
Toys, Techniques, and Tools for Teaching Computer Science

Fran Trees

Drew University

Madison, NJ

Kinesthetic Learning Activities (KLA)

- A pedagogical tool involving physical movement by students.
 - Students are actively, physically engaged
 - This engagement supports some specific learning objective

And the research says...

- KLAs can be effective in practically any learning environment
- No single learning style fits everyone
 - Some students prefer information to be structured as facts about things
 - Some students prefer information to be based on relationships among things
 - Some students prefer a deductive reasoning approach (start with first principles)
 - Some students prefer an inductive reasoning approach (start with examples)

Moral of the story ?

This presentation will include...

- A sharing of ideas for using "toys" in the classroom to support or introduce concepts in teaching computer science
- An introduction to some cool tools to support computer science learning
- A discussion of techniques that can be used in the classroom to support learning

Toys

Know the "WHY" → supports some specific learning objective

The First Day (any CS course)

- Algorithm Development
 - Algorithm development answers the question “**How** can the program accomplish what must be done?”
 - The ability to identify and suggest generalized solutions to particular algorithmic problems can help students find underlying simplicity in difficult problems or, as often happens in computer science, reduce a new problem to one that already has been solved.
- Object-Oriented Programming
 - Learning object oriented design and programming is a challenging task for many beginning students.
 - While objects-first is effective in teaching key concepts of object-oriented programming, it does not go far enough in helping students learn problem solving skills.

Introduction to Algorithm Development

- Peanut and Butter Sandwich

- Materials

- one loaf of bread
 - one jar of peanut butter
 - one jar of jelly
 - one knife
 - one spoon
 - one fork
 - one plate
 - one paper towel

- http://www.cs.duke.edu/courses/cps004games/fall06/classwork/01_pbj/index.html

Introduction to Algorithm Development

- Paper Airplane
 - One sheet of paper. Common notebook paper or stationery should work quite well.
 - The paper used must be rectangular in shape with straight, smooth edges and square (90°) corners.
 - A smooth, flat work surface.
 - OPTIONAL: One pair of scissors or paper shears.

- <http://jdc card.com/engl3007/airplane.htm>

Introduction to Algorithm Development

■ Counting M&Ms

□ Materials

- Bags of M&Ms

- <http://www.cs.duke.edu/courses/cps004/spring01/classwork/comparing.html>

Introduction to Algorithm Development

■ Replicate Drawing

□ Materials

- Paper
- Pencil
- Line Drawing

□ <http://tell.fll.purdue.edu/JapanProj//FLClipart/>

- <http://www.cs.duke.edu/courses/cps004/spring01/classwork/picture/directions.html>



Introduction to Algorithm Development

■ Maps

- **Class will be divided into 4 groups**
- **Materials needed (modify to fit your area):**
 - 4 maps (for Northern New Jersey: 2 South Essex, 2 North Essex) to be used in Part I
 - 4 blank sheets for written street directions (Part I)
 - 4 blank maps: 2 South Essex, 2 North Essex (Part II)
 - 4 highlighters
- http://apcentral.collegeboard.com/apc/members/courses/teachers_corner/45403.html

Introduction of Classes/Objects

- **Classes and Objects**

- **Materials**

- Play-Doh
 - Cookie Cutter
 - Rolling pin

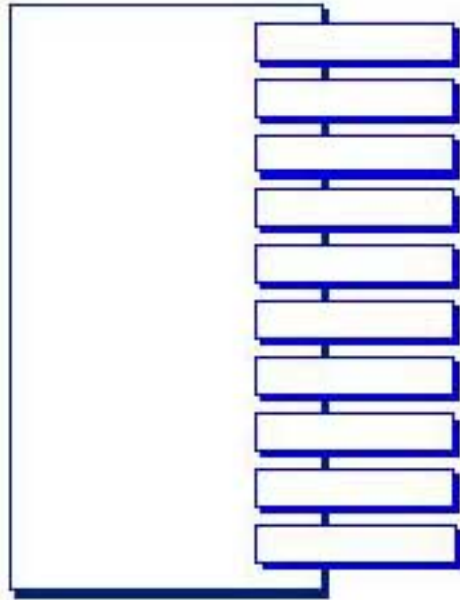
Introduction of Classes/Objects

- Role Play
 - Materials
 - Role Play Script
 - Actors
- <http://www.cs.sbu.edu/dlevine/RolePlay/roleplay.html>

Introduction of Classes/Objects

- Object Diagram

- http://apcentral.collegeboard.com/apc/members/courses/teachers_corner/45405.html



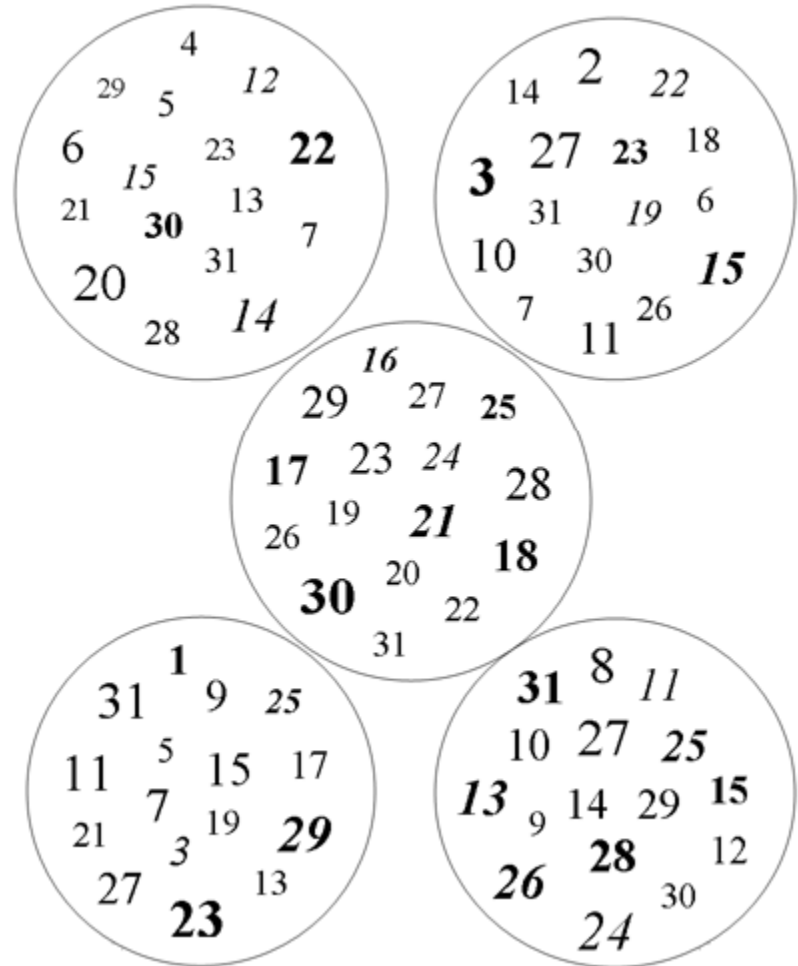
Introduction of Classes/Objects and Inheritance

- Radios

Introducing Number Systems

■ Binary Basics

- http://apcentral.collegeboard.com/apc/members/courses/teachers_corner/45338.html



Algorithm Demonstrations

- Binary Search

- Materials

- Phonebook (or dictionary)

- http://apcentral.collegeboard.com/apc/members/courses/teachers_corner/45426.html

Introduction to `while` loops

Also: $n \log n$ sorts and binary search

■ Paper Folding

- Folding a piece of paper in half results in a folded sheet twice as thick as the original sheet. Similarly, folding this doubled sheet results in a folded sheet four times as thick as the original sheet, and so on. If this process is continued, how many folds would be necessary to obtain a folded sheet whose thickness stretched from the earth to the sun? Assume that a single sheet of paper is .002 inches thick.
- http://apcentral.collegeboard.com/apc/members/courses/teachers_corner/45336.html

Algorithm Demonstration

- **Sorting**

- **Materials**

- Deck of Cards

- or

- Towers of Hanoi (Selection Sort)

- or

- word papers (Merge Sort)

- or

- people

- or

- building blocks

- http://apcentral.collegeboard.com/apc/members/courses/teachers_corner/45476.html

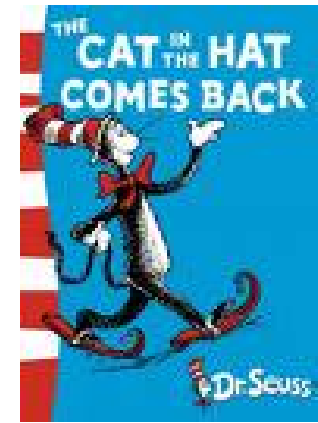
Algorithm Demonstrations

- Sorting
 - Transparencies
 - Animations

Introduction to Recursion

- Cat in the Hat Comes Back

- Dr. Seuss



- Martin and the Dragon

- <http://manaspathak.blogspot.com/2006/06/martin-and-dragon.html>

- David Touretzky's Common List: A Gentle Introduction to Symbolic Logic (Chapter 8)

- <http://www.cs.cmu.edu/~dst/LispBook/index.html>

Computer Science Unplugged

- A collection of activities designed to teach the fundamentals of computer science without requiring a computer.
- **And it's free!**
- <http://csunplugged.com/>

Computer Science Unplugged

- 01 - Binary Numbers
- 02 - Image Representation
- 03 - Text Compression
- 04 - Error Detection
- 05 - Information Theory
- 06 - Searching Algorithms
- 07 - Sorting Algorithms
- 08 - Sorting Networks
- 09 - Minimal Spanning Trees
- 10 - Routing and Deadlock
- 11 - Finite State Automata
- 12 - Programming Languages
- 13 - Graph Colouring
- 14 - Dominating Sets
- 15 - Steiner Trees
- 16 - Information Hiding
- 17 - Cryptographic Protocols
- 18 - Public Key Encryption
- 19 - Human Interface Design
- 20 - The Turing Test

Techniques for Teaching CS

Cool Ideas and Pedagogical Patterns

Techniques for Teaching Computer Science

- Cool Ideas
 - FAQs
 - Company Rules
 - Student Generated Quizzes
 - Golden Rule of Programming
 - Raffle Tickets

Techniques for Teaching Computer Science

- Pedagogical Patterns

- Early Bird
- Spiral
- Consistent Metaphor
- Toy Box
- Tool Box
- Lay of the Land
- Fixer Upper
- Larger Than Life
- Student Design Sprint
- Mistake
- Test Tube
- Fill in the Blanks
- Gold Star
- Grade it Again Sam

- <http://pclc.pace.edu/~bergin/PedPat1.3.html>

Techniques for Teaching Computer Science

- Other cool suggestions
 - Three Stars and a Wish
 - Move Around
 - Look but do not Touch
 - Listen
 - Crafted Questions
 - Learning Contract
 - Piece of Mind

Cool Tools

Jeliot

- Jeliot 3 is a Program Visualization application. It visualizes how a Java program is interpreted. Method calls, variables, operation are displayed on a screen as the animation goes on, allowing the student to follow step by step the execution of a program.
- Programs can be created from scratch or they can be modified from previously stored code examples. The Java program being animated does not need any kind of additional calls, all the visualization is automatically generated.
- Jeliot 3 understands most of the Java constructs and it is able to animate them. Special effort is currently being addressed to animate object oriented features, such as inheritance.

Jeliot tool (Visualization tool)

<http://cs.joensuu.fi/jeliot/>

```
import ProgTools.IOTools;

public class MyClass {
    public static void main() {
        System.out.println("Array_01");
        int[] a = new int[10];
        int[] b = {1,2,3,4,5,6,7,8};
        listForward(a);
        listForward(b);
        listBackward(b);
        System.out.println(a);    //prints something cr
    }

    public static void listForward(int[] aList)
    {
        System.out.println("Array Printed Forward");
        for (int i=0; i < aList.length; i++)
            System.out.println("\t" + i);
        System.out.println();
    }

    public static void listBackward(int[] tList)
    {
        System.out.println("Array Printed Backward");
        for (int x = tList.length-1; x >= 0; x--)
            System.out.println("\t" + tList[x]);
        System.out.println();
    }
}
```

The image shows the Jeliot tool interface with the following components:

- Method Area:** Displays the execution of the `MyClass.listBackward` method. It shows a local variable `int[] tList` and a local variable `int x` with the value 1.
- Expression Evaluation Area:** Shows the evaluation of the expression `2 + 2 = 2`.
- Instance and Array Area:** Contains two data structures:
 - Array 1:** An array of length 8 with values [1, 2, 3, 4, 5, 6, 7, 8].
 - Array 2:** An array of length 10 with values [0, 0, 0, 0, 0, 0, 0, 0, 0, 0].
- Constant Area:** A box labeled "CONSTANTS".

Java Visualizer

- Lightweight Java Visualizer.
 - Information on LJV is located here:
 - <http://www.cs.auckland.ac.nz/~j-hamer/LJV/WhyUseLJV.html>
 - The java file for LJV is located here:
 - <http://www.cs.auckland.ac.nz/~j-hamer/index.html#LJV>
 - You also need graphVIZ another freeware located here:
 - <http://www.research.att.com/sw/tools/graphviz/download.html>

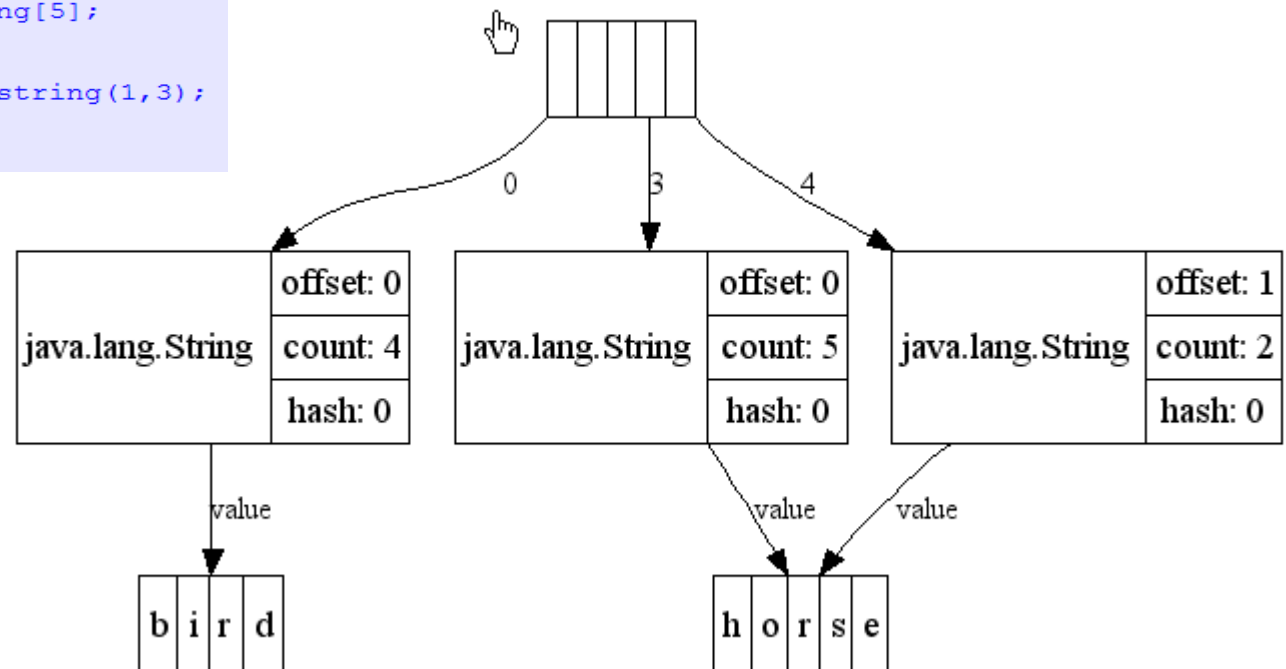
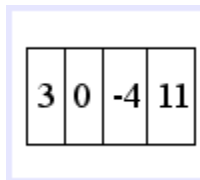
Lightweight Java Visualizer

```
int[] numbers = new int[4];
numbers[0] = 3;
numbers[3] = 11;
numbers[2] = -4;
numbers[1] = 0;

LJV.drawGraph( numbers );

String[] animals = new String[5];
animals[0] = "bird";
animals[3] = "horse";
animals[4] = animals[3].substring(1,3);

LJV.drawGraph( animals );
```



JavaBat

- JavaBat is a free site of live Java coding problems to build coding skill ([example problem](#)), created by [Nick Parlante](#) who is computer science lecturer at Stanford. The coding problems give immediate feedback, so it's an opportunity to practice and solidify understanding of the concepts.
- The problems could be used as homework, or for self-study practice, or in a lab, or as live lecture examples. The problems, all listed off the [JavaBat home](#), have low overhead: short problem statements (like an exam) and immediate feedback in the browser.
- <http://www.javabat.com/about.html>

Javabat

The parameter `weekday` is true if it is a weekday, and the parameter `vacation` is true if we are on vacation. We sleep in if it is not a weekday or we're on vacation. Return true if we sleep in.





`sleepIn(false, false) → true`
`sleepIn(true, false) → false`
`sleepIn(false, true) → true`
`sleepIn(true, true) → true`

Show Solution

```
public boolean sleepIn(boolean weekday, boolean vacation) {  
    return (!weekday || vacation);  
}
```

Go

...Save, Compile, Run

Expected	This Run	
<code>sleepIn(false, false) → true</code>	true	PASS 
<code>sleepIn(true, false) → false</code>	false	PASS 
<code>sleepIn(false, true) → true</code>	true	PASS 
<code>sleepIn(true, true) → true</code>	true	PASS 



All Correct

[next](#) | [chance](#)

JavaBat > [Warmup](#)

[done page](#)

Code is saved so long as this session is active. [Create an account](#) to save code past this session.

JavaBat

Warmup

Simple problems to get started (solutions avail)

[✓sleepIn](#) [H](#) [✓monkeyTrouble](#) [H](#) [✓sumDouble](#) [H](#) [more](#)

String1

Basic string problems -- no loops

[✓helloName](#) [✓makeAbba](#) [✓makeTags](#) [more](#)

Logic

Little boolean logic puzzles -- if else && || !

[✓cigarParty](#) [H](#) [✓dateFashion](#) [H](#) [✓squirrelPlay](#) [more](#)

String2

Medium String problems -- 1 loop

[✓doubleChar](#) [H](#) [✓countHi](#) [H](#) [✓catDog](#) [more](#)

Array2

Medium array problems -- 1 loop

[✓countEvens](#) [✓bigDiff](#) [✓centeredAverage](#) [more](#)

Recur1

Basic recursion problems

[✓factorial](#) [H](#) [✓bunnyEars](#) [H](#) [✓fibonacci](#) [more](#)

Warmup2

Simple string and array loops (solutions avail)

[✓stringTimes](#) [H](#) [✓frontTimes](#) [H](#) [✓stringBits](#) [H](#) [more](#)

Array1

Basic array problems -- no loops

[✓firstLast6](#) [✓sameFirstLast](#) [✓makePi](#) [more](#)

AP1

AP CS practice problems

[✓scoresIncreasing](#) [✓scores100](#) [✓scoresClump](#) [more](#)

String3

Harder String problems -- 2 loops

[✓countYZ](#) [✓withoutString](#) [✓equalIsNot](#) [more](#)

Array3

Harder array problems -- 2 loops, more complex logic

[✓maxSpan](#) [✓fix34](#) [✓fix45](#) [more](#)

Recur2

Harder recursion problems

[✓groupSum](#) [H](#) [✓groupSum6](#) [✓groupNoAdj](#) [more](#)

Introduction to Computer Science

- Alice
- Karel J. Robot
- Jeroo
- Greenfoot
- GridWorld
- Scratch
- Squeak
- Phrogram
- Lego Mindstorms

Alice

- Alice is an innovative 3D programming environment that makes it easy to create an animation for telling a story, playing an interactive game, or a video to share on the web.
- Alice is a **freely available** teaching tool designed to be a student's first exposure to object-oriented programming.
- www.alice.org

Alice



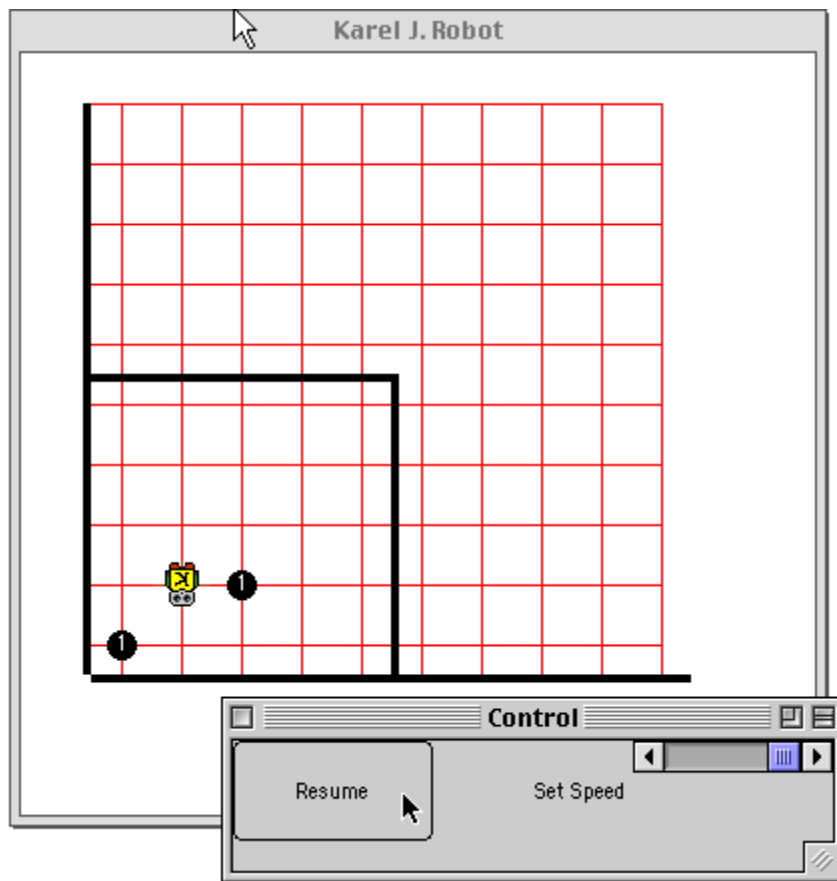
World.my first method
World.my first method *No parameters*
No variables

- IceSkater.prepare to skate
- IceSkater.do simple spin

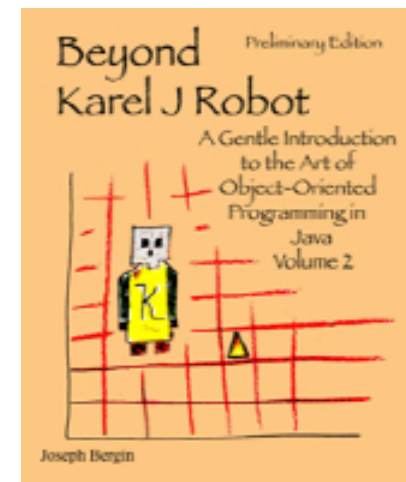
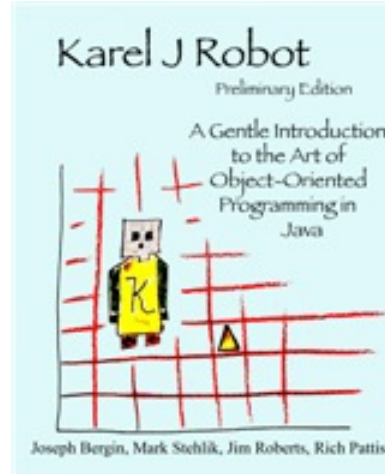
World.my first method **IceSkater.do simple spin**
IceSkater.do simple spin *No parameters*
No variables

- Do together**
 - IceSkater set pose IceSkater.pose3 more...
 - Do in order**
 - Wait 0.25 seconds
 - IceSkater turn right 2.25 revolutions more...
 - IceSkater set pose IceSkater.pose more...

Karel J. Robot (Java)



- A Gentle Introduction to Object-Oriented Programming



Jeroo

- Jeroo is a tool that helps novices learn fundamental concepts of object-oriented programming, including Instantiating and using objects
 - Writing methods to extend behavior
 - Selecting and using fundamental control structures
 - Jeroo engages students with
 - Story telling
 - Animated execution
 - Simultaneous code highlighting

Greenfoot (Java)

- Consider Greenfoot as a combination between a framework for creating two-dimensional grid assignments in Java and an integrated development environment (class browser, editor, compiler, execution, etc.) suitable for novice programmers.
- While Greenfoot supports the full Java language, it is especially useful for programming exercises that have a visual element. In Greenfoot, object visualization and object interaction are the key elements.

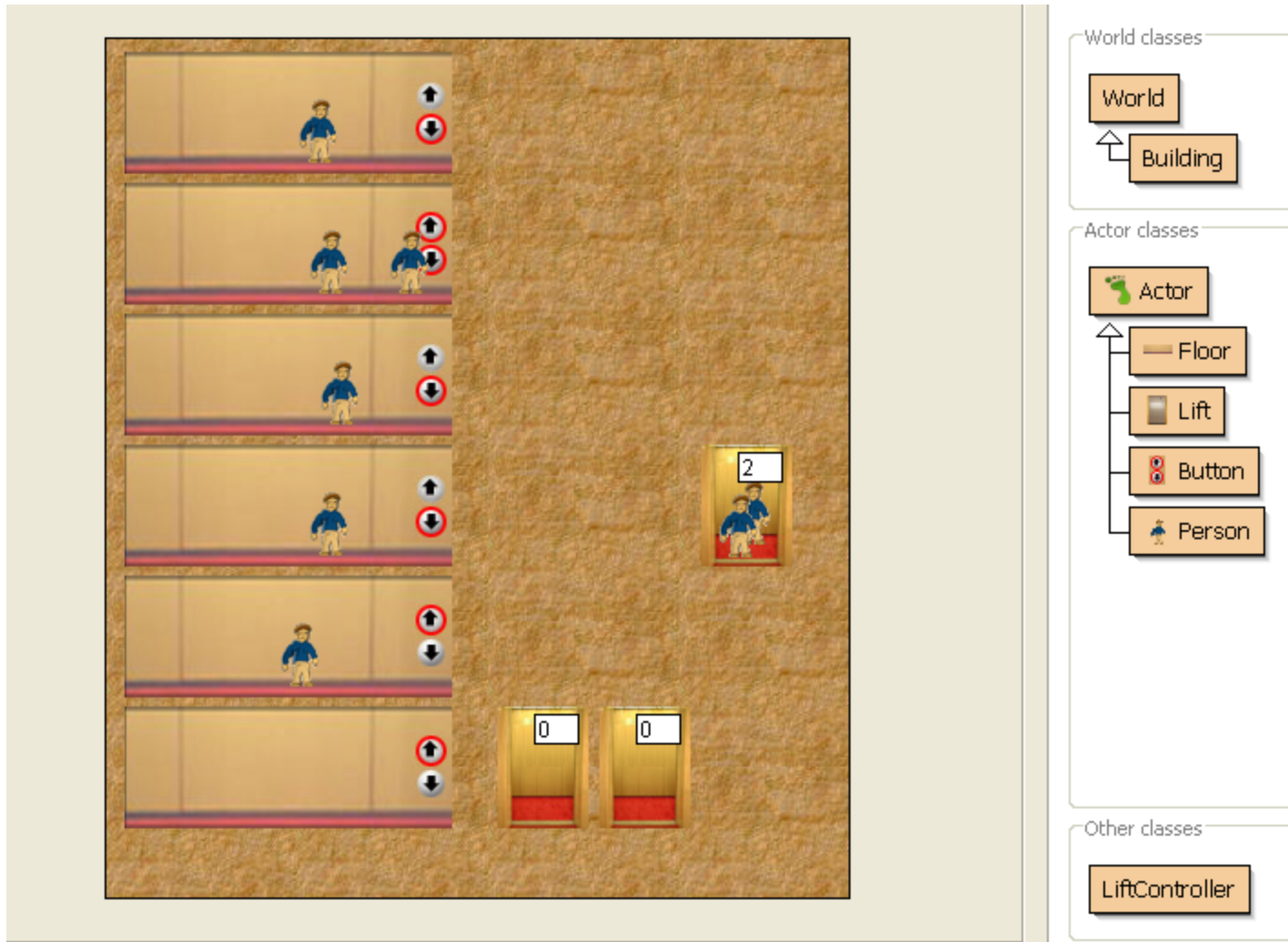
Greenfoot

The screenshot displays the Greenfoot IDE interface for a project named "Greenfoot: wombats". The main workspace shows a 6x6 grid representing the "WombatWorld". The grid contains several objects: a Wombat in the center, another Wombat in the bottom-left, several Leaves scattered throughout, and several Rocks. A mouse cursor is positioned over the top-right cell of the grid.

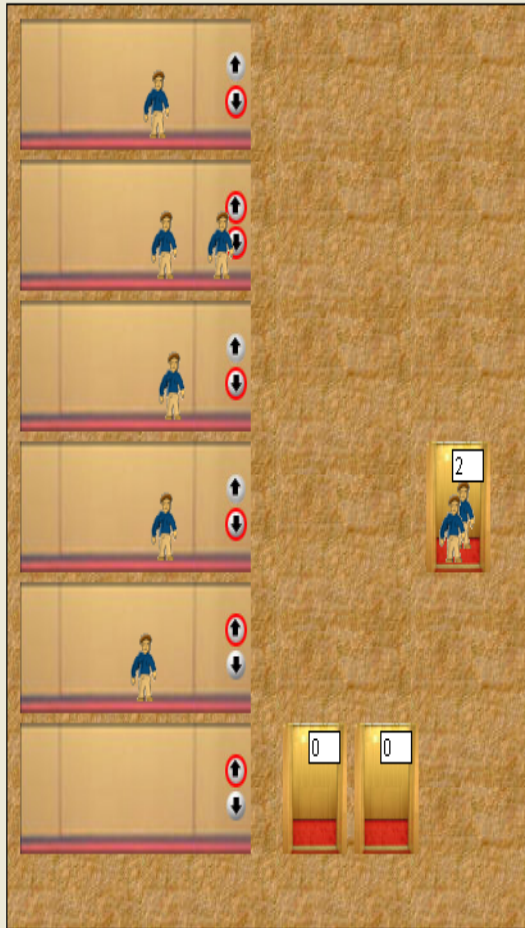
On the right side, there is a "Project Information" tab and a class hierarchy panel. The "World classes" section shows a hierarchy where "WombatWorld" inherits from "World". The "Actor classes" section shows a hierarchy where "Leaf", "Wombat", and "Rock" all inherit from "Actor". A small Rock icon with a red 'X' is visible at the bottom of the class hierarchy panel.

At the bottom of the IDE, there are control elements: "Act" and "Run" buttons, a "Speed:" slider, and a "Compile All" button.

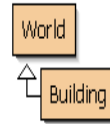
Greenfoot



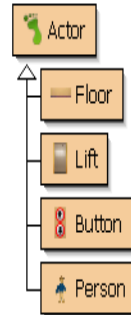
Greenfoot



World classes



Actor classes



Other classes



```
import greenfoot.World;
import greenfoot.Actor;
```

```
public class Person extends Actor
{
```

```
    private static final int ST_ON_FLOOR = 0;
    private static final int ST_IN_LIFT = 1;
```

```
    private Floor originFloor;
    private int targetFloor;
    private int status;
```

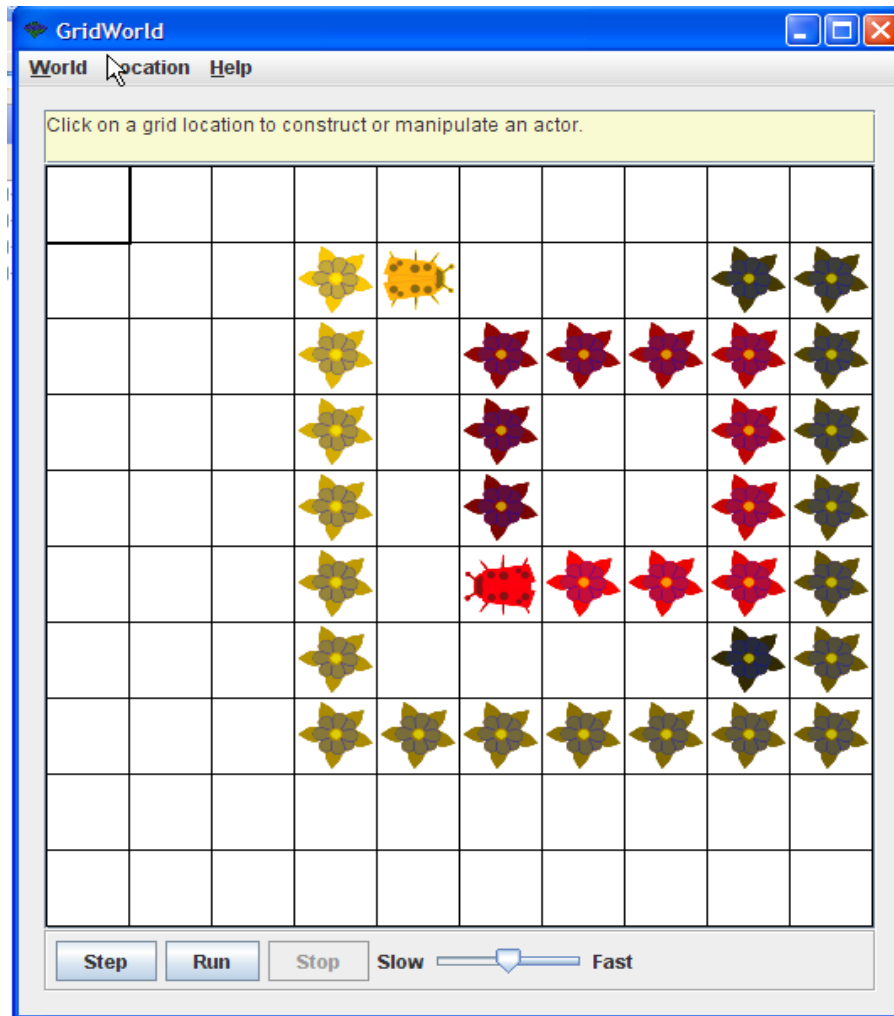
```
    public Person()
    {
        this(null, null);
    }
```

```
    public Person(Floor floor, Building building)
    {
        setImage("person.gif");
        originFloor = floor;
        status = ST_ON_FLOOR;
        if ((floor != null) && (building != null))
```

GridWorld (Java)

- GridWorld is the case study for the 2008 AP CS exam (and beyond). GridWorld uses an engaging environment that allows students to create and test actors with a wide variety of behaviors.

GridWorld

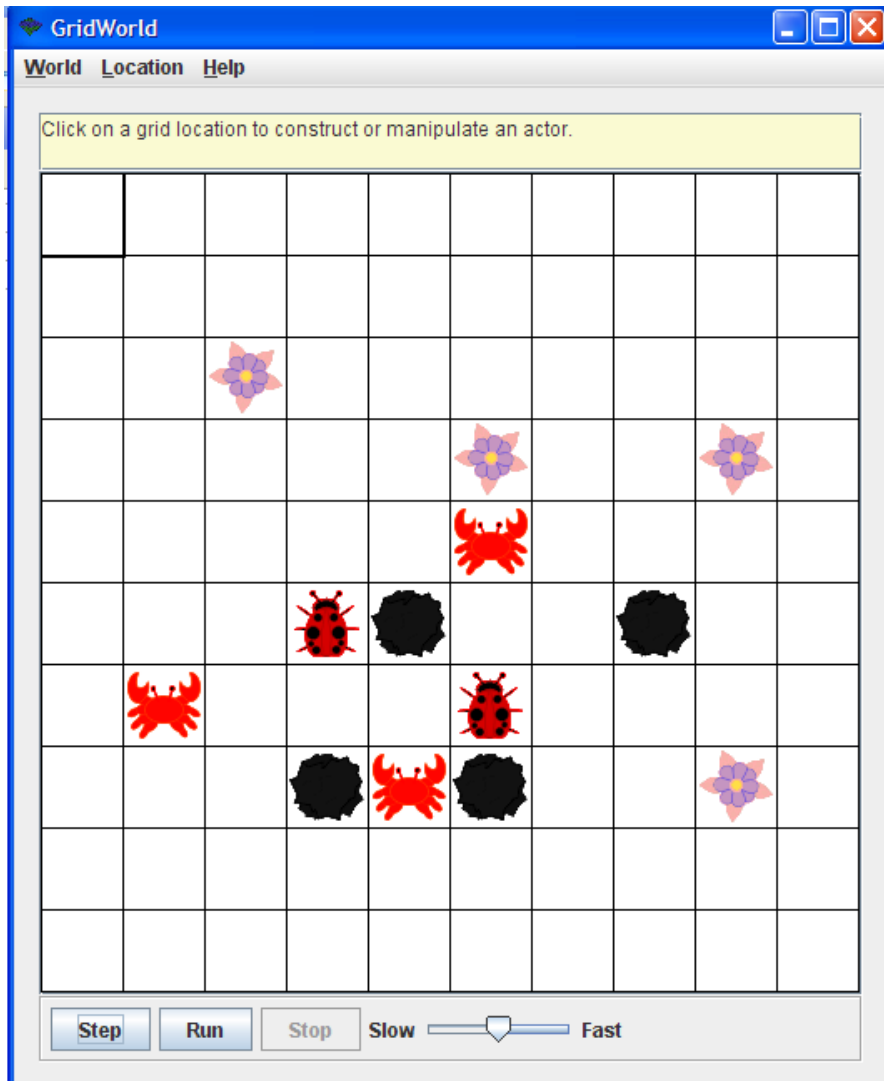


```
public class BoxBug extends Bug
{
    private int steps;
    private int sideLength;

    /**
     * Constructs a box bug that traces a square of a given side length
     * @param length the side length
     */
    public BoxBug(int length)
    {
        steps = 0;
        sideLength = length;
    }

    /**
     * Moves to the next location of the square.
     */
    public void act()
    {
        if (steps < sideLength && canMove())
        {
            move();
            steps++;
        }
        else
        {
            turn();
            turn();
            steps = 0;
        }
    }
}
```

GridWorld

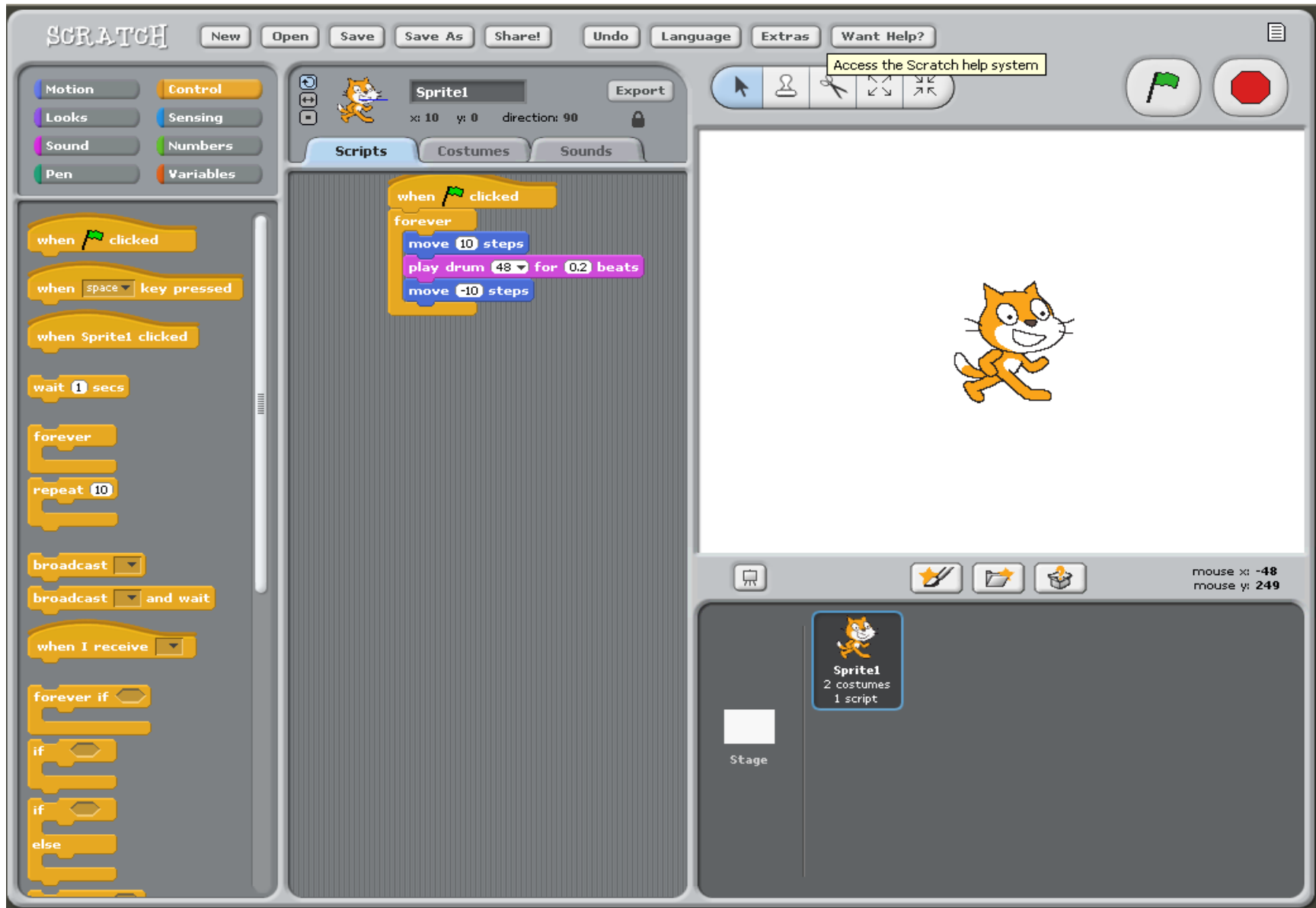


```
public class CrabRunner
{
    public static void main(String[] args)
    {
        ActorWorld world = new ActorWorld();
        world.add(new Location(7, 5), new Rock());
        world.add(new Location(5, 4), new Rock());
        world.add(new Location(5, 7), new Rock());
        world.add(new Location(7, 3), new Rock());
        world.add(new Location(7, 8), new Flower());
        world.add(new Location(2, 2), new Flower());
        world.add(new Location(3, 5), new Flower());
        world.add(new Location(3, 8), new Flower());
        world.add(new Location(6, 5), new Bug());
        world.add(new Location(5, 3), new Bug());
        world.add(new Location(4, 5), new CrabCrawler());
        world.add(new Location(6, 1), new CrabCrawler());
        world.add(new Location(7, 4), new CrabCrawler());
        world.show();
    }
}
```

Scratch

- Scratch is a new programming language that makes it easy to create your own interactive stories, animations, games, music, and art -- and share your creations on the web.
- Scratch is designed to help young people (ages 8 and up) develop 21st century learning skills. As they create Scratch projects, young people learn important mathematical and computational ideas, while also gaining a deeper understanding of the process of design.
- Scratch is available free of charge
- <http://scratch.mit.edu/>

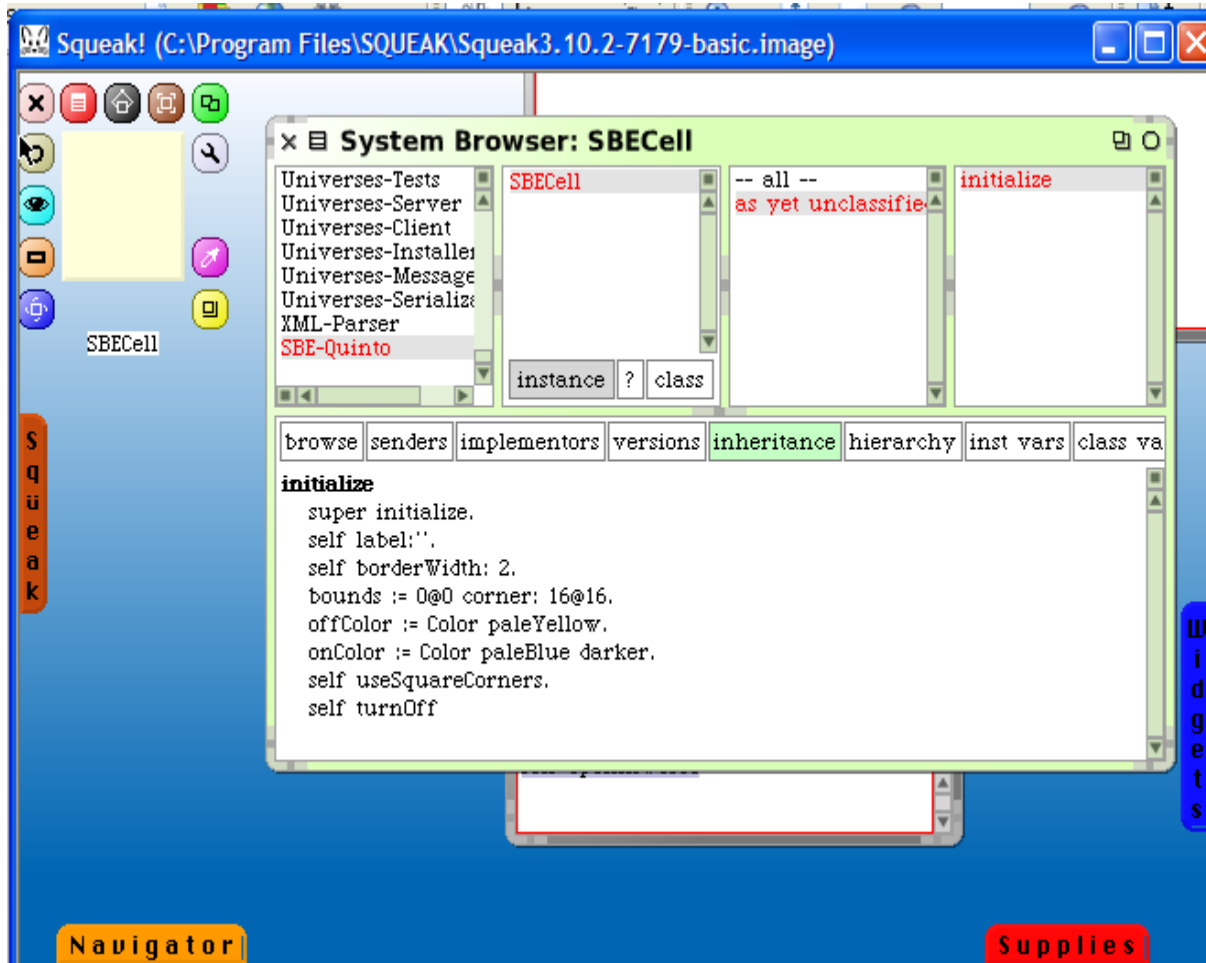
Scratch



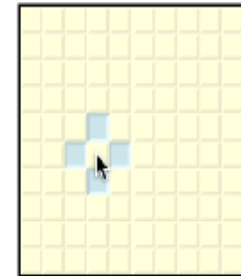
Squeak (Smalltalk-OO language)

- Squeak is a modern, open source, fully-featured implementation of the Smalltalk programming language and environment.
- Squeak is highly portable — even its virtual machine is written entirely in Smalltalk, making it easy to debug, analyze, and change.
- Squeak is the vehicle for a wide range of innovative projects from multimedia applications and educational platforms to commercial web development environments.
- <http://www.squeak.org/>

Squeak



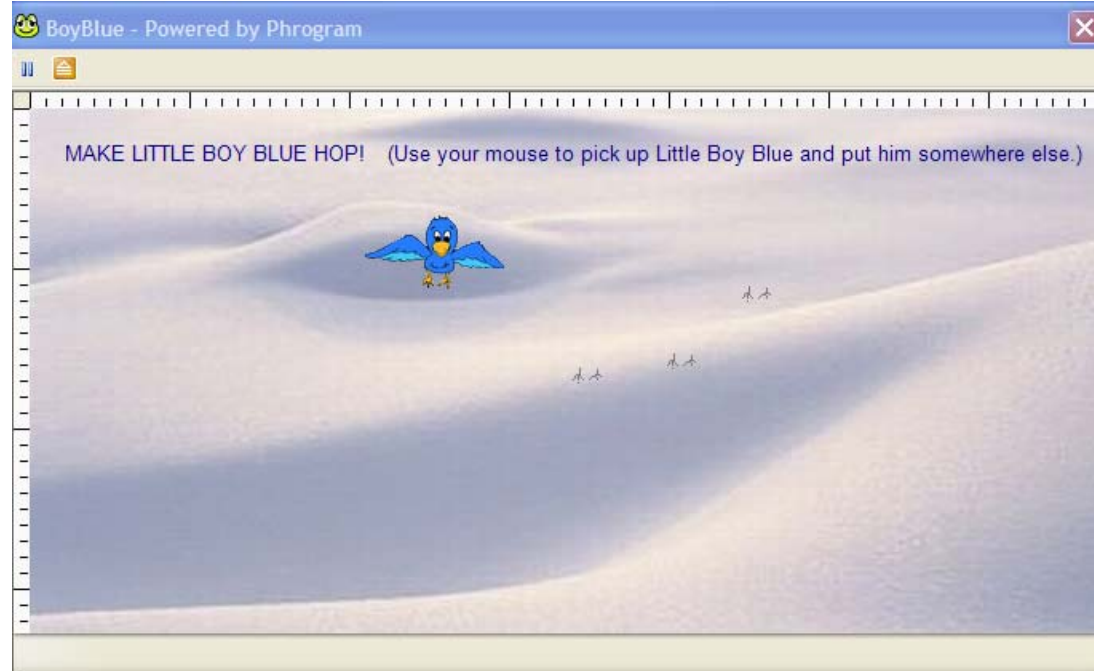
The Quinto game



Phrogram

- Phrogram is a programming language and integrated development environment, or IDE, bearing some similarities to Visual Basic.
- A Phrogram program is a collection of nested code blocks. On the highest level is a *Program* block, and within this *Method* blocks and *Function* blocks are defined. Functions and Methods are both chunks of reusable code, available in the Program scope; Functions return values, while Methods may not. Data structures are defined within the Program scope. Variables must be declared and typed at the time of declaration.
- <http://phrogram.com/>

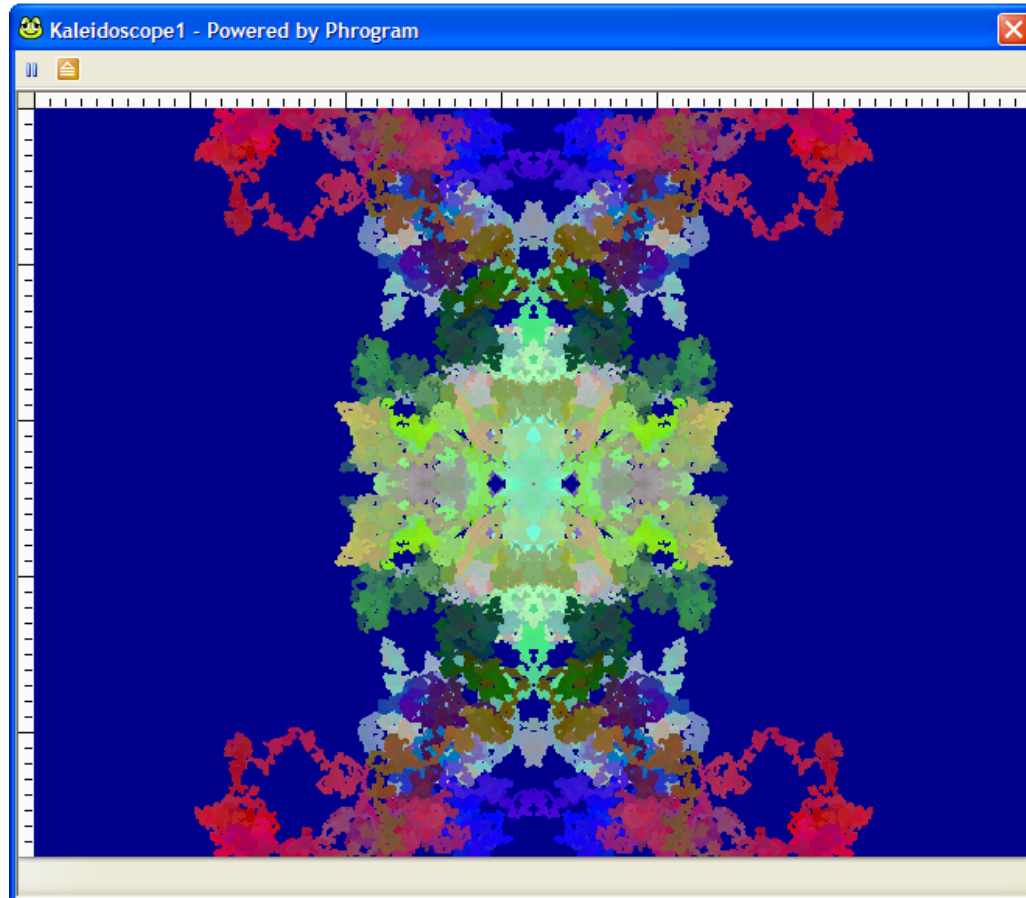
Phrogram



```
While True
```

```
    // If the user presses a mouse button, see if it's on the
    // bird. If so, start dragging.
    If MouseEvent = "ButtonDown" Then
        If Not Dragging Then
            // If we've just started dragging, turn Dragging on
            // and calculate where the mouse is relative to the
            // top left corner of the bird.
            Dragging = Bird.ContainsPoint ( MouseX, MouseY )
            If Dragging Then
                OffsetX = MouseX - Bird.Left
                OffsetY = MouseY - Bird.Top
            End If
        End If
    End If
End If
```

Phrogram



```
Function ChangeColor ( Tint As Integer ) As Integer
// ChangeColor changes one color component randomly by -1, 0, or 1 but
// keeps it between 0 and 255.
    Tint = Tint + Random ( -1, 1 )
    Tint = Max ( Tint, 0 )
    Tint = Min ( Tint, 255 )
    Return tint
End Function
```

Lego Mindstorms

- Kits are available from www.legoeducation.com . You can use java to program the

Others????

References (where links not provided)

- Paolo A. G. Sivilotti and Scott M. Pike. A collection of kinesthetic learning activities for a course on distributed computing: ACM SIGACT news distributed computing column 26. SIGACT News, 38(2):56–74, 2007.