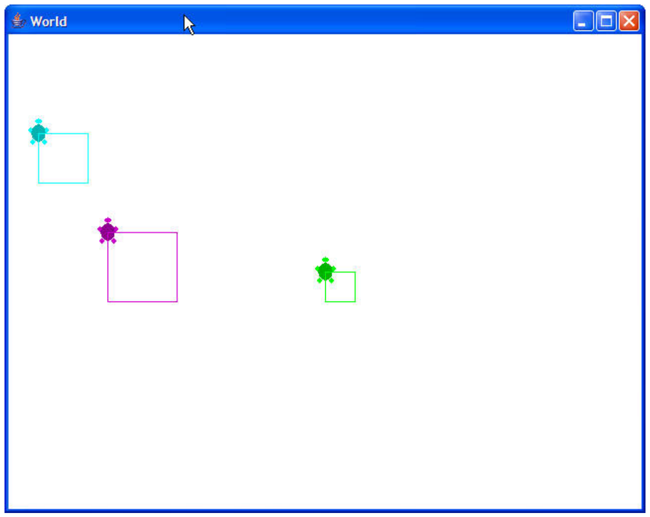
```java
   * Method to test
   */
  public static void main(String[] argv)
  {
    // create a customer and a pirate customer
    Customer customer = new Customer("Barbara", "Ericson", 300);
    Customer pirateCustomer = new Customer("Edward", "Teach", 0);

    // get the customer's bank account and the pirate bank account
    BankAccount account = customer.getAccount();
    BankAccount pirateAccount = pirateCustomer.getAccount();

    // show the result
    System.out.println(account);
    System.out.println(pirateAccount);

    // change the account balance in the customer account
    double amount = account.withdraw(pirateCustomer,200);
    pirateAccount.deposit(amount);

    // show the result
    System.out.println(account);
    System.out.println(pirateAccount);

  }
}
```

```
> java Customer
Account: 1 owned by: Barbara Ericson balance: 300.0
Account: 2 owned by: Edward Teach balance: 0.0
Account: 1 owned by: Barbara Ericson balance: 300.0
Account: 2 owned by: Edward Teach balance: 0.0
>
```

C:\EricsonConsulting\HighSchoolWorkshops\AP\exerciseAnswers\Bank-ex1\Customer.java 1:0

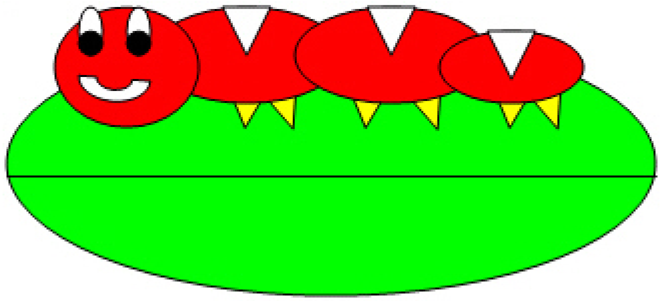
Syllabus

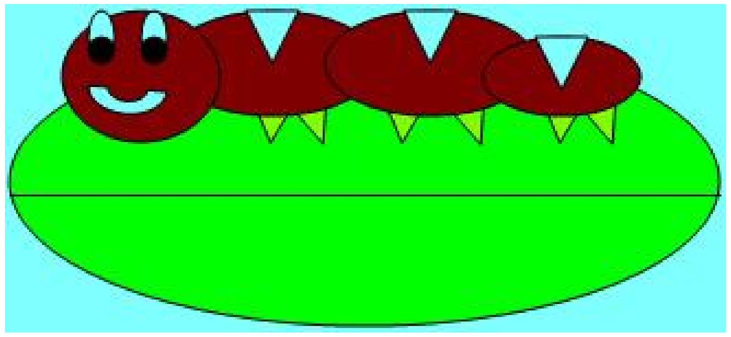
- Introduction to Java
  - Math operators, printing results, data types, casting, relational operators, Strings, variables
- Introduction to Programming
  - Creating and naming objects
    - Using a turtle and a world
  - Creating new Turtle methods
    - Draw simple shapes
    - Using parameters



Syllabus - Continued

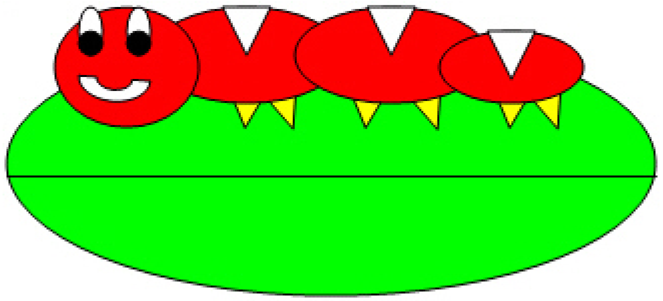
- Modifying Pictures using Loops
  - One-dimensional arrays
  - Use for-each, while, and for loops to
  - Increase/decrease colors, fake a sunset, lighten and darken, create a negative, and grayscale

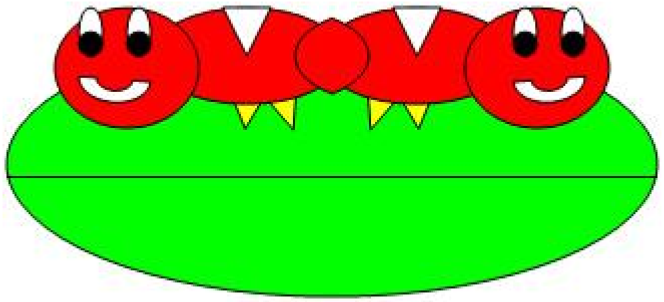




Syllabus - Continued

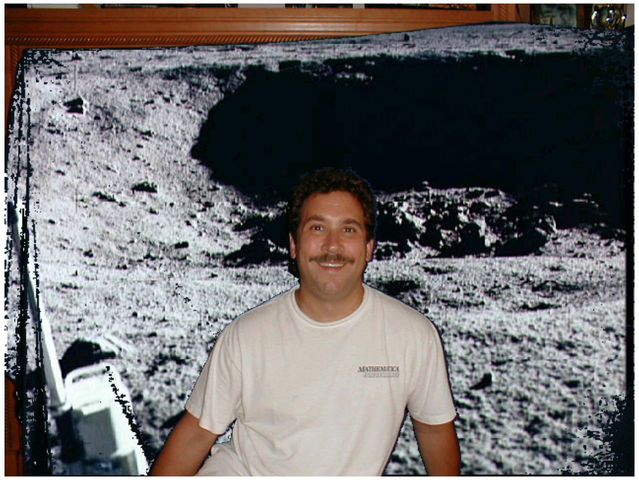
- Modifying Pixels in a Matrix
  - Two-dimensional arrays
  - Nested loops
  - Copying, mirroring, blending, rotation, scaling





Syllabus - Continued

- Conditionally Modifying Pixels
  - Boolean expressions
  - Using && and ||
  - Replacing a color, reducing red-eye, edge detection, sepia-toned, posterize, highlight extremes, blurring, background subtractions, chromakey

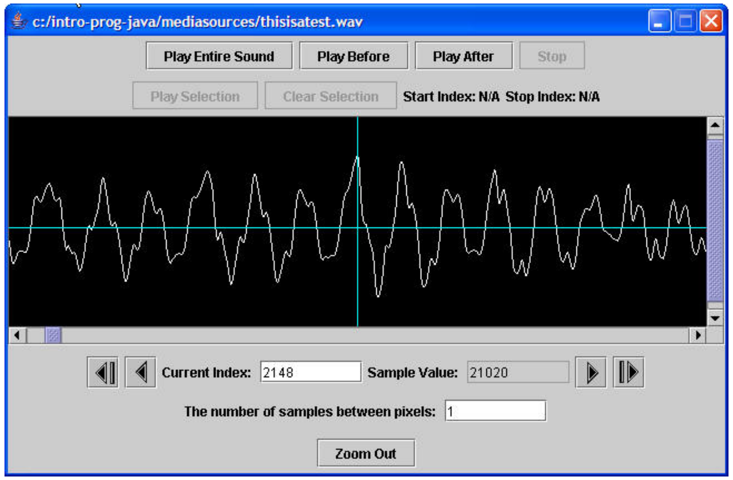


Syllabus - Continued

- Drawing on Pictures
  - Using existing Java classes
  - Inheritance
  - Interfaces
  - Drawing simple shapes, drawing text, general copy, general scale, shearing, gradient paint, general blending, clipping

Syllabus - Continued

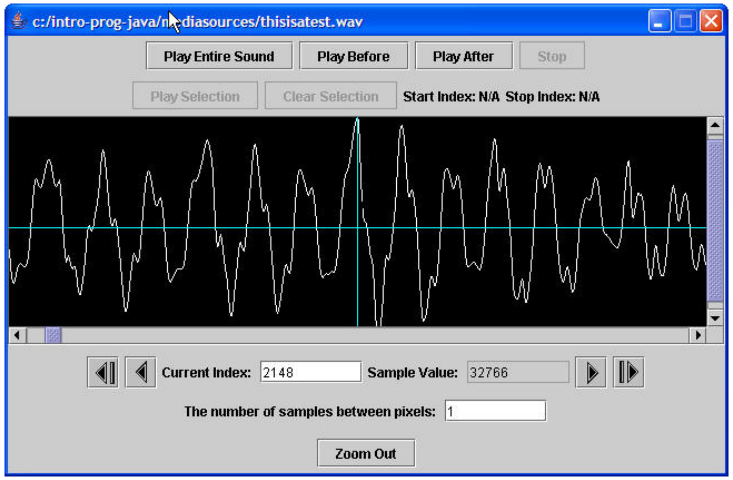
- Modifying all Samples in a Sound
  - 1D arrays
  - Loops
  - Conditional execution
  - Change volume, normalizing a sound (make it as loud as possible), force to extremes



Start Index: N/A Stop Index: N/A

Current Index: 2148 Sample Value: 21020

The number of samples between pixels: 1



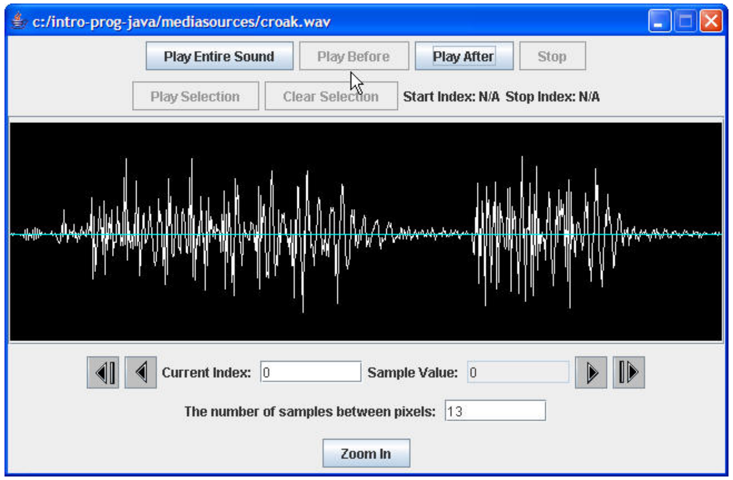
Start Index: N/A Stop Index: N/A

Current Index: 2148 Sample Value: 32766

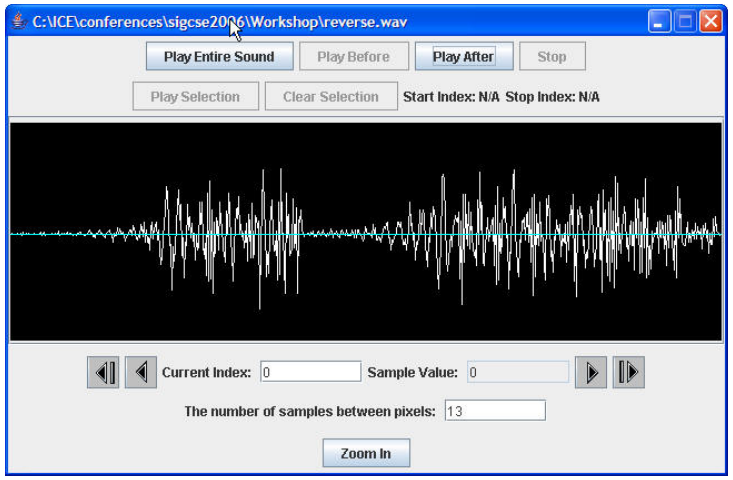
The number of samples between pixels: 1

Syllabus - Continued

- Modifying Samples using Ranges
  - Loops
  - Clipping, splicing, reversing, mirroring



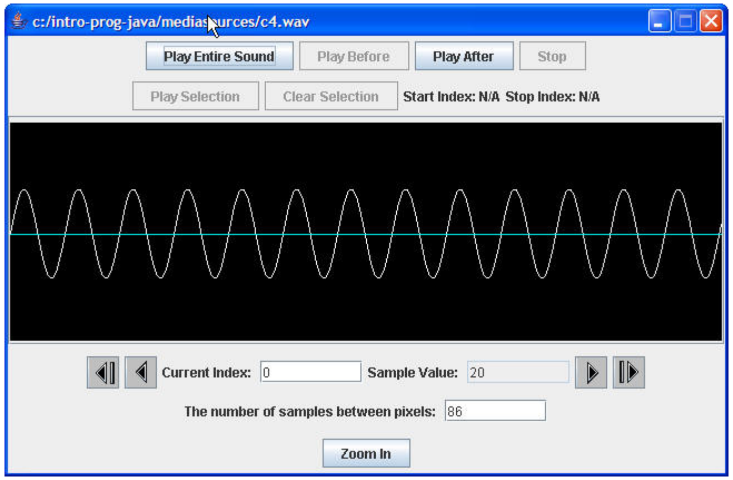
c:/intro-prog-java/mediasources/croak.wav
Play Entire Sound
Play Before
Play After
Stop
Play Selection
Clear Selection
Start Index: N/A Stop Index: N/A
Current Index: 0
Sample Value: 0
The number of samples between pixels: 13
Zoom In

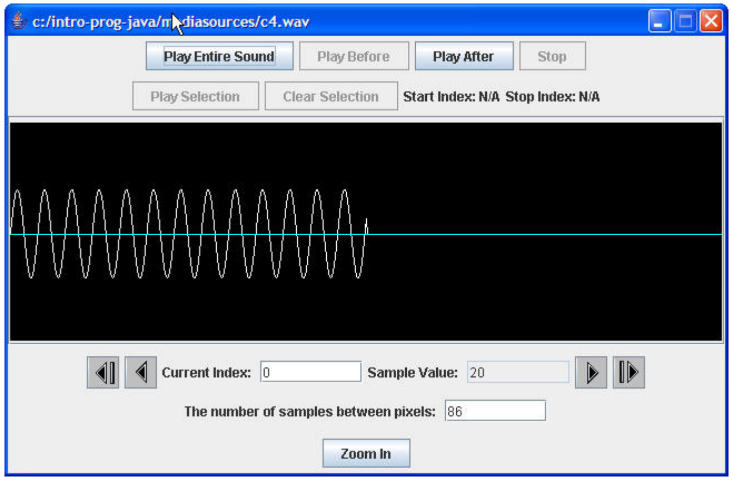


C:\ICE\conferences\sigcse2006\Workshop\reverse.wav
Play Entire Sound
Play Before
Play After
Stop
Play Selection
Clear Selection
Start Index: N/A Stop Index: N/A
Current Index: 0
Sample Value: 0
The number of samples between pixels: 13
Zoom In

Syllabus - Continued

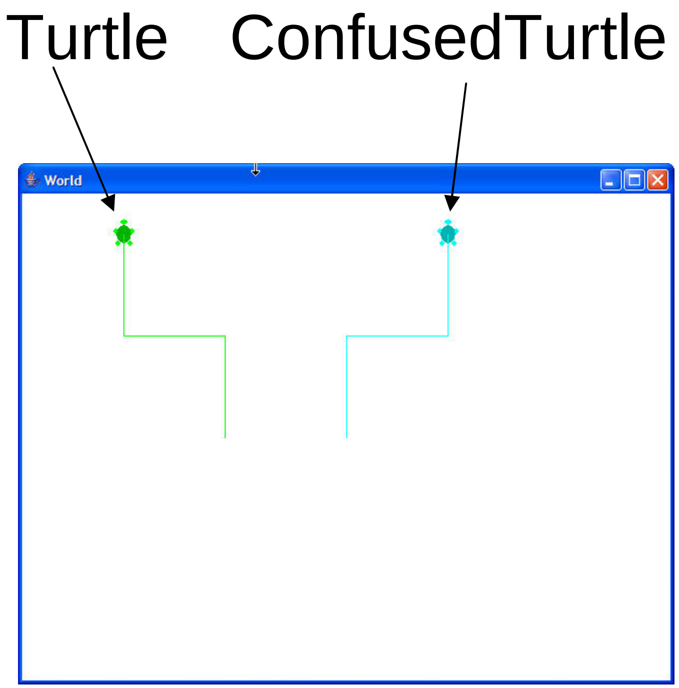
- Combining and Creating Sounds
  - Class and private methods
  - Composing sounds, blending sounds, changing frequencies, and creating echoes
  - Creating sounds
    - Sine Waves, Square Waves, Triangle Waves
  - MP3 and MIDI





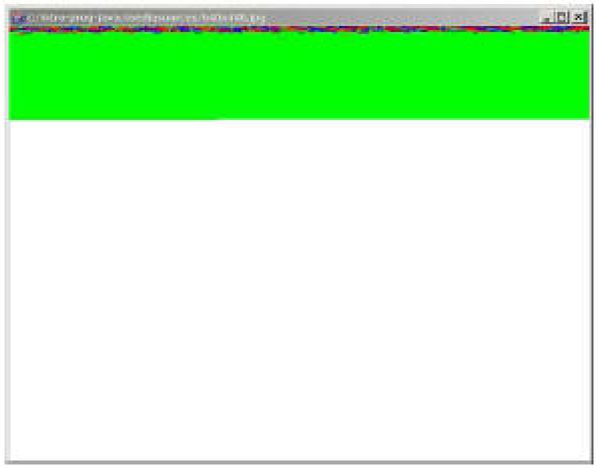
Syllabus - Continued

- Creating Classes
  - Identifying objects and classes
  - Defining a class
  - Overloading constructors
  - Creating and initializing an array
  - Creating getters and setters
  - Creating a main method
  - Javadoc comments
- Reusing a class via inheritance
  - ConfusedTurtle



Syllabus - Continued

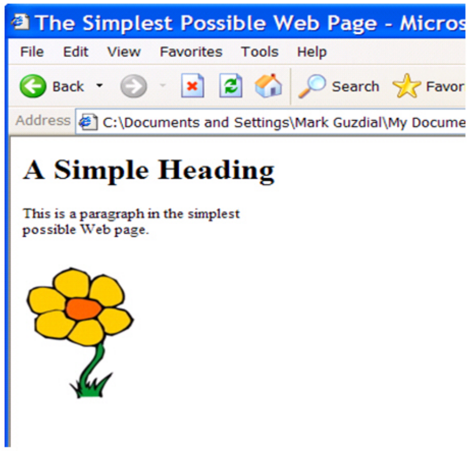
- Creating and Modifying Text
  - String methods
  - Reading from and writing to files
    - Handling Exceptions
  - Creating a form letter
  - Modifying programs
  - Getting text from networks
  - Creating random sentences
  - Using text to shift between media



Syllabus - Continued

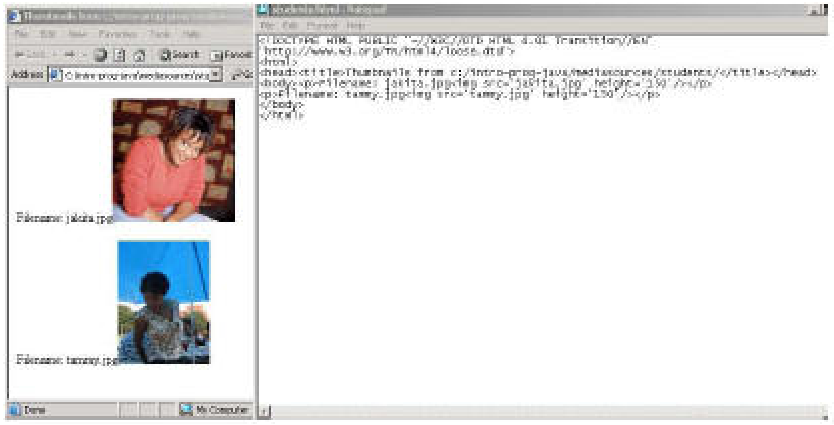
- Making Text for the Web
  - Throwing exceptions, “unnamed” package, HashMap, Generics, and Iterators
  - Generating HTML
  - Create a web page from a directory
  - Create a web page from other web pages
  - Databases
  - Creating a web page from a database

```
<!DOCTYPE HTML PUBLIC "-
    //W3C//DTD HTML 4.01
    Transition//EN"
    "http://www.w3.org/TR/html4/loose.dtd
    ">
<html>
<head>
<title>The Simplest Possible Web
    Page</title>
</head>
<body>
<h1>A Simple Heading</h1>
<p>This is a paragraph in the simplest
    <br /> possible Web page.</p>
<image src="mediasources/flower1.jpg“ />
</body>
</html>
```


The Simplest Possible Web Page - Micros
File Edit View Favorites Tools Help
Back Search Favor
Address C:\Documents and Settings\Mark Guzdial\My Docume

A Simple Heading

This is a paragraph in the simplest
possible Web page.

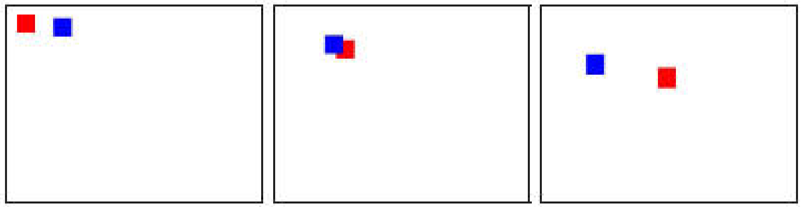


```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head><title>Thumbnails from c:/intro-prog-java/mediasources/students/</title></head>
<body><p>Filename: jakita.jpg<img src='jakita.jpg' height='150' /></p>
<p>Filename: tammy.jpg<img src='tammy.jpg' height='150' /></p>
</body>
</html>
```

Syllabus - Continued

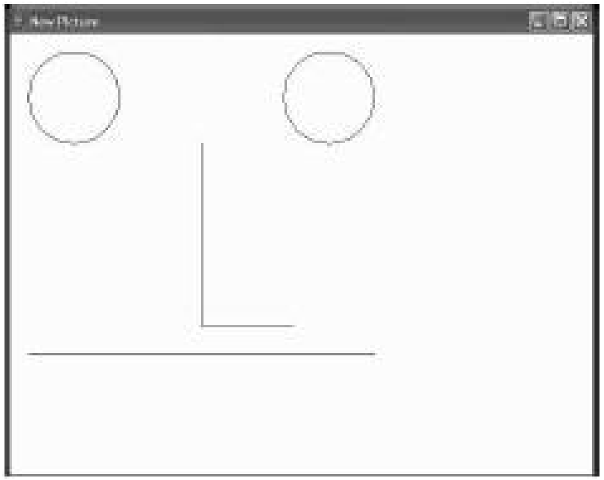
- Encoding, Manipulating, and Creating Movies
  - Frame-based animations with simple shapes and text
  - Special effects – fade out, fake sunset, and chromakey

Buy mo
Buy more widgets
dgets



Syllabus - Continued

- Speed
  - What makes programs fast?
    - Compilers and Interpreters
    - Writing a graphics interpreter and compiler
    - Searching
    - Algorithms that can't be written
  - What makes computers fast?
    - Clock rates, Storage, Display

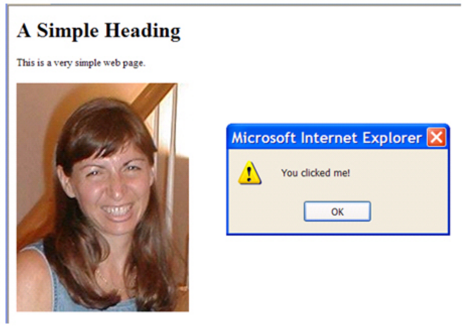


```
circle 20 20 100
circle 300 20 100
line 210 120 210 320
line 210 320 310 320
line 20 350 400 350
```

Syllabus - Continued

- Javascript
  - Syntax
  - User Interfaces
  - Multimedia

```
<body>
<h1>A Simple Heading</h1>
<p>This is a very simple web page.</p>
<p><image
    src="mediasources/barbara.jpg"
    onClick='alert("You clicked me!")' />
</p>

</body>
```

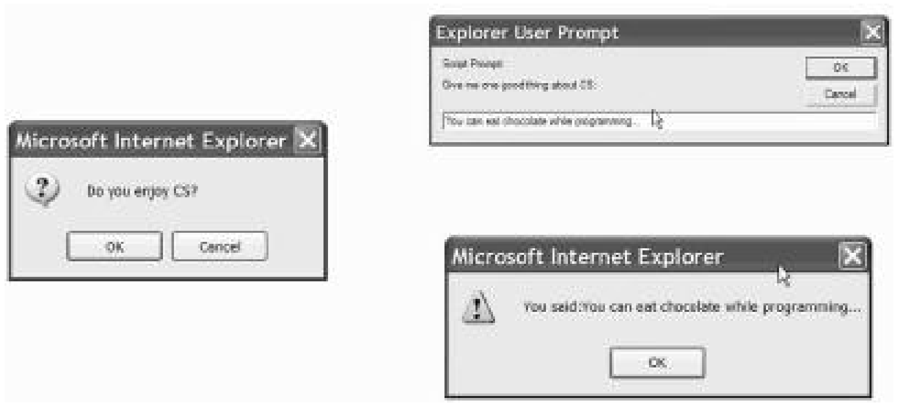


A Simple Heading

This is a very simple web page.

Microsoft Internet Explorer

You clicked me!



Explorer User Prompt

Script Prompt:

Give me one good thing about CS:

You can eat chocolate while programming...

Microsoft Internet Explorer

Do you enjoy CS?

Microsoft Internet Explorer

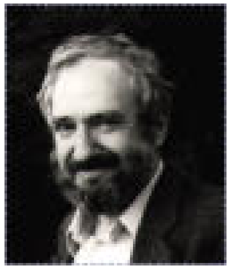
You said:You can eat chocolate while programming...

Turtles: A way of introducing objects

- A first computational object

Computers as Simulators

- "The computer is the Proteus of machines. Its essence is its universality, its power to simulate. Because it can take on a thousand forms and serve a thousand functions, it can appeal to a thousand tastes." Seymour Papert in *Mindstorms*

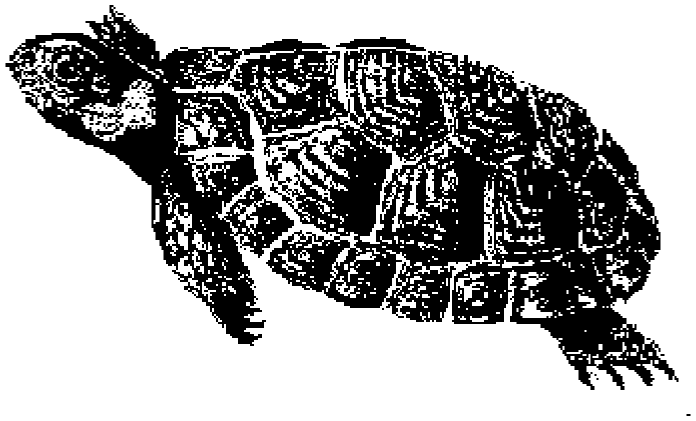


History of Turtles

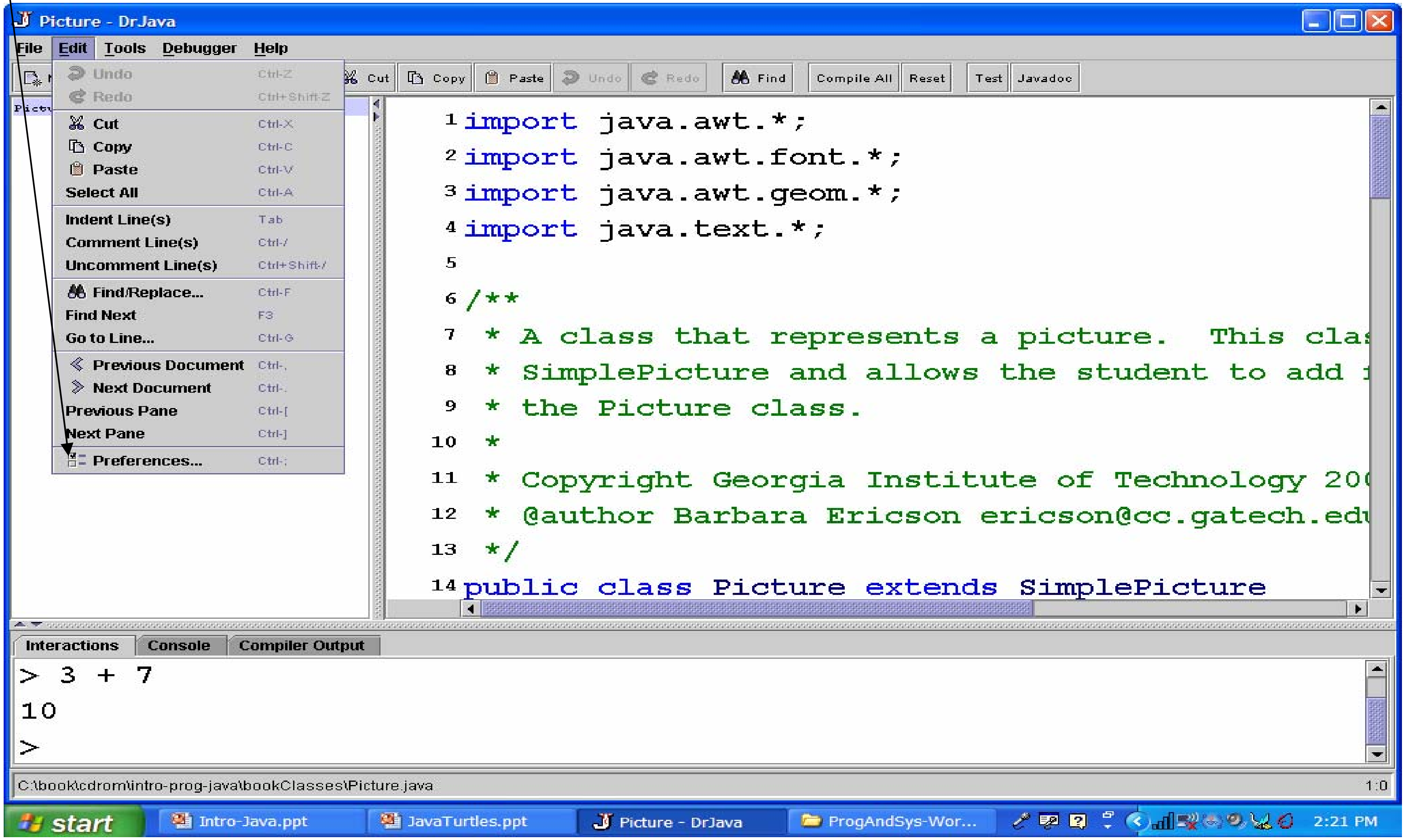
- Seymour Papert at MIT in the 60s
  - By teaching the computer to do something the kids are thinking about thinking
    - Develop problem solving skills
    - Learn by constructing and debugging something
      - Learn by making mistakes and fixing them

Using Turtles

- The Turtle Class was is part of several classes created at Georgia Tech
  - As part of a undergraduate class
- Add bookClasses to your classpath to use these classes



Open Preferences in DrJava

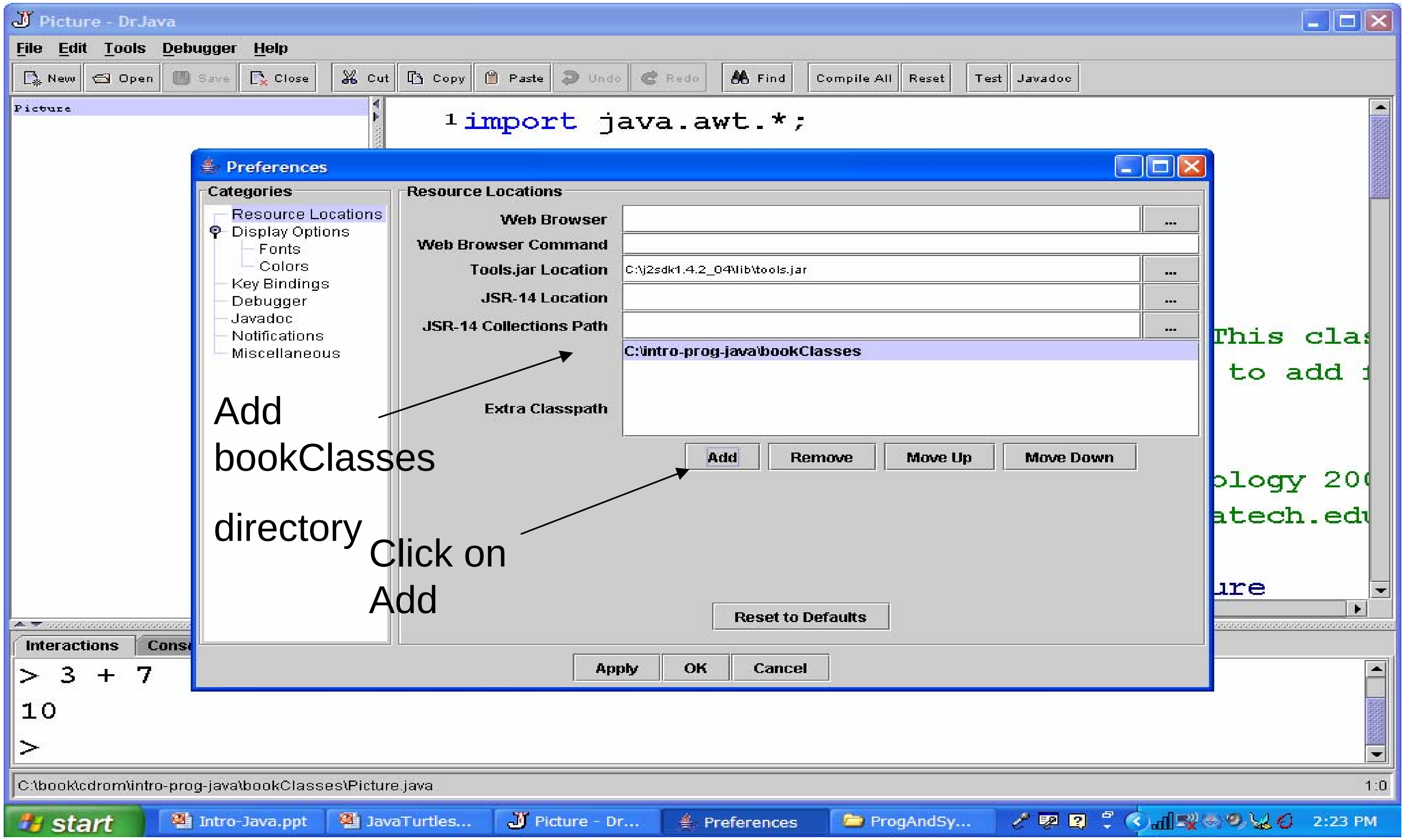


```
import java.awt.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.text.*;

/**
 * A class that represents a picture.  This clas
 * SimplePicture and allows the student to add f
 * the Picture class.
 *
 * Copyright Georgia Institute of Technology 200
 * @author Barbara Ericson ericson@cc.gatech.edu
 */
public class Picture extends SimplePicture
```

```
> 3 + 7
10
>
```

C:\book\cdrom\intro-prog-java\bookClasses\Picture.java 1:0

Adding Book Classes to Classpath


Picture - DrJava
File Edit Tools Debugger Help
New Open Save Close Cut Copy Paste Undo Redo Find Compile All Reset Test Javadoc
Preferences
Categories
Resource Locations
Display Options
Fonts
Colors
Key Bindings
Debugger
Javadoc
Notifications
Miscellaneous
Add Remove Move Up Move Down
Reset to Defaults
Apply OK Cancel
Interactions Console
start Intro-Java.ppt JavaTurtles... Picture - Dr... Preferences ProgAndSy... 2:23 PM

```
import java.awt.*;
```

Resource Locations

Web Browser	
Web Browser Command	
Tools.jar Location	C:\j2sdk1.4.2_04\lib\tools.jar
JSR-14 Location	
JSR-14 Collections Path	
Extra Classpath	C:\intro-prog-java\bookClasses

```
> 3 + 7
10
>
```

C:\book\cdrom\intro-prog-java\bookClasses\Picture.java 1:0

Add bookClasses directory

Click on Add

Creating Objects in Java

- In Java to create an object of a class you use

  new *Class*(*value*, *value*, …);

- Our Turtle objects live in a World object
  - We must create a World object first
  - Try typing the following in the interactions pane:

```
new World();
```

Creating Objects

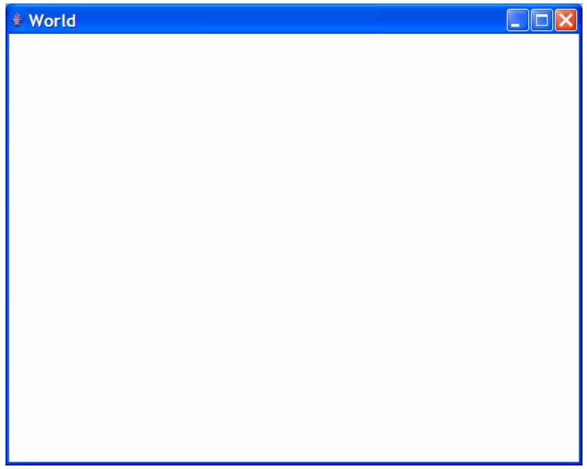
- If you just do
  - new World();
- You will create a new World object and it will display
  - But you will not have any way to refer to it again
  - Once you close the window the object can be garbage collected
    - The memory can be reused
- We need a way to refer to the new object
  - to be able to work with it again

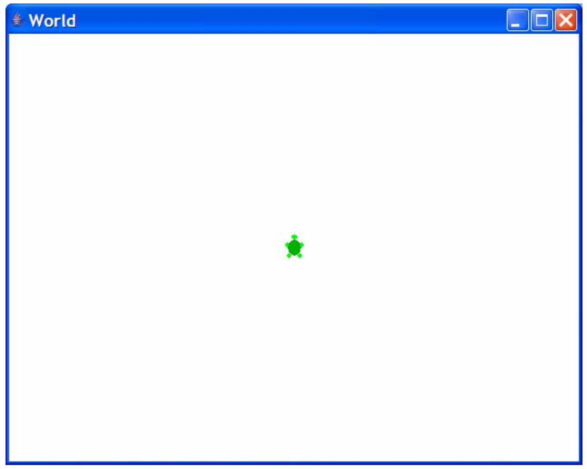
Turtle Basics

- The world starts off with a size of 640 by 480
  - With no turtles

  World world1 = new World();
- The turtle starts off facing north and in the center of the world by default
  - You must pass a World object when you create the Turtle object
    - Or you will get an error: java.lang.NoSuchMethodException: Turtle constructor

  Turtle turtle1 = new Turtle(world1);





Creating Several Objects

- You can create several World object

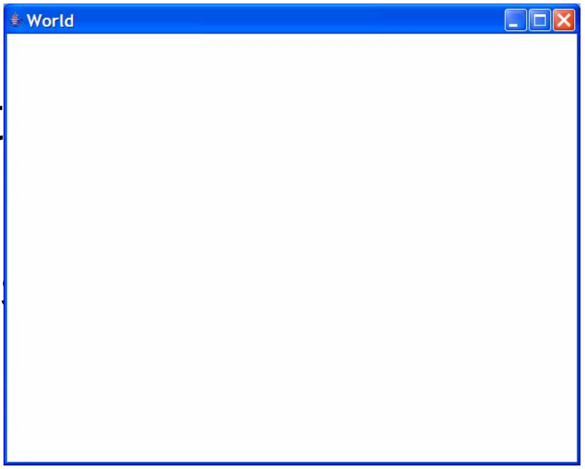
  World world2 = new World();

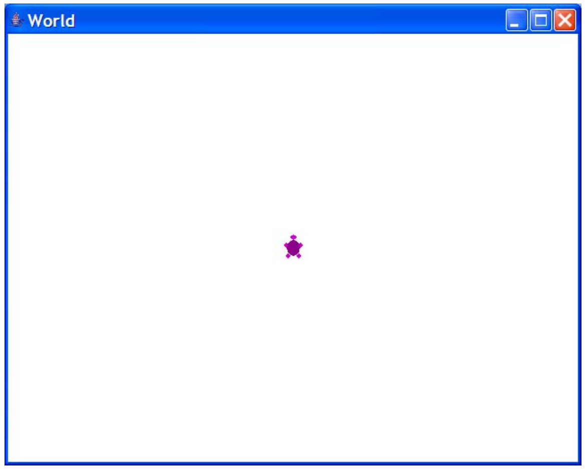
- You can create several Turtle object

  Turtle turtle2 = new Turtle(world2);

  Turtle turtle3 = new Turtle(world2);

  - One turtle is on top of the other





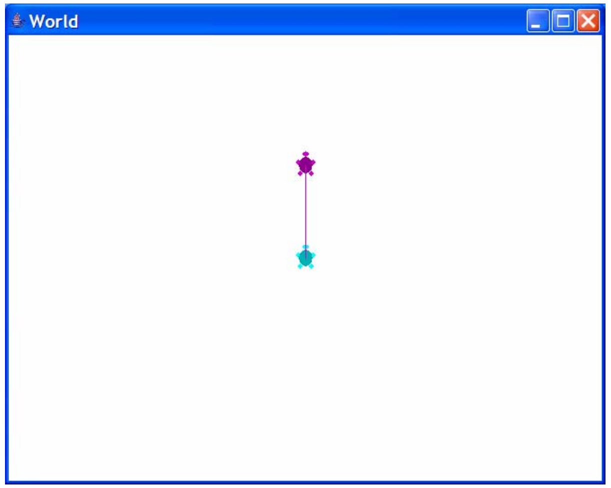
Moving a Turtle

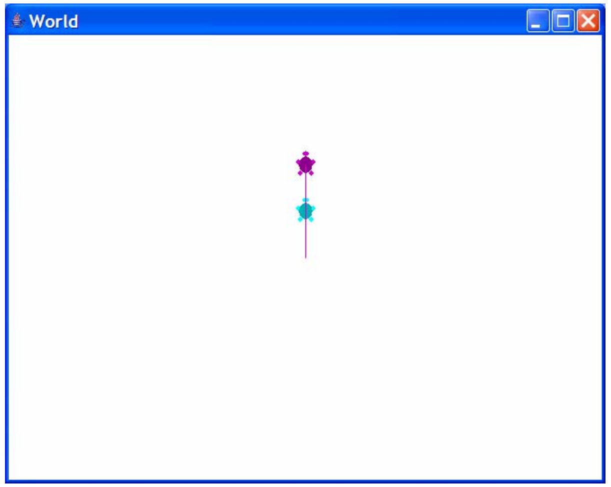
- Turtles can move forward

  ```
  turtle3.forward();
  ```

  - The default is to move by
    - 100 steps (pixels)
- You can also tell the turtle how far to move

  ```
  turtle2.forward(50);
  ```





Turning a Turtle

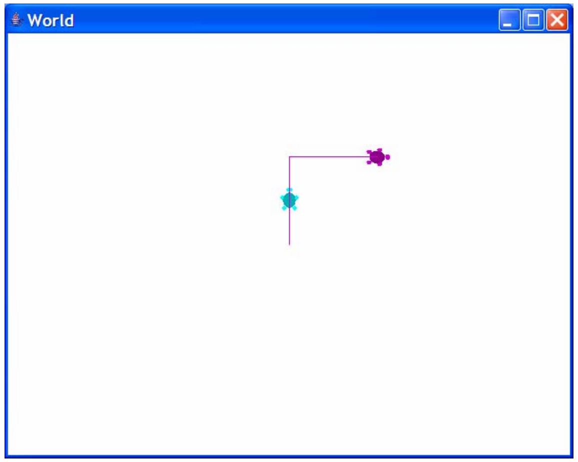
- Turtles can turn
  - Right

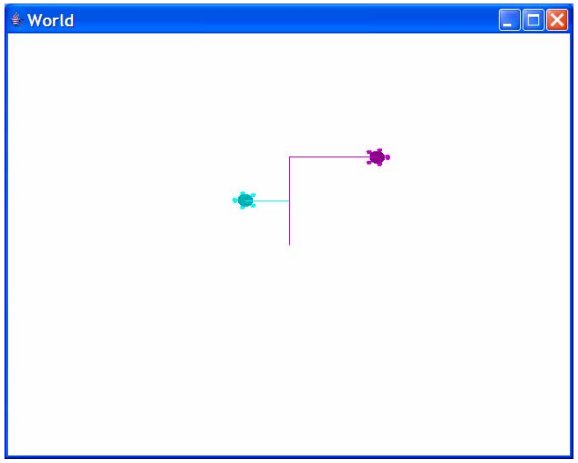
    turtle3.turnRight();

    turtle3.forward();
  - Left

    turtle2.turnLeft();

    turtle2.forward(50);





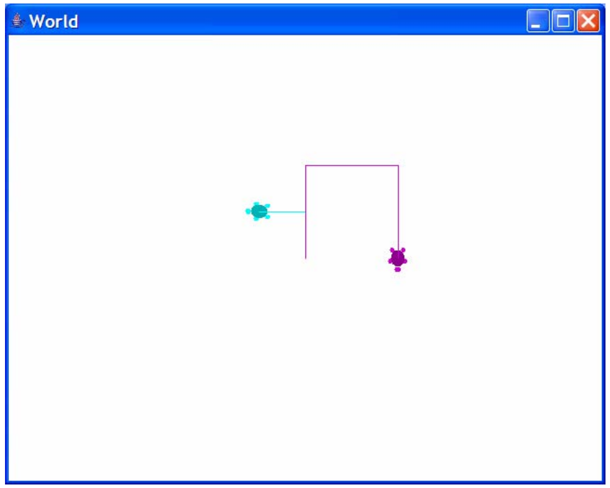
Turning a Turtle

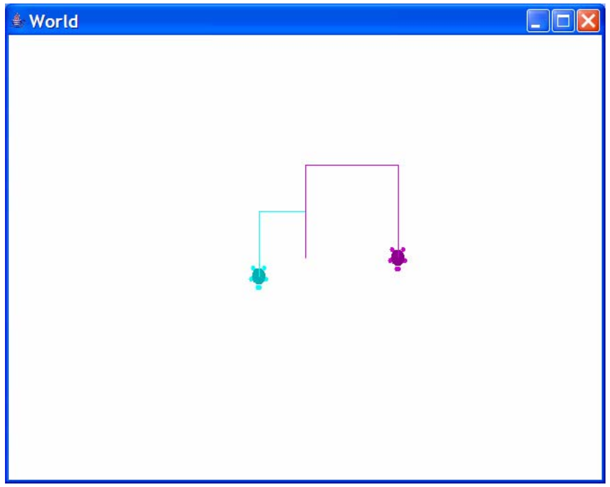
- Turtles can turn by a specified amount
  - A positive number turns the turtle the right

    ```
    turtle3.turn(90);
    turtle3.forward(100);
    ```

  - A negative number turns the turtle to the left

    ```
    turtle2.turn(-90);
    turtle2.forward(70);
    ```





The Pen

- Each turtle has a pen
  - The default is to have the pen down to leave a trail
  - You can pick it up:

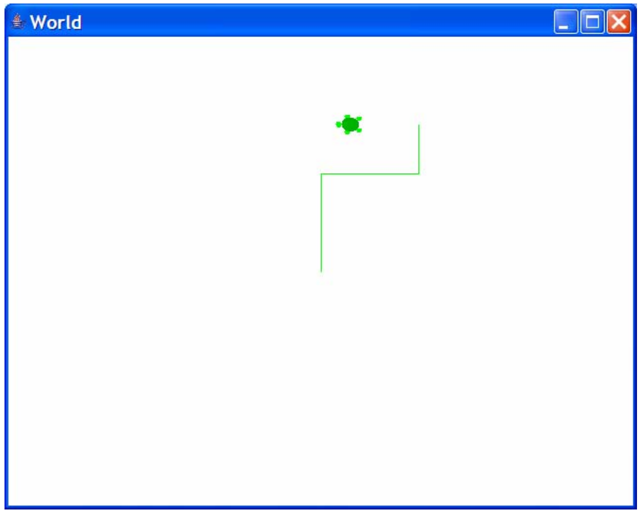
    turtle1.penUp();

    turtle1.turn(-90);

    turtle1.forward(70);
  - You can put it down again:

    turtle1.penDown();

    turtle1.forward(100);



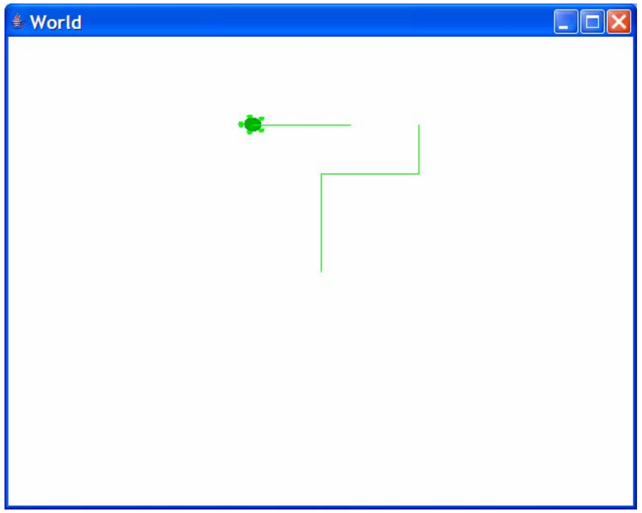


Image Processing in Java

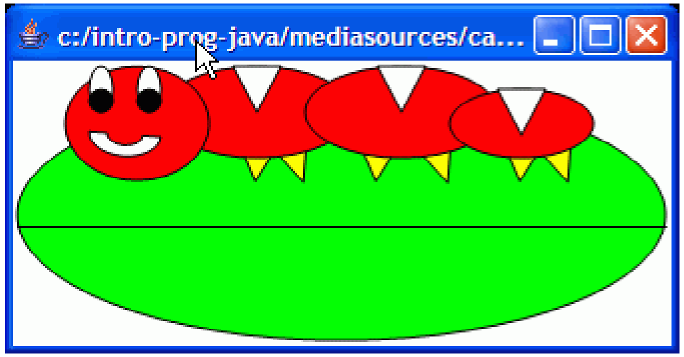
- Simple picture manipulation
- Copying and transforming pictures

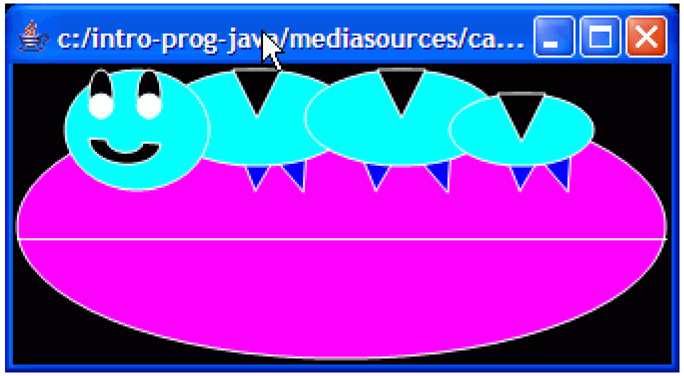
Modifying Pictures using Loops

- Introduce one-dimensional arrays
- Change pixel colors one at a time by hand
- Change pixel colors in a loop
  - For-each
  - While
  - For
- Image Manipulations
  - Decrease/increase a color
  - Set a color to zero
  - Negate
  - Grayscale

Negating an Image

- How would you turn a picture into a negative?
  - White should become black
    - 255,255,255 becomes 0,0,0
  - Black should become white
    - 0,0,0 becomes 255,255,255
- The new color is
  - 255 – red, 255 – green, 255 - blue



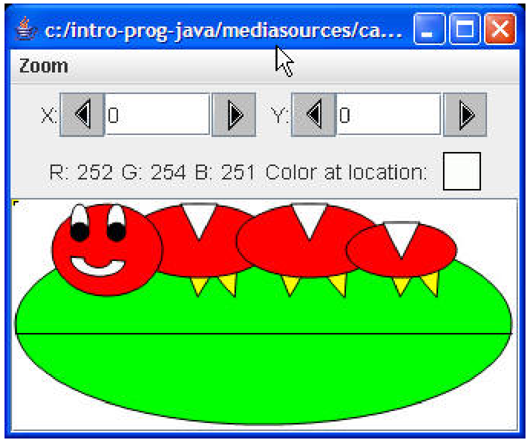


Negate Method

```
/**
 * Method to negate the picture
 */
public void negate()
{
    Pixel[] pixelArray = this.getPixels();
    Pixel pixelObj = null;
    int redValue, blueValue, greenValue = 0;

    // loop through all the pixels
    for (int i = 0; i < pixelArray.length; i++)
 {
   // get the current pixel
   pixelObj = pixelArray[i];

   // get the values
   redValue = pixelObj.getRed();
   greenValue =
                pixelObj.getGreen();
   blueValue = pixelObj.getBlue();

   // set the pixel's color
   pixelObj.setColor(
         new Color(255 - redValue,
                       255 - greenValue,
                       255 - blueValue));
  }
}
```

Testing Negate

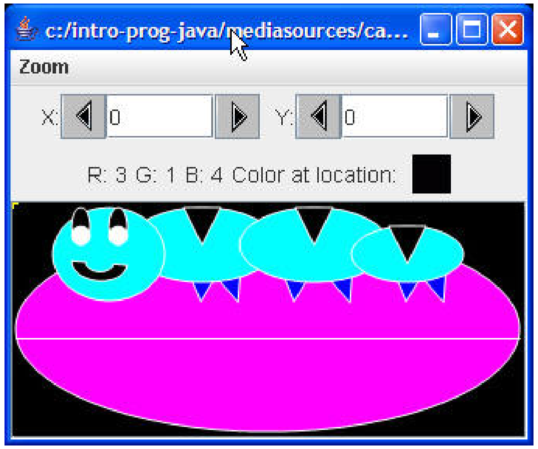
String file =
    “c:/intro-prog-java/mediasources/caterpillar.jpg”;
Picture pictureObj = new Picture(file);
pictureObj.explore();
pictureObj.negate();
pictureObj.explore();


c:/intro-prog-java/mediasources/ca...

Zoom

X: 0 Y: 0

R: 252 G: 254 B: 251 Color at location:


c:/intro-prog-java/mediasources/ca...

Zoom

X: 0 Y: 0

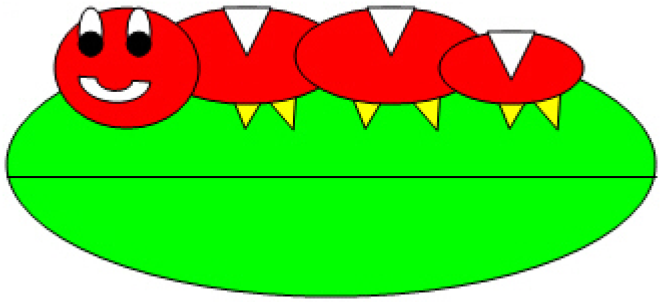
R: 3 G: 1 B: 4 Color at location:

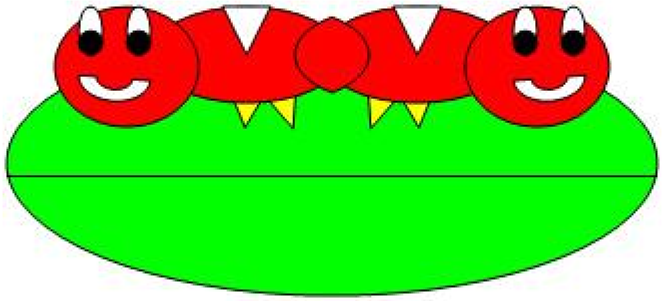
Modifying Pixels in a Matrix

- Introduce two-dimensional arrays
- Use Nested Loops
- Image Manipulations
  - Copying
  - Mirroring
  - Blending
  - Rotation
  - Scaling
  - Create a collage

Vertical Mirroring

- What if we want to pretend to place a mirror in the middle of the picture
  - We would see the left side of the picture mirrored on the right side





Thinking Through Vertical Mirroring

- If we just think of a number at each x and y location instead of a color
  - The mirroring would look like this:
- Can you find the algorithm to do this?

	0	1	2	3	4
0	1	2	3	4	5
1	5	4	3	2	1
2	1	2	3	4	5

1	2	3	2	1
5	4	3	4	5
1	2	3	2	1

What is the Vertical Mirror for this?

- Try the solve the problem for small samples
- If you can’t solve it on a small sample
  - You can’t write a program to solve it

	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9

	0	1	2
0			
1			
2			

Mirror Vertical Algorithm

- Loop through all the rows (y starts at 0, increments by 1, and is less than the picture height)
  - Loop with x starting at 0 and x less than the midpoint (mirror point) value
    - Get the left pixel at x and y
    - Get the right pixel at width – 1 - x
    - Set the color for the right pixel to be the color of the left pixel

1	2	3	4	5
5	4	3	2	1
1	2	3	4	5

1	2	3	2	1
5	4	3	4	5
1	2	3	2	1

Mirror Vertical Algorithm to Code

- We are going to need the midpoint

  ```
  int midpoint = this.getWidth() / 2;
  ```

- Loop through the rows (y values)

  ```
  for (int y = 0; y < this.getHeight(); y++) {
  ```

  - Loop through x values (starting at 1)

    ```
    for (int x = 0; x < midpoint; x++) {
    ```

    - Set right pixel color to left pixel color

      ```
      Pixel leftPixel = this.getPixel(x, y);
      Pixel rightPixel = this.getPixel(this.getWidth() - 1 - x, y);
      rightPixel.setColor(leftPixel.getColor());
      ```

Mirror Vertical Method

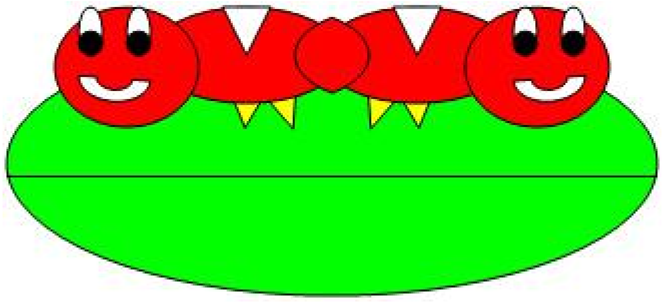
```
public void mirrorVertical()
 {
   int mirrorPoint = this.getWidth() / 2;
   Pixel leftPixel = null;
   Pixel rightPixel = null;

   // loop through the rows
   for (int y = 0; y < this.getHeight(); y++)
   {
     // loop from 0 to just before the mirror point
     for (int x = 0; x < mirrorPoint; x++)
     {
```

Mirror Vertical Method - Continued

```
      leftPixel = this.getPixel(x, y);
      rightPixel = this.getPixel(this.getWidth() – 1 – x, y);
      rightPixel.setColor(leftPixel.getColor());
    }
  }
}
```

Trying Mirror Vertical

- Create the picture
  - Picture p1 = new Picture(
    FileChooser.getMediaPath(“caterpillar.jpg”);
- Invoke the method on the picture
  - p1.mirrorVertical();
- Show the picture
  - p1.show();



Scaling Down a Picture

- passionFlower.jpg is 640pixels wide and 480 pixels high
- If we copy every other pixel we will have a new picture with width (640 / 2 = 320) and height (480 / 2 = 240)

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

0	2
8	10

Scaling Down Method

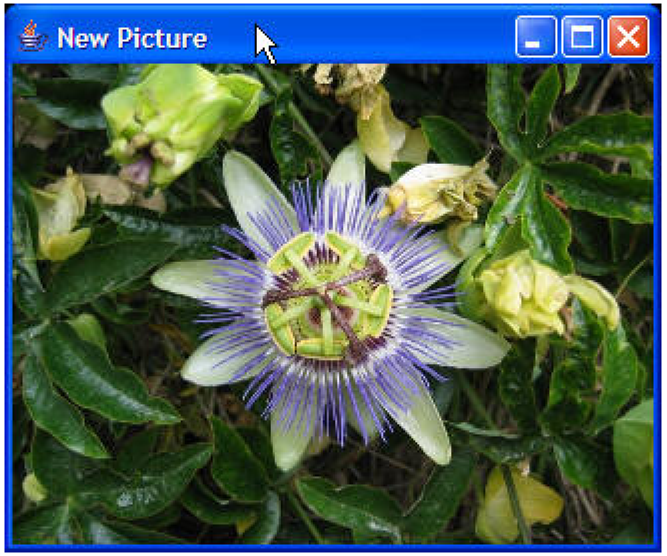
```
public void copyFlowerSmaller()
  {
    Picture flowerPicture =
      new Picture(
        FileChooser.getMediaPath(“passionFlower.jpg"));
    Pixel sourcePixel = null;
    Pixel targetPixel = null;

    // loop through the columns
    for (int sourceX = 0, targetX=0;
         sourceX < flowerPicture.getWidth();
         sourceX+=2, targetX++)
    {
```

Scaling Down Method - Continued

```
   // loop through the rows
   for (int sourceY=0, targetY=0;
        sourceY < flowerPicture.getHeight();
        sourceY+=2, targetY++)
   {
     sourcePixel =
         flowerPicture.getPixel(sourceX,sourceY);
     targetPixel = this.getPixel(targetX,targetY);
     targetPixel.setColor(sourcePixel.getColor());
   }
  }
}
```

Trying Copy Flower Smaller

- Create a new picture half the size of the original picture (+ 1 if odd size)
  - Picture p1 = new Picture(320,240);
- Copy the flower to the new picture
  - p1.copyFlowerSmaller();
- Show the result
  - p1.show();

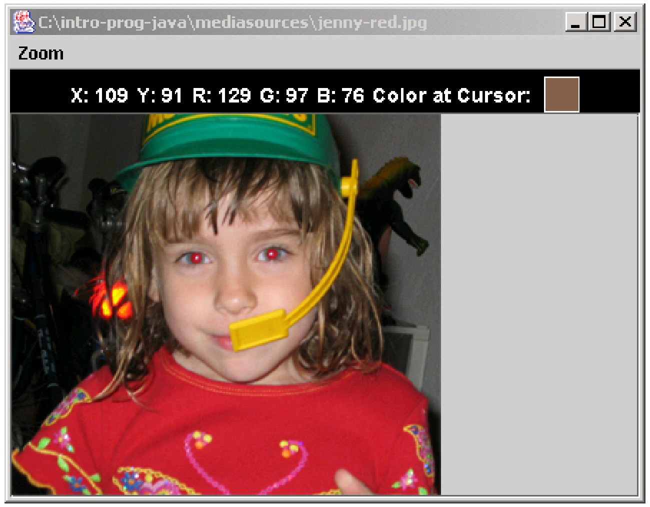


Conditionally Modifying Pixels

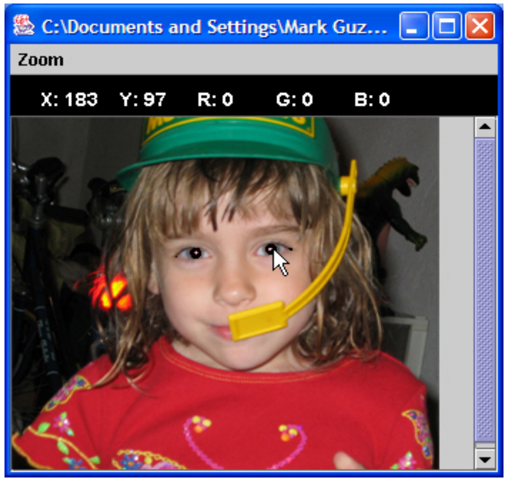
- Boolean expressions
- Using && and || to combine boolean expressions
- Image Manipulations
  - Replacing a color
  - Reducing red-eye
  - Edge detection
  - Sepia-toned
  - Posterize
  - Highlight extremes
  - Blurring
  - Background subtraction and Chromakey

Remove Red Eye

- Red eye is when the flash from the camera is reflected from the subject's eyes
- We want to change the red color in the eyes to another color
  - But not change the red of her dress



X: 109 Y: 91 R: 129 G: 97 B: 76 Color at Cursor:



X: 183 Y: 97 R: 0 G: 0 B: 0

Red Eye Algorithm

- We can find the area around the eyes to limit where we change the colors
  - Using pictureObj.explore()
  - But we still just want to change the pixels that are “close to” red.
  - We can find the distance between the current color and our definition of red
    - And change the color of the current pixel only if the current color is within some distance to the desired color

Color Distance

- The distance between two points is computed as
  - Square root of ((x1 – x2)^2 + (y1 – y2)^2)
- The distance between two colors can be computed
  - Square root of ((red1 – red2)^2 + (green1-green2)^2 + (blue1 – blue2)^2)
  - There is a method in the Pixel class to do this
    - double dist = pixelObj.colorDistance(color1);

Remove Red Eye Method

```
public void removeRedEye(int startX, int startY, int endX,
                           int endY, Color newColor)
  {
    Pixel pixelObj = null;

    // loop through the pixels in the rectangle defined by the // startX,
     startY, and endX and endY
    for (int x = startX; x < endX; x++)
    {
      for (int y = startY; y < endY; y++)
      {

        // get the current pixel
        pixelObj = getPixel(x,y);
```

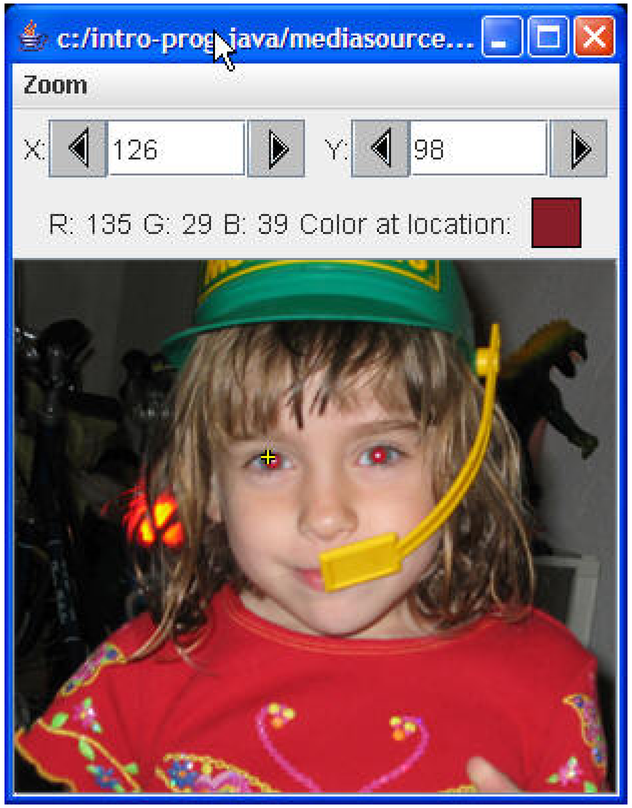
Remove Red Eye Method

```
        // if the color is near red then change it
        if (pixelObj.colorDistance(Color.red) < 167)
       {
         pixelObj.setColor(newColor);
       }
      }
    }
  }
```

Testing removeRedEye

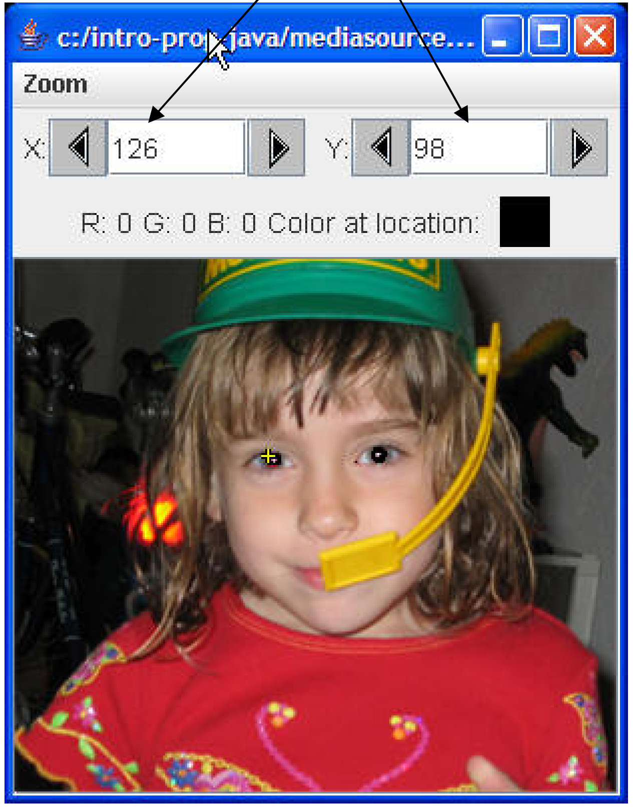
```
String file =
        FileChooser.getMediaPath(“jenny-red.jpg”);
Picture p = new Picture(file);
p.explore();
p.removeRedEye(110,91,192,103,
   java.awt.Color.BLACK);
p.explore();
```

Exploring Remove Red Eye



R: 135 G: 29 B: 39 Color at location:

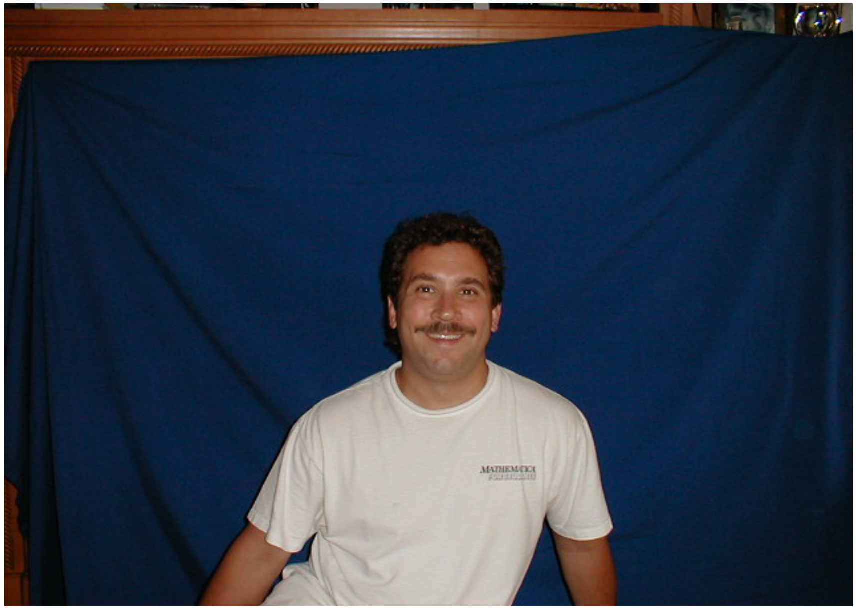
Type in x and y



R: 0 G: 0 B: 0 Color at location:

Chroma Key – Blue Screen

- For TV and movie special effects they use a blue or green screen
    - Here just a blue sheet was used
    - Professionally you need an evenly lit, bright, pure blue background
        - With nothing blue in the scene



Chroma Key

- Write the method chromakey that takes a new background picture as an input parameter
  - It will loop through all the pixels
  - If the pixel color is blue (red + green < blue)
  - Replace the pixel color with the color from the new background pixel (at the same location)

Chromakey Method

```
public void chromakey(Picture newBg)
  {
    Pixel currPixel = null;
    Pixel newPixel = null;

    // loop through the columns
    for (int x=0; x<getWidth(); x++)
    {

      // loop through the rows
      for (int y=0; y<getHeight(); y++)
      {

        // get the current pixel
        currPixel = this.getPixel(x,y);
```

Chromakey Method - Cont

```
        /* if the color at the current pixel is mostly blue
        * (blue value is greater than red and green combined),
        * then use the new background color
        */
       int combindedColor =
         currPixel.getRed() + currPixel.getGreen();
       if (combindedColor < currPixel.getBlue())
       {
         newPixel = newBg.getPixel(x,y);
         currPixel.setColor(newPixel.getColor());
       }
     }
   }
 }
```

Testing chromakey

- Picture markP = new Picture(FileChooser.getMediaPath(“blue-mark.jpg”));
- Picture newBack = new Picture(FileChooser.getMediaPath(“moon-surface.jpg”));
- markP.chromakey(newBack);
- markP.show();

Sound Processing

- Some simple Java examples
  - new Sound(filename)
  - sound.getSamples()
  - sample.getValue(), sample.setValue()

Changing Sound Samples

- One-dimensional arrays
- Looping through all sound samples
  - for-each, while, for
- Conditional Execution
  - Using if and else
- Sound Manipulations
  - Decrease/increase volume
  - Normalize a sound (make as loud as possible)
  - Force to extremes

Force to Extremes

- What if we want to make all values in a sound the maximum positive or negative value?
  - If the value is positive (>=0) make it 32,767
  - else make it -32,768
- We need a way to execute code based on if a test is true
  - We can use a conditional (if and else)

Force to Extremes Method

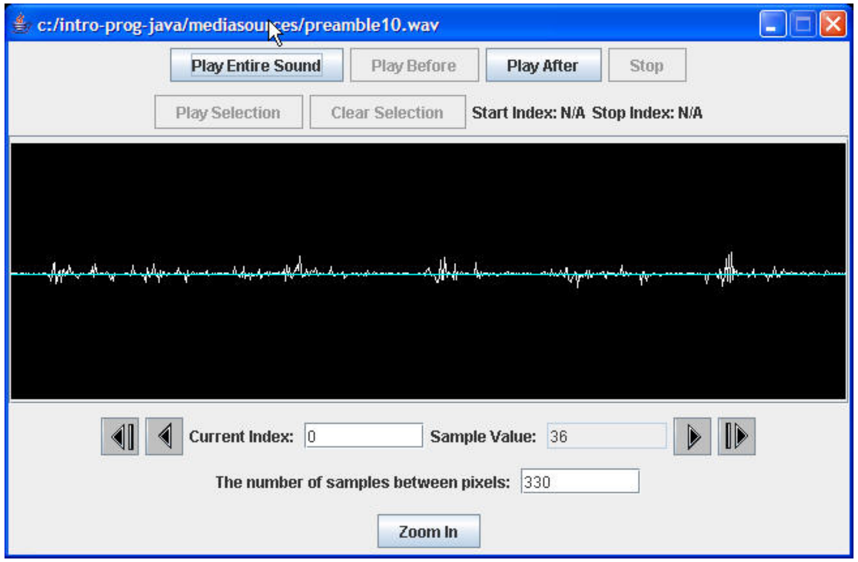
```
public void forceToExtremes()
  {

    SoundSample[] sampleArray =
     this.getSamples();
    SoundSample sample = null;

    // loop through the sample
     values
    for (int i = 0; i <
     sampleArray.length; i++)
    {
      // get the current sample
      sample = sampleArray[i];

      /* if the value was positive or 0
      set to the maximum
        * positive value
        */
       if (sample.getValue() >= 0)
         sample.setValue(32767);

       /* else (must be less than 0)
      so set it to the minimum
        * negative value
        */
       else
         sample.setValue(-32768);
    }
  }
```

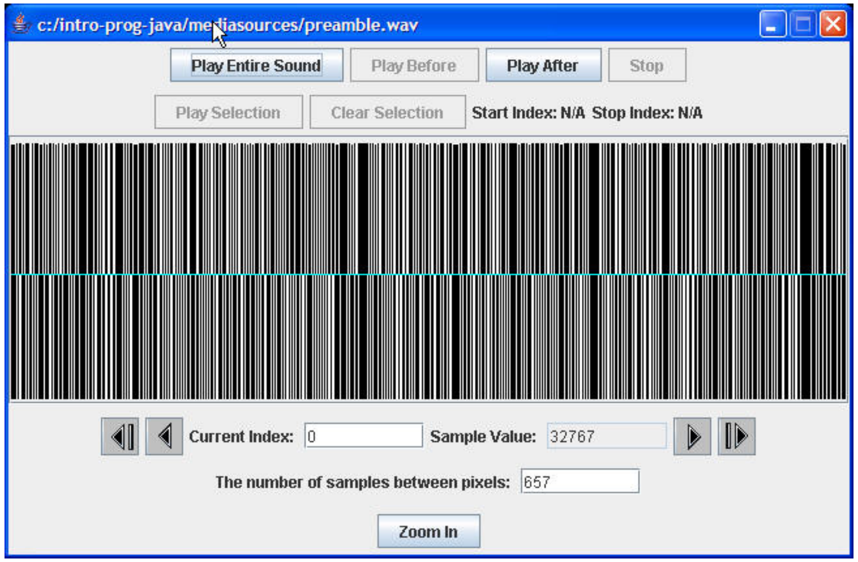
Testing forceToExtremes

- String file =
  FileChooser.getMediaPath(“preamble10.wav“);
- Sound soundObj = new Sound(file);
- soundObj.explore();
- soundObj.forceToExtremes();
- soundObj.explore();

Before After

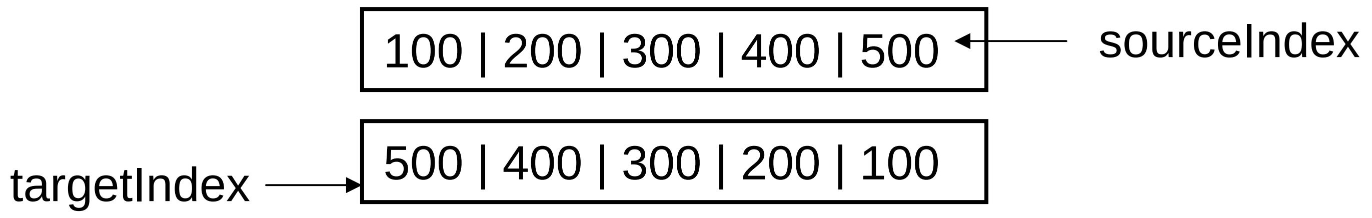
Exploring Force To Extremes


c:/intro-prog-java/mediasources/preamble10.wav
Play Entire Sound
Play Before
Play After
Stop
Play Selection
Clear Selection
Start Index: N/A Stop Index: N/A
Current Index: 0
Sample Value: 36
The number of samples between pixels: 330
Zoom In


c:/intro-prog-java/mediasources/preamble.wav
Play Entire Sound
Play Before
Play After
Stop
Play Selection
Clear Selection
Start Index: N/A Stop Index: N/A
Current Index: 0
Sample Value: 32767
The number of samples between pixels: 657
Zoom In

Reversing a Sound

- To reverse a sound
  - Create a copy of the original sound
    - Sound orig = new Sound(this.getFileName());
  - Then loop starting the sourceIndex at the last index in the source and the targetIndex at the first index in the target
    - Decrement the sourceIndex each time
    - Increment the targetIndex each time



Reversing Method

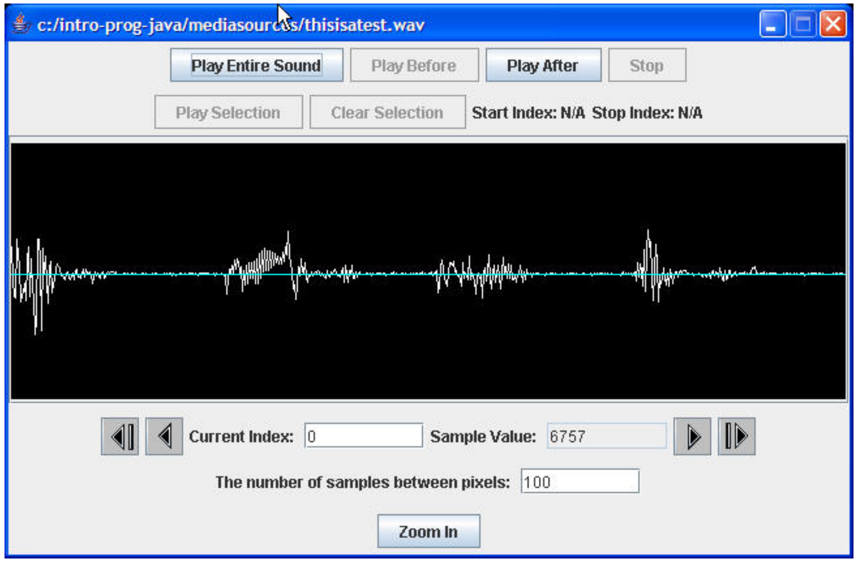
```
public void reverse()
{
    Sound orig = new Sound(this.getFileName());
    int length = this.getLength();

    // loop through the samples
    for (int targetIndex = 0, sourceIndex = length - 1;
         targetIndex < length && sourceIndex >= 0;
         targetIndex++, sourceIndex--)
      this.setSampleValueAt(targetIndex,
                            orig.getSampleValueAt(sourceIndex));
}
```

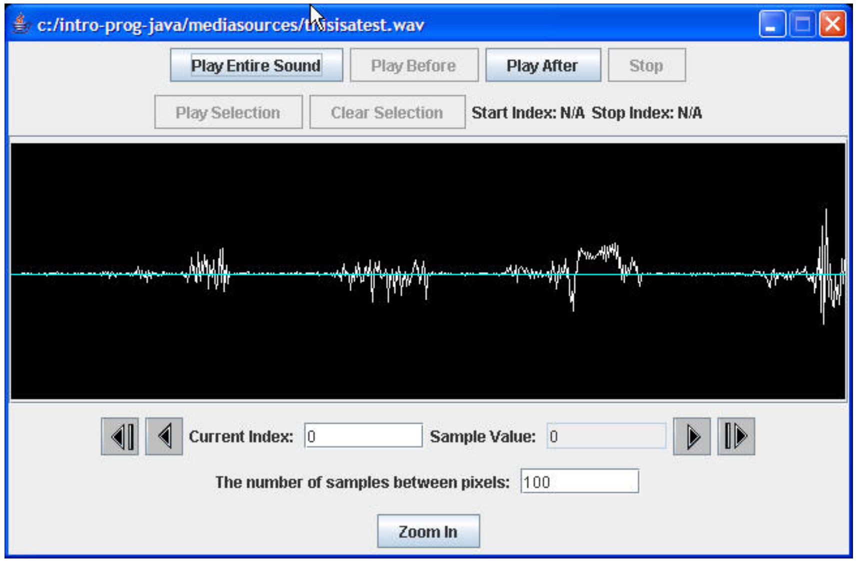
Testing the Reverse Method

```
String file = FileChooser.getMediaPath(
                  “thisisatest.wav”);
Sound s = new Sound(file);
s.explore();
s.reverse();
s.explore();
```

Before After

Exploring Reverse


c:/intro-prog-java/mediasources/thisisatest.wav
Play Entire Sound
Play Before
Play After
Stop
Play Selection
Clear Selection
Start Index: N/A Stop Index: N/A
Current Index: 0
Sample Value: 6757
The number of samples between pixels: 100
Zoom In

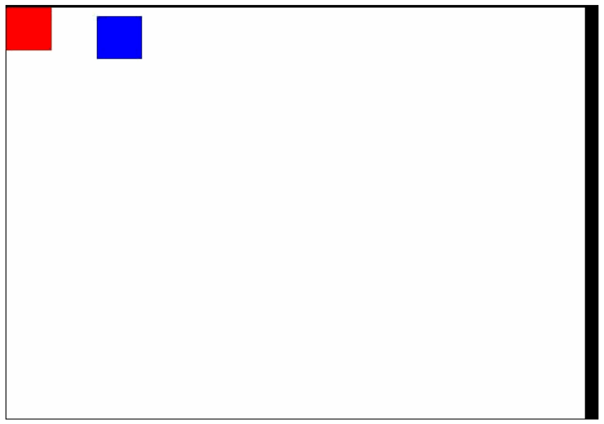

c:/intro-prog-java/mediasources/thisisatest.wav
Play Entire Sound
Play Before
Play After
Stop
Play Selection
Clear Selection
Start Index: N/A Stop Index: N/A
Current Index: 0
Sample Value: 0
The number of samples between pixels: 100
Zoom In

Movie Processing in Java

- Creating animations
- Some simple video processing
  - ***With digital video special effects***

Movies, Animation, and Video … oh my!

- We will refer to captured (recorded) motion as movies
  - Including motion generated by graphical drawings, which is usually called animation
  - And motion generated by some sort of photographic process, usually called video



Psychophysics of Movies

- What makes movies work is persistence of vision
  - We don't notice when we blink
  - Because we retain the last image we saw
  - If not the world would "go away" when we blink
- Have three people observe another person
  - Don’t tell the person what is being counted
  - Count how many times s/he blinks in two minutes
  - Do the people who are counting agree?

Manipulating Movies

- Movies are a series of pictures (frames)
  - Like flip-book animation
- The frame rate is the number of frames shown per second
  - 16 frames per second is the lower bound on our perception of continuous motion
    - Silent movies were 16 fps
    - Later the movie standard became 24 fps to work better with sound
    - Digital video captures 30 frames per second
  - Some people can tell the difference between 30 and 60 frames per second
    - Air force studies show pilots can recognize something in 1/200th of a second

Storing Movies

- One second of a 640 by 480 picture at 30 frames per second (fps) is
  - 640 * 480 * 30 = 9, 216,000 pixels
- Using 24 bit color that means
  - 3 * 9, 216,000 = 27, 648, 000 bytes or over 27 megabytes per second
- For a 90 minute film that is
  - 90 * 60 * 27,648,000 bytes = 149 gigabytes

Compressing Movies

- A DVD only stores 6.47 gigabytes
  - So movies are stored in a compressed format
- Compressed formats
  - MPEG, QuickTime, and AVI
    - Don't record every frame
    - They record key frames and then the differences between the frames
  - JVM records every frame
    - But each frame is compressed

Generating Frame-Based Animations

- We will make movies by
  - Creating a series of JPEG pictures and then displaying them
- Use a FrameSequencer
  - To handle naming and storing the frames
    - Using leading zeros to keep them in order alphabetically
  - And displaying the movie from the frames
    - Using the MoviePlayer class
- Other ways to create a movie from frames
  - Use QuickTime Pro http://www.apple.com/quicktime
  - ImageMagick http://www.imagemagick.org/
  - Windows Movie Maker

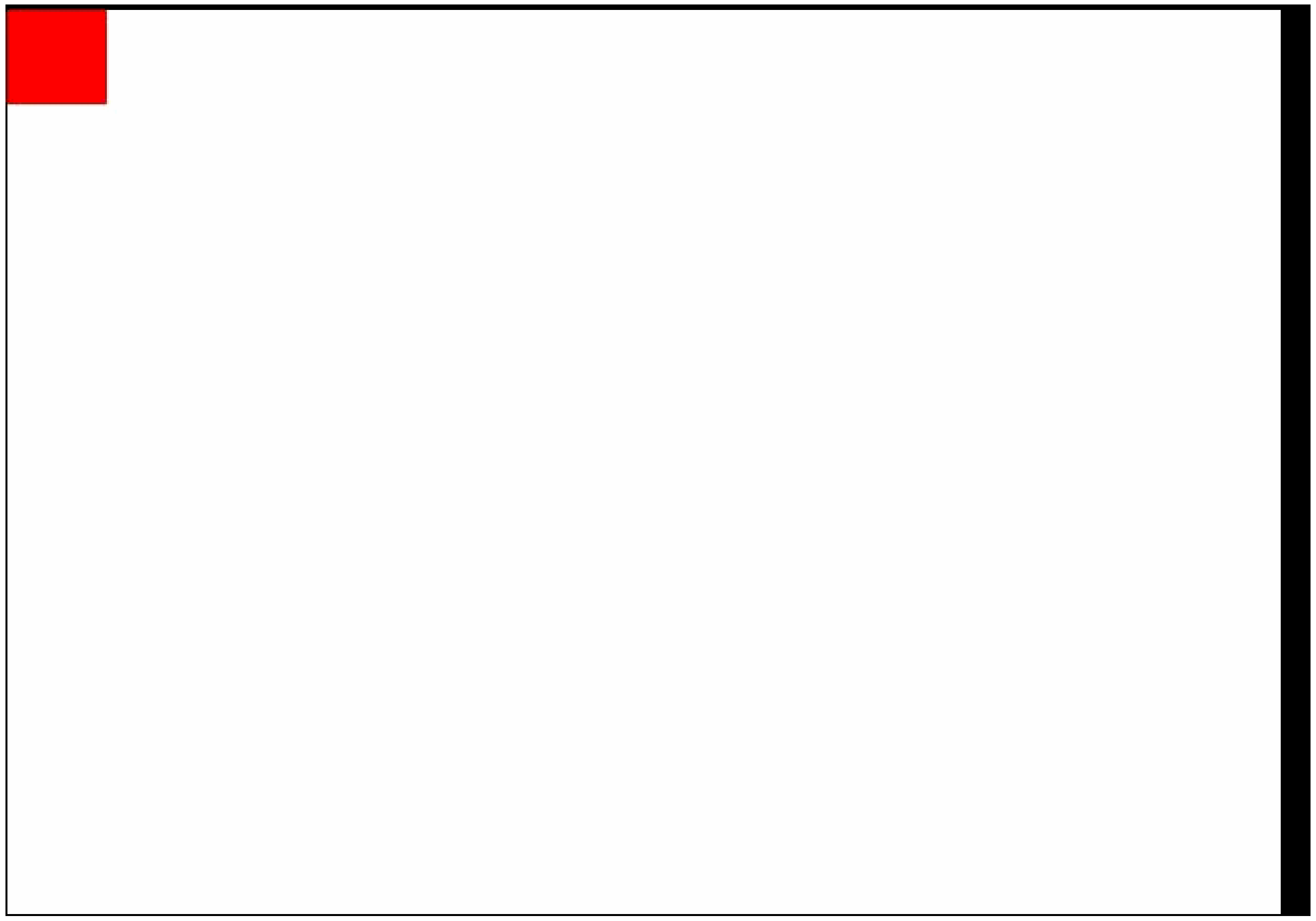
Code for Rectangle Movie (class MovieMaker)

```
public void makeRectangleMovie(String directory)
  {
    int framesPerSec = 30;
    Picture p = null;
    Graphics g = null;
    FrameSequencer frameSequencer =
                                        new FrameSequencer(directory);
    frameSequencer.setShown(true);

    // loop through the first second
    for (int i = 0; i < framesPerSec; i++)
    {
```

Code for Rectangle Movie - Cont

```
      // draw a filled rectangle
      p = new Picture(640,480);
      g = p.getGraphics();
      g.setColor(Color.RED);
      g.fillRect(i * 10, i * 5, 50,50);

      // add frame to sequencer
      frameSequencer.addFrame(p);
    }

    // play the movie
    frameSequencer.play(framesPerSec);
  }
```

Rectangle Movie



MovieMaker Class

- Add the makeRectangleMovie to a new class MovieMaker
- Use the following main to test it
  - Set the directory to any empty directory

```
public static void main(String[] args)
 {
    MovieMaker movieMaker = new MovieMaker();
    String dir = "c:/intro-prog-java/movies/rectangle/";
    movieMaker.makeRectangleMovie(dir);
}
```

Out of Memory Error

- We go through lots of memory when we make movies
  - Java can run out of memory
- You can specify how much memory to use
  - In DrJava click on Edit->Preferences
    - This displays the Preferences Window
      - Select Miscellaneous under Categories
      - Enter –Xmx512m –Xms128m to start with 128 megabytes and set the max to 512 megabytes
      - Click on OK
      - Click on Reset
  - You can specify a maximum that is larger than the amount of RAM in your machine
    - It will swap unused items to the disk (virtual memory)

How the Movie Works

- The key part is

```
g.fillRect(i * 10, i * 5, 50,50);
```

- The rectangle will be drawn at a different location on a blank picture each time
  - And FrameSequencer will write out a file with the resulting picture in it
    - With leading zeros in the name to keep them in order
- The first few calls are

```
g.fillRect(0,0,50,50);
g.fillRect(10,5,50,50);
g.fillRect(20,10,50,50);
g.fillRect(30,15,50,50);
```

Generating a Tickertape Movie

- You can animate text by using drawString
  - At different locations over time on a blank picture
  - drawString("text to draw",baseX,baseY);
- The method drawString will automatically clip the string if it goes off the edge of the picture

Click in the rectangle
to see the
tickertape movie

Code for Tickertape Movie

```
public void makeTickerTapeMovie(String directory,
                                String message)
 {
   int framesPerSec = 30;
   Picture p = null;
   Graphics g = null;
   FrameSequencer frameSequencer =
     new FrameSequencer(directory);
   Font font = new Font("Arial",Font.BOLD,24);

   // loop for 2 seconds of animation
   for (int i = 0; i < framesPerSec * 2; i++)
   {
```

Code for Tickertape Movie - Cont

```
      // draw the string
      p = new Picture(300,100);
      g = p.getGraphics();
      g.setColor(Color.BLACK);
      g.setFont(font);
      g.drawString(message,300 - (i * 10), 50);

      // add frame to sequencer
      frameSequencer.addFrame(p);
    }

    // play the movie
    frameSequencer.play(framesPerSec);
  }
```

Main for Testing

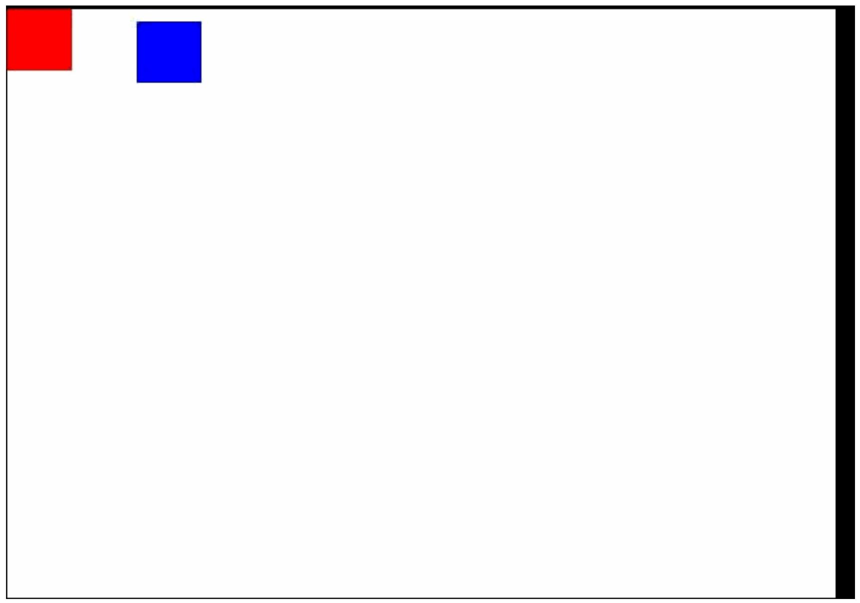
```
public static void main(String[] args)
  {
    MovieMaker movieMaker = new MovieMaker();
    String dir =
       "c:/intro-prog-java/movies/tickertape/";
    movieMaker.makeTickerTapeMovie(dir,
                       "Buy more widgets");
}
```

Exercise

- Create a new method in MovieMaker
- Show text starting at the bottom of a picture and moving to the top
  - You can even change the font size as it moves
- Or you can move the text from top left to bottom right

Moving Two Objects

- You can have more than one object moving in your movie
  - Just draw more than one object in a different location in each frame
  - Here we use the same red rectangle from the previous movie
  - But add a blue rectangle moving in a circular way



Code for Two Rectangle Movie

```
public void makeTwoRectangleMovie(String directory)
 {
   int framesPerSec = 30;
   Picture p = null;
   Graphics g = null;
   FrameSequencer frameSequencer =
     new FrameSequencer(directory);

   // loop through the first second
   for (int i = 0; i < framesPerSec; i++)
   {
```

Code for Two Rectangle Movie - Cont

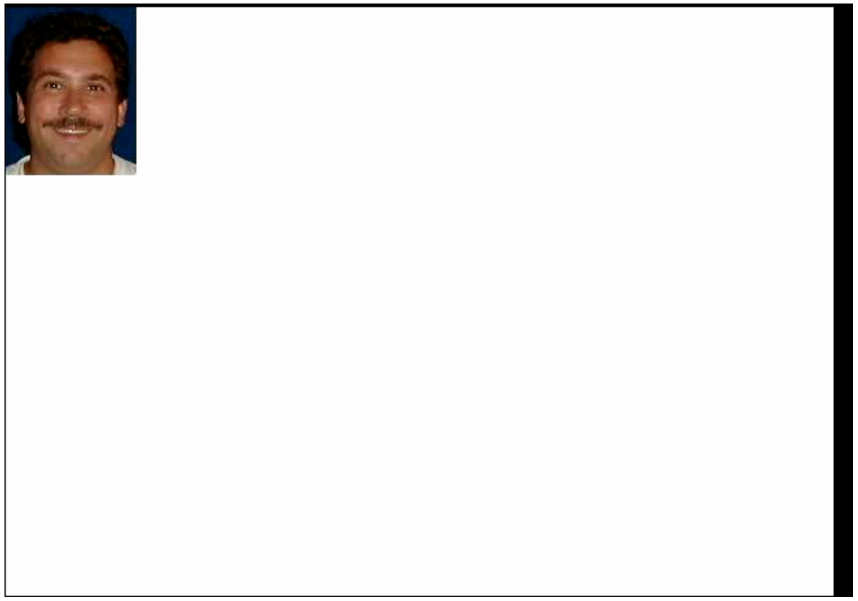
```
      // draw a filled rectangle
      p = new Picture(640,480);
      g = p.getGraphics();
      g.setColor(Color.RED);
      g.fillRect(i * 10, i * 5, 50,50);
      g.setColor(Color.BLUE);
      g.fillRect(100 + (int) (10 * Math.sin(i)),
                 4 * i + (int) (10 * Math.cos(i)),
                 50,50);

      // add frame to sequencer
      frameSequencer.addFrame(p);
    }

    // play the movie
    frameSequencer.play(framesPerSec);
  }
```

Main for Testing

```
public static void main(String[] args)
  {
    MovieMaker movieMaker = new MovieMaker();
    String dir =
              "c:/intro-prog-java/movies/rectangle2/";
    movieMaker.makeTwoRectangleMovie(dir);
}
```

Moving Images in Movies

- You can copy an image to a picture
  - Using our general copy method
    - Or using graphics.drawImage
- Copy the image to different locations
  - In each frame



Code for Move Mark's Head Movie

```
public void moveMarksHead(String directory)
  {
    // load the picture of Mark
    String fName = FileChooser.getMediaPath(
                                            "blue-Mark.jpg");
    Picture markP = new Picture(fName);

    // declare other variables
    Picture target = null;
    FrameSequencer frameSequencer =
      new FrameSequencer(directory);
    int framesPerSec = 30;
```

Code for Move Mark's Head Movie - Cont

```
  // loop creating the frames
  for (int i = 0; i < framesPerSec; i++)
  {
    target = new Picture(640,480);
    target.copy(markP,281,164,382,301,i * 10, i * 5);
    frameSequencer.addFrame(target);
  }

  // play the movie
  frameSequencer.play(framesPerSec);
}
```

Main for Testing

```
public static void main(String[] args)
  {
    MovieMaker movieMaker = new MovieMaker();
    String dir = "c:/intro-prog-java/movies/mark/";
    movieMaker.moveMarksHead(dir);
}
```

Exercises

- Create new methods in MovieMaker
- Make the turtle in turtle.jpg crawl across the beach in beach.jpg
    - Add a new copy method that copies non-white pixels only
- Make the robot in robot.jpg move across the moon in moon-surface.jpg

Reusing Picture Methods

- We can reuse picture methods from previous chapters
  - To create a movie
- To create a sunset movie
  - Create a new makeSunset method
    - That takes as a parameter the amount to reduce the blue and green in the picture
    - Reuse the same picture to accumulate the effect
  - Remember that you can have two methods with the same name
    - As long as the parameter lists are different

Make Sunset Method

```
public void makeSunset(double reduction)
  {
    Pixel[] pixelArray = this.getPixels();
    Pixel pixel = null;
    int value = 0;
    int i = 0;

    // loop through all the pixels
    while (i < pixelArray.length)
    {
```

Make Sunset Method - Cont

```
      // get the current pixel
      pixel = pixelArray[i];

      // change the blue value
      value = pixel.getBlue();
      pixel.setBlue((int) (value * reduction));

      // change the green value
      value = pixel.getGreen();
      pixel.setGreen((int) (value * reduction));

      // increment the index
      i++;
    }
  }
```

Code for Sunset Movie

```
public void makeSunsetMovie(String directory)
  {
    // load the picture of the beach
    String fName = FileChooser.getMediaPath(
                                          "beach-smaller.jpg");
    Picture beachP = new Picture(fName);

    // declare other variables
    Picture target = null;
    FrameSequencer frameSequencer =
      new FrameSequencer(directory);
    int framesPerSec = 30;
```

Code for Sunset Movie - Cont

```
  // loop creating the frames
  for (int i = 0; i < framesPerSec; i++)
  {
    beachP.makeSunset(0.95);
    frameSequencer.addFrame(beachP);
  }

  // play the movie
  frameSequencer.play(framesPerSec);
}
```

Main for Testing

```
 public static void main(String[] args)
  {
    MovieMaker movieMaker =
                        new MovieMaker();
    String dir =
             "c:/intro-prog-java/movies/sunset/";
    movieMaker.makeSunsetMovie(dir);
}
```

Fake Sunset Movie

Fading Out or In

- You can change the threshold on

```
swapBackground(backgroundPicture,
                              newBackPicture,threshold);
```

  - In the loop
  - To swap more background over time
  - You can even make the person in the picture disappear over time

Fade Out Movie

Code for Fade Out Movie

```
public void makeFadeOutMovie(String directory)
  {
    // load the pictures
    String kidF = FileChooser.getMediaPath("kid-in-frame.jpg");
    Picture kidP = null;
    String wallF = FileChooser.getMediaPath("bgframe.jpg");
    Picture wallP = new Picture(wallF);
    String beachF = FileChooser.getMediaPath("beach.jpg");
    Picture beachP = new Picture(beachF);

    // declare other variables
    FrameSequencer frameSequencer =
      new FrameSequencer(directory);
    int framesPerSec = 30;
```

Code for Fade Out Movie - Cont

```
  // loop creating the frames
  for (int i = 0; i < framesPerSec * 2; i++)
  {
    kidP = new Picture(kidF);
    kidP.swapBackground(wallP,beachP,i);
    frameSequencer.addFrame(kidP);
  }

  // play the movie
  frameSequencer.play(framesPerSec);
}
```

Main for Testing

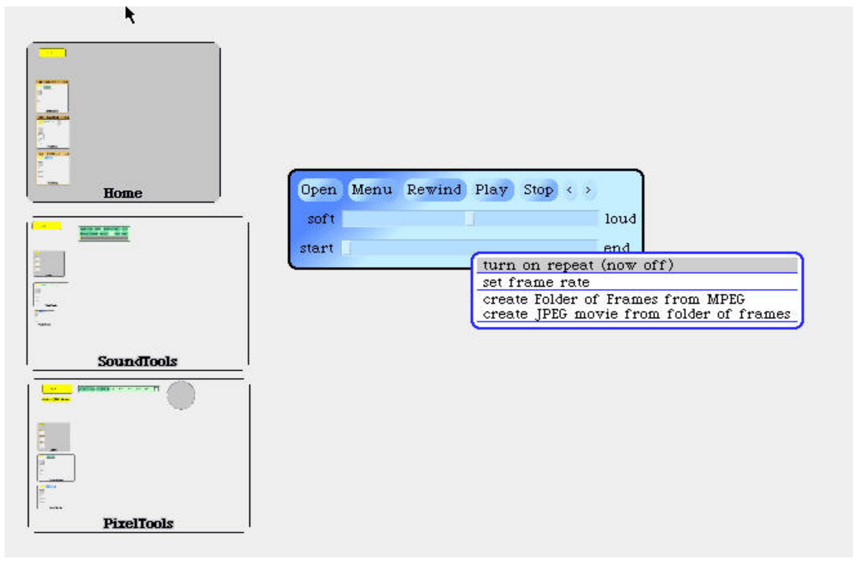
```
 public static void main(String[] args)
  {
    MovieMaker movieMaker =
                          new MovieMaker();
    String dir =
         "c:/intro-prog-java/movies/fade/";
    movieMaker.makeFadeOutMovie(dir);
}
```

Exercise

- Create a new method in MovieMaker
- Make a movie where you increase the amount of edge detection over time which will make the picture disappear over time
  - Be sure to start with the original picture each time

“Bursting” a movie into frames

- You can make a series of JPEG frames from a movie (MPEG)
  - Using MediaTools
    - Click in Video Tools
    - Click on the Menu Button
      - Then click on Create Folder of Frames from MPEG
  - Or use QuickTime Pro



Adding Objects to Movies

- Create a File object on the directory that holds the JPEG frames of the movie
- Get a list of file names in the directory
  - Using list()
- Create a Picture of the image you want to copy
- Loop through all the file names
  - Create a picture from the current file name in the movie
  - Copy into the picture the picture you want to copy
    - Change the location each time through the loop
  - Add the picture to the FrameSequencer

Mommy Watching Katie Dance

Code for Mommy Watching Movie

```
public void makeMommyWatchingMovie(String dir)
 {
   String barbF = FileChooser.getMediaPath("barbaraS.jpg");
   String katieDir =
     FileChooser.getMediaPath("kid-in-bg-seq/");
   Picture barbP = new Picture(barbF);
   FrameSequencer frameSequencer = new FrameSequencer(dir);
   Picture currP = null;

   // get the array of files in the directory
   File dirObj = new File(katieDir);
   String[] fileArray = dirObj.list();
```

Code for Mommy Watching Movie - Cont

```
  // loop through the array of files
  for (int i = 0; i < fileArray.length; i++)
  {
    if (fileArray[i].indexOf(".jpg") >= 0)
    {
      currP = new Picture(katieDir + fileArray[i]);
      currP.copy(barbP,22,9,93,97,i * 3, i * 3);
      frameSequencer.addFrame(currP);
    }
  }

  // play the movie
  frameSequencer.play(30);
}
```

Main for Testing

```
public static void main(String[] args)
 {
    MovieMaker movieMaker =
                          new MovieMaker();
    String dir =
          "c:/intro-prog-java/movies/mommy/";
    movieMaker.makeMommyWatchingMovie(dir);
}
```

Changing the Background of a Movie

- Many movies are shot in front of a blue or green screen
- And then the green or blue is replaced with a different background
  - So that the action looks like it is happening somewhere else
- There is a movie of three kids crawling in front of a blue (well…) screen in the folder kids-blue
  - Use chromakey to put them on the moon

Add Parameters to Chromakey

- The chromakey method will be more reusable
  - If we pass in the color to replace with the new background picture
  - In the kids-blue movie the sheet is blue but without proper lighting it is closer to black
- You can have several methods with the same name in a class
  - As long as the parameter list is different

Chromakey Method

```
public void chromakey(Picture newBg, Color color
                                    double dist)
 {
   Pixel currPixel = null;
   Pixel newPixel = null;

   // loop through the columns
   for (int x=0; x<getWidth(); x++)
   {

     // loop through the rows
     for (int y=0; y<getHeight(); y++)
     {
```

Chromakey Method - Cont

```
        // get the current pixel
        currPixel = this.getPixel(x,y);

        /* if the color at the current pixel is mostly blue
         * (blue value is greater than red and green combined),
         * then use the new background color
         */
        double currDist = currPixel.colorDistance(color);
        if (currDist <= dist)
        {
          newPixel = newBg.getPixel(x,y);
          currPixel.setColor(newPixel.getColor());
        }
      }
    }
  }
```

Code for Kids on Moon Movie

```
public void makeKidsOnMoonMovie(String dir)
  {
    String kidsDir = FileChooser.getMediaPath("kids-blue/");
    String moonF = FileChooser.getMediaPath("moon-surface.jpg");
    Picture moonP = new Picture(moonF);
    FrameSequencer frameSequencer = new FrameSequencer(dir);
    Picture currP = null;

    // get the array of files in the directory
    File dirObj = new File(kidsDir);
    String[] fileArray = dirObj.list();
```

Code for Kids on Moon Movie - Cont

```
  // loop through the array of files
  for (int i = 0; i < fileArray.length; i++)
  {
    if (fileArray[i].indexOf(".jpg") >= 0)
    {
      currP = new Picture(kidsDir + fileArray[i]);
      currP.chromakey(moonP,Color.black,100.0);
      frameSequencer.addFrame(currP);
    }
  }

  // play the movie
  frameSequencer.play(30);
}
```

Main for Testing

```
public static void main(String[] args)
  {
    MovieMaker movieMaker = new MovieMaker();
    String dir = "c:/intro-prog-java/movies/moon/";
    movieMaker.makeKidsOnMoonMovie(dir);
}
```

Kids on the Moon Movie

Exercise

- Create a new method in MovieMaker
- First take a movie with a student doing some action in front of a blue or green screen
    - You can make a screen out of paper
- Use the MediaTools to pull turn the frames into JPEG
- Put the student on the moon or on the beach instead in the movie

Correcting the Color in a Movie

- Movies shot underwater look too blue
  - Water filters out red and yellow light
  - See the images in the fish directory
- Add a new Picture method that will change the red and green values by passed multipliers
  - Yellow is a mixture of red and green
  - Call the new method changeRedAndGreen

Change Red and Green Method

```
public void changeRedAndGreen(double redMult,
                                double greenMult)
 {
   Pixel[] pixelArray = this.getPixels();
   Pixel pixel = null;
   int value = 0;
   int index = 0;

   // loop through all the pixels
   while (index < pixelArray.length)
   {
     // get the current pixel
     pixel = pixelArray[index];
```

Change Red and Green Method - Cont

```
      // change the red value
      value = pixel.getRed();
      pixel.setRed((int) (value * redMult));

      // change the green value
      value = pixel.getGreen();
      pixel.setGreen((int) (value * greenMult));

      // increment the index
      index++;
    }
  }
```

Code for Fish Movie

```
public void makeFishMovie(String dir)
  {
    String movieDir = FileChooser.getMediaPath("fish/");
    FrameSequencer frameSequencer = new FrameSequencer(dir);
    Picture currP = null;

    // get the array of files in the directory
    File dirObj = new File(movieDir);
    String[] fileArray = dirObj.list();
```

Code for Fish Movie - Cont

```
  // loop through the array of files
  for (int i = 0; i < fileArray.length; i++)
  {
    if (fileArray[i].indexOf(".jpg") >= 0)
    {
      currP = new Picture(movieDir + fileArray[i]);
      currP.changeRedAndGreen(2.0,1.5);
      frameSequencer.addFrame(currP);
    }
  }

  // play the movie
  frameSequencer.play(16);
}
```

Main for Testing

```
public static void main(String[] args)
{
  MovieMaker movieMaker = new MovieMaker();
  String dir = "c:/intro-prog-java/movies/fish/";
  movieMaker.makeFishMovie(dir);
}
```

The Fish Movie

Homework Assignment!

- Make a movie!
  - Make at least three things in motion
  - Make it at least 10 seconds long
  - Use any techniques you want.
- Once you have the frames, try generating a movie.
  - Upload to http://home.cc.gatech.edu/gacomputes