



Programming with Scratch

Harvey Siy

hsiy@unomaha.edu

<http://www.cs.unomaha.edu/~hsiy>





Outline



-
- Computational thinking
 - Introduction to Scratch
 - Short exercise
 - A more complicated example
 - Discussions



Computational Thinking is...



A problem solving approach...

... solving a problem by
explaining the steps
needed to arrive at the solution.

To explain the steps, you have to:

- come up with the steps
- communicate them
- defend them



Explaining the steps



- come up with the steps
 - involves *creativity*
 - involves *recognizing* similar situations
- communicate the steps
 - how can I *express* them clearly?
 - involves creating an algorithm*
 - forms the basis of computer programming
- defend the steps
 - are these the *right* steps?
 - will these lead to a *correct* solution?
 - involves *logical reasoning* and *critical thinking*

* *step-by-step procedure
for solving a problem*



Teaching Computational Thinking



- Coding is the most fun way to practice computational thinking.
- Lots of resources for all grade levels from Hour of Code:
<http://code.org/learn>

MIT App Inventor

AppInventor Hour of Code
MIT Center for Mobile Learning @ The Media

Create a holiday card
Scratch

Write your first computer program
Code.org

Learn the basic concepts of Computer Science with drag and drop programming. This is a game-like, self-directed tutorial starring video lectures by Bill Gates, Mark Zuckerberg, Angry Birds and Plants vs. Zombies. Learn repeat-loops, conditionals, and basic algorithms. Available in 20 languages

Ages 6-106 | Modern browsers, smartphones, tablets

11,696,752 participants

<http://hourofcode.com/co>
Teacher's Notes

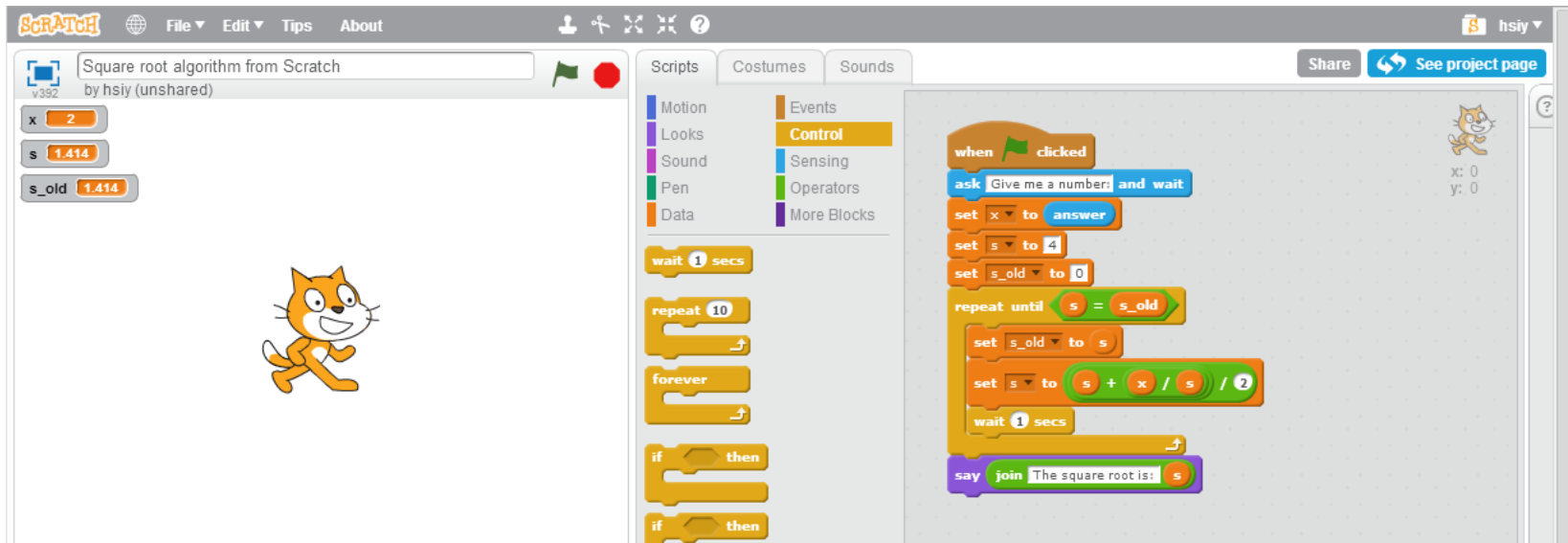
Go



Scratch



- <http://scratch.mit.edu>
- Fun and easy to learn
- Develops creativity





Development Environment





Development Environment



Scripts | Costumes | Sounds

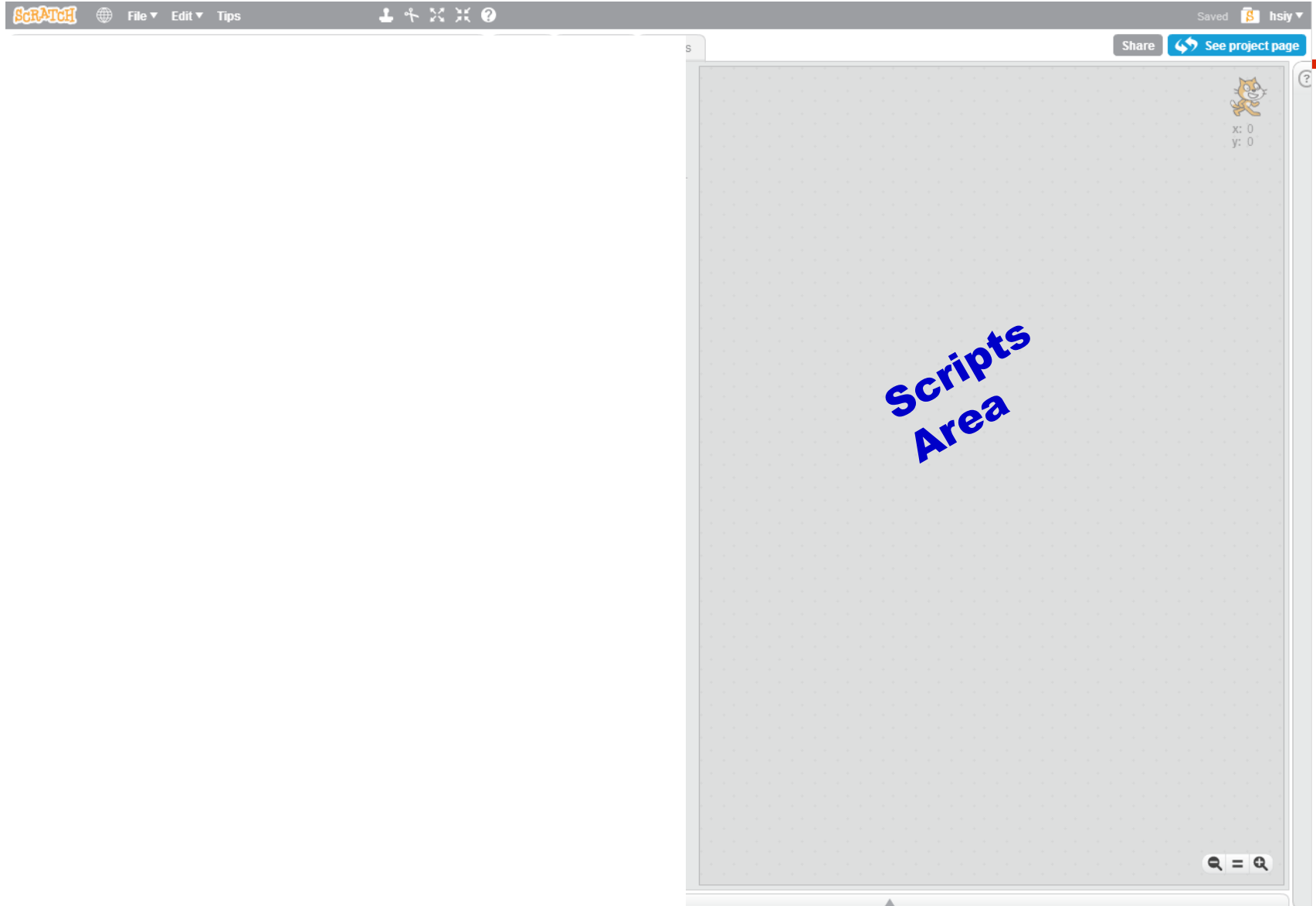
Motion | Events
Looks | Control
Sound | Sensing
Pen | Operators
Data | More Blocks

move 10 steps
turn 15 degrees
turn 15 degrees
point in direction 90
point towards
go to x: 0
go to x: 0 y: 0
change x by 10
set x to 0
change y by 10
set y to 0
if on edge, bounce
set rotation style left-right

x position
 y position
 direction

Block
Palette

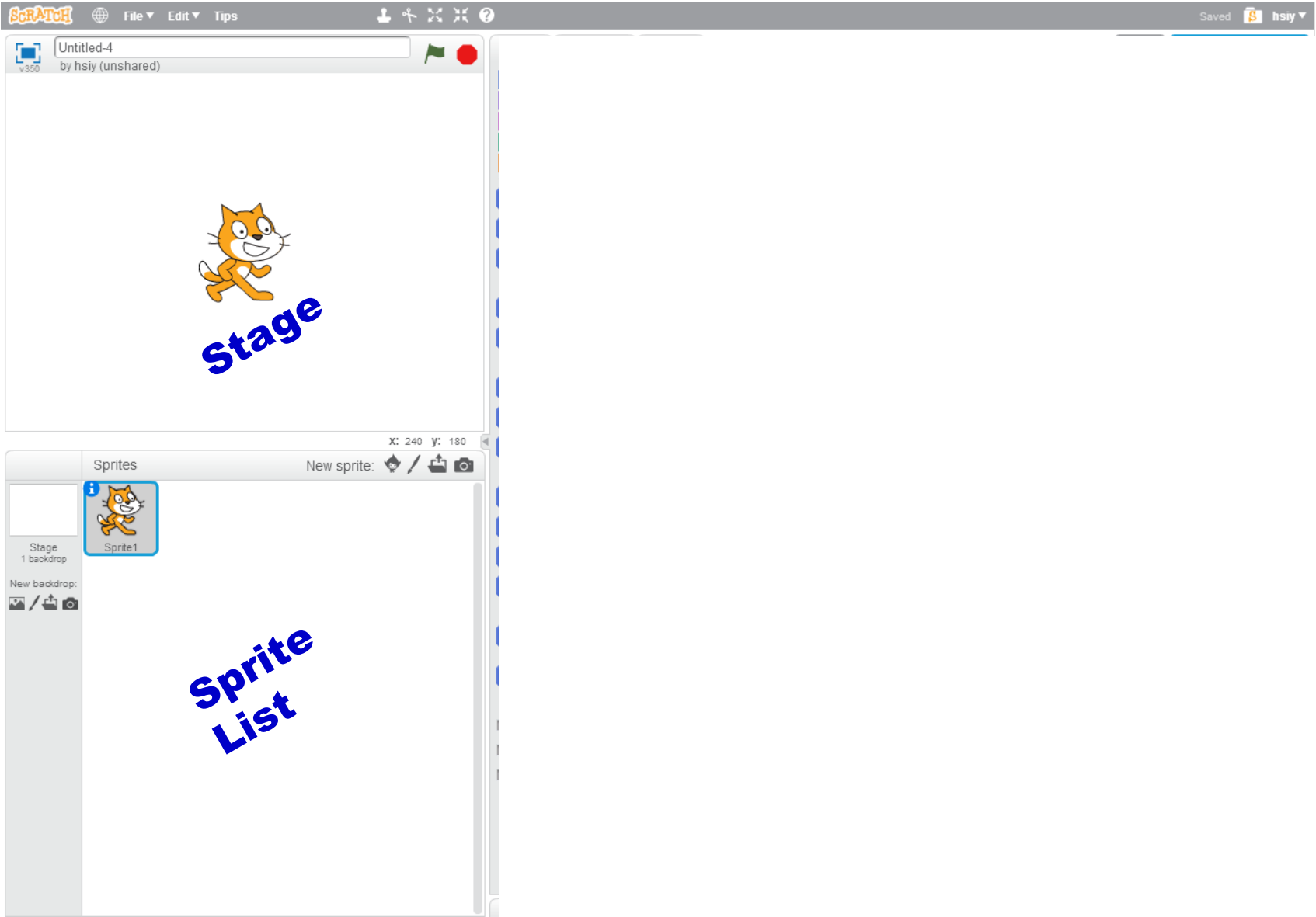
Development Environment



The image shows a screenshot of the Scratch development environment. The interface includes a top menu bar with 'Scratch', 'File', 'Edit', and 'Tips'. A toolbar with various icons is located below the menu. The main workspace is a large grey area with a grid pattern. In the center of the workspace, the text 'Scripts Area' is written in a large, blue, slanted font. In the top right corner of the workspace, there is a small orange cat icon (Scratch's mascot) and the coordinates 'x: 0' and 'y: 0'. The top right of the interface features a 'Share' button and a 'See project page' button. The bottom right corner of the workspace has a small search icon and a refresh icon.

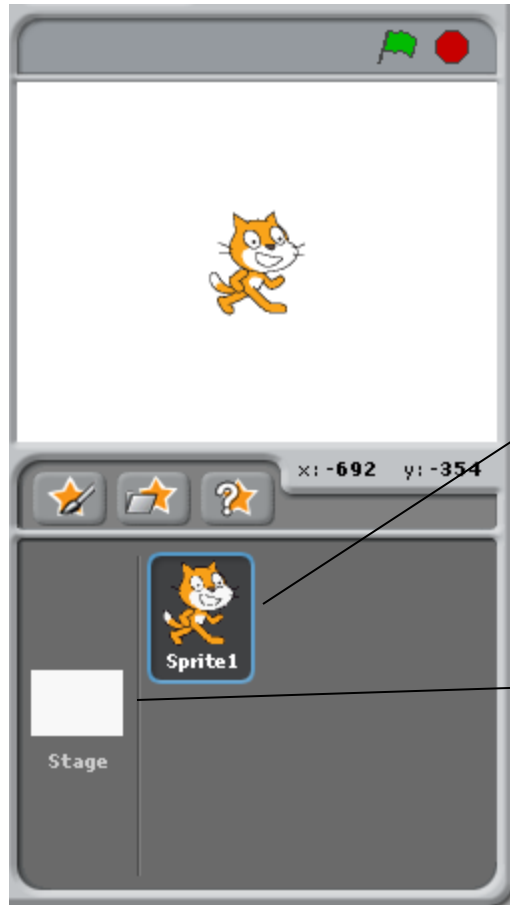


Development Environment





Sprite



The stage and sprites can have scripts.

Sprite

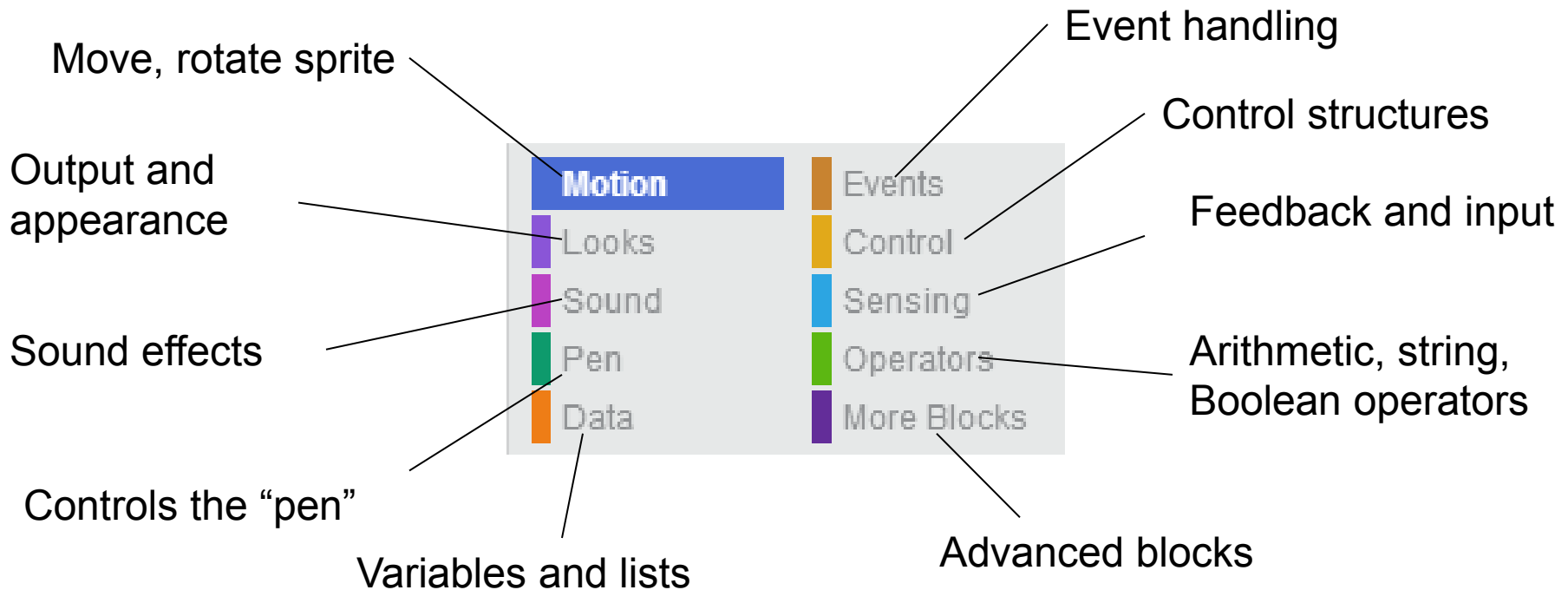
- An object that performs some action
- Can define “indefinite” number of sprites in a program

Stage

- A special sprite
- 480x360
- Always in the background



Categories of blocks





Observations (subject to change)



- Real numbers are 64-bit floating point.
- Integers appear to be unbounded.
- Lists can only have numbers, strings, Booleans.
 - No list of lists.
 - No list of sprites.
- Recursion is not allowed
 - Except tail recursion.



Flow of execution



-
- Sequence – one instruction followed by the next
 - Decisions – compute a condition
 - if true, follow one set of instructions
 - if false, follow an alternative set of instructions
 - Loops – allows a set of instructions to be executed repeatedly
 - Events – determines when to start an execution



Example 1: Computing square roots



- Use algorithm by Hero of Alexandria

To compute square root of x

1. Start with arbitrary positive value s
2. Replace s by $(s + x/s)/2$
3. Repeat #2 until s has stopped changing

Solution

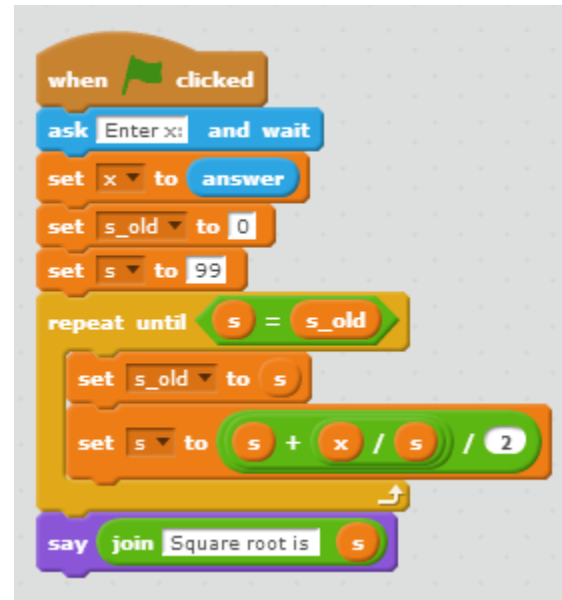


To compute square root of x

1. Start with arbitrary positive value s
2. Replace s by $(s + x/s)/2$
3. Repeat #2 until s has stopped changing



Need to
make
variables



Access the working version at:

<http://scratch.mit.edu/projects/11656266/>



Observations from Example 1



-
- Can make something other than games or stories (for more examples, see also <http://code.google.com/p/scratch-unplugged/>)
 - Students can see the order of operations in expressions by the 3-D treatment of the operation templates.

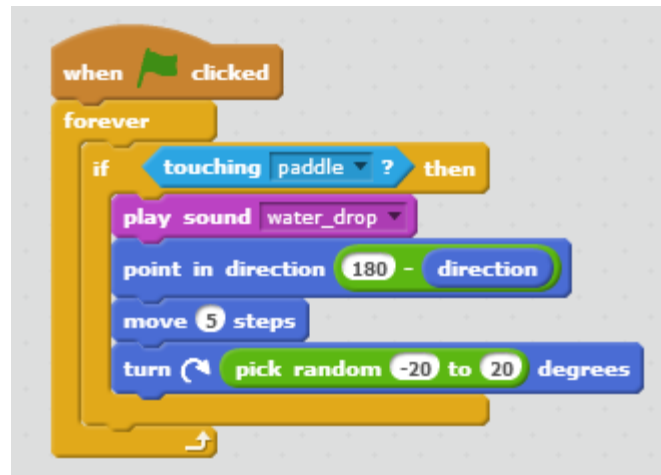
Example 2: Pong



Move ball



When paddle hits ball



When paddle misses



Access the working version at:

<http://scratch.mit.edu/projects/10597215/>



Example 3: Sorting



-
- Download the Scratch programs from <http://code.google.com/p/scratch-unplugged/>
 - Find the selection sort program.
 - Gives example of:
 - Multiple sprites
 - Event handling
 - Lists



Physical interactions



-
- Scratch offers additional forms of interaction beyond the traditional keyboard and mouse.
 - Sensor boards
 - Picoboard (currently 1.4)
 - Makey Makey
 - Lego WeDo (currently 1.4)
 - Kinect (currently 1.4)
 - Webcams (2.0 only)



Example 4: Webcam interaction



```

when green flag clicked
  turn video on
  set x to 0
  set y to -150
  forever loop
    if video motion on this sprite > 20 then
      change y by 20
      next costume
    if video motion on this sprite < 5 then
      if y position > -150 then
        change y by -10
    wait 0.1 secs
  
```

Detect motion by webcam

“Faster” motion detected: move sprite up

“Slower” motion detected: move sprite down

```

when space key pressed
  turn video off
  
```

Remember to turn off webcam when done!

Access the working version at:

<http://scratch.mit.edu/projects/10673482/>



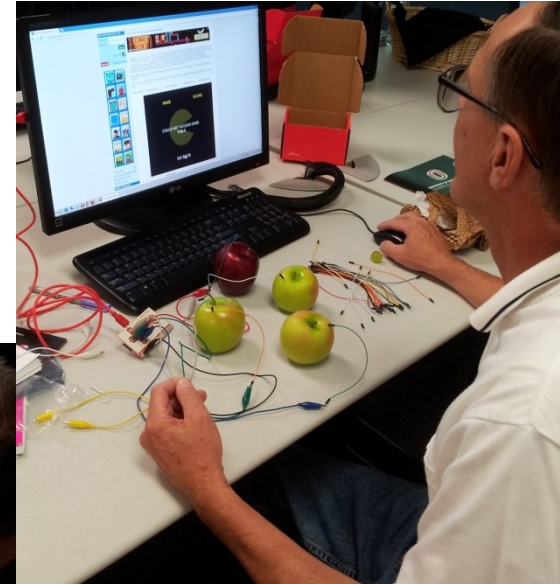
Makey Makey



See demo at: <http://vimeo.com/60307041>

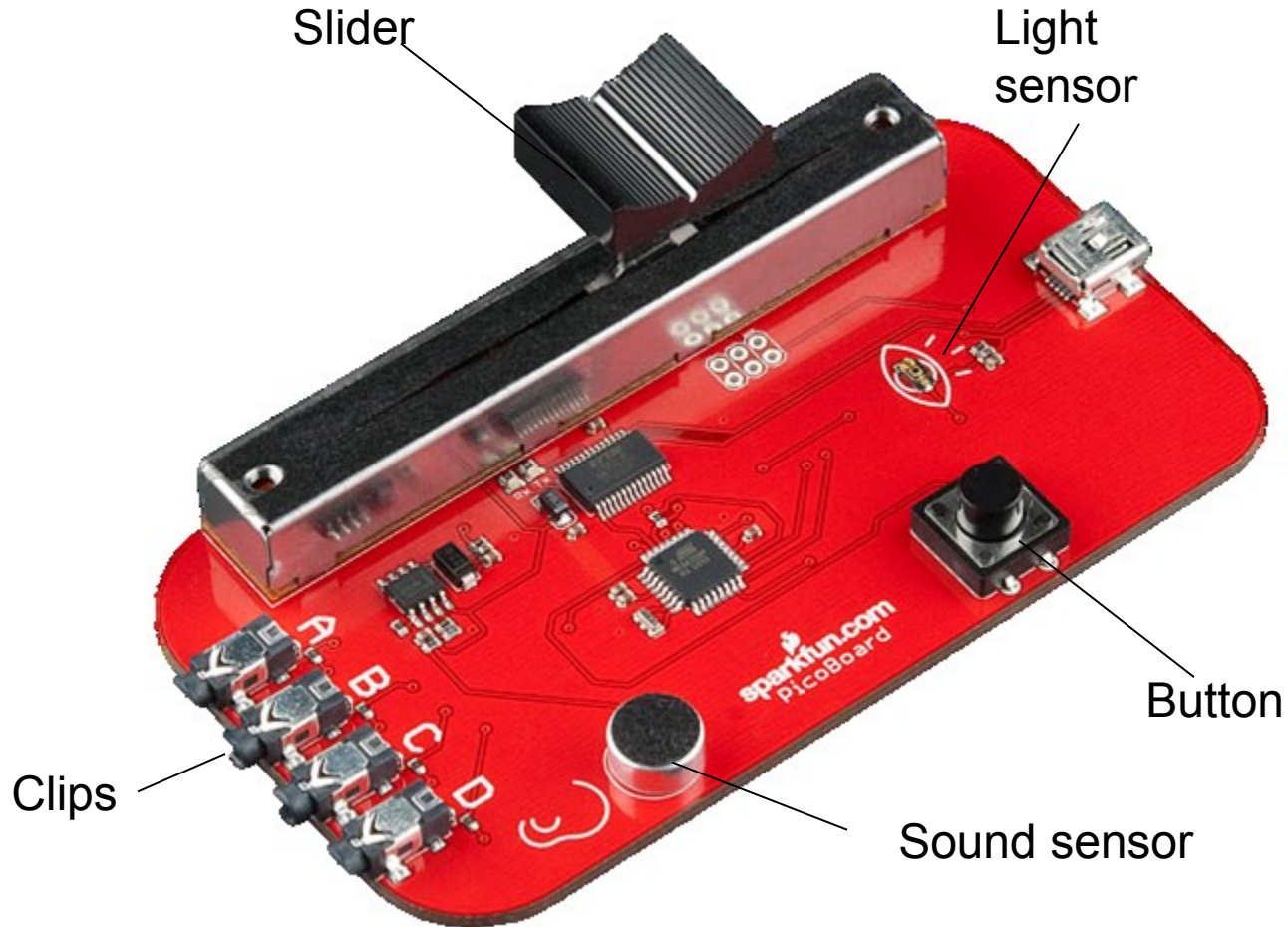


Makey Makeys in action





Picoboard





Picoboard controls (1.4)



The image shows the Sensing block palette in Scratch. It includes a category menu with buttons for Motion, Control, Sensing (highlighted), Looks, Operators, Sound, Pen, and Variables. Below the menu are various sensing blocks: 'touching' (with a dropdown and question mark), 'touching color' (with a color swatch and question mark), 'color is touching' (with a color swatch and question mark), 'ask' (with a text input field and 'and wait' button), 'answer' (checkbox), 'mouse x' and 'mouse y' (input fields), 'mouse down?' (boolean), 'key' (dropdown) 'pressed?' (boolean), 'distance to' (dropdown), 'reset timer' (boolean), 'timer' (checkbox), 'x position' (dropdown) 'of' (dropdown) 'Sprite1' (input field), 'loudness' (checkbox), 'loud?' (boolean), 'slider' (dropdown) 'sensor value' (input field), and 'sensor' (checkbox) 'button pressed?' (dropdown) 'question mark'.

Sensing
button

sensor value

sensor condition test



Comparable languages





Alice



Alice (2.2 03/01/2012) D:\tools\Alice2.2\AliceAndWhiteRabbit.a2zw

File Edit Tools Help

Play Undo Redo

world

- camera
- light
- ground
- whiteRabbit
 - rightArm
 - head
 - leftLeg
 - leftArm
 - rightLeg
 - tail
- aliceLiddell

Events

create new event

When the world starts, do world.my first method

whiteRabbit's details

properties methods functions

create new method

- whiteRabbit move
- whiteRabbit turn
- whiteRabbit roll
- whiteRabbit resize
- whiteRabbit say
- whiteRabbit think

world.my first method

world.my first method No parameters

No variables

create new parameter

create new variable

Do in order

- aliceLiddell.neck.head point at camera more...
- aliceLiddell say Oh, hello there! duration = 2 seconds fontSize = 30 more...
- whiteRabbit point at camera more...
- whiteRabbit say Uhm, yes. Hello there! duration = 2 seconds fontSize = 30 more...
- aliceLiddell say My name is Alice Liddell. duration = 2 seconds fontSize = 30 more...
- whiteRabbit say And I am the White Rabbit. duration = 2 seconds fontSize = 30 more...

Do in order Do together If/Else Loop While For all in order For all together Wait print

Snap

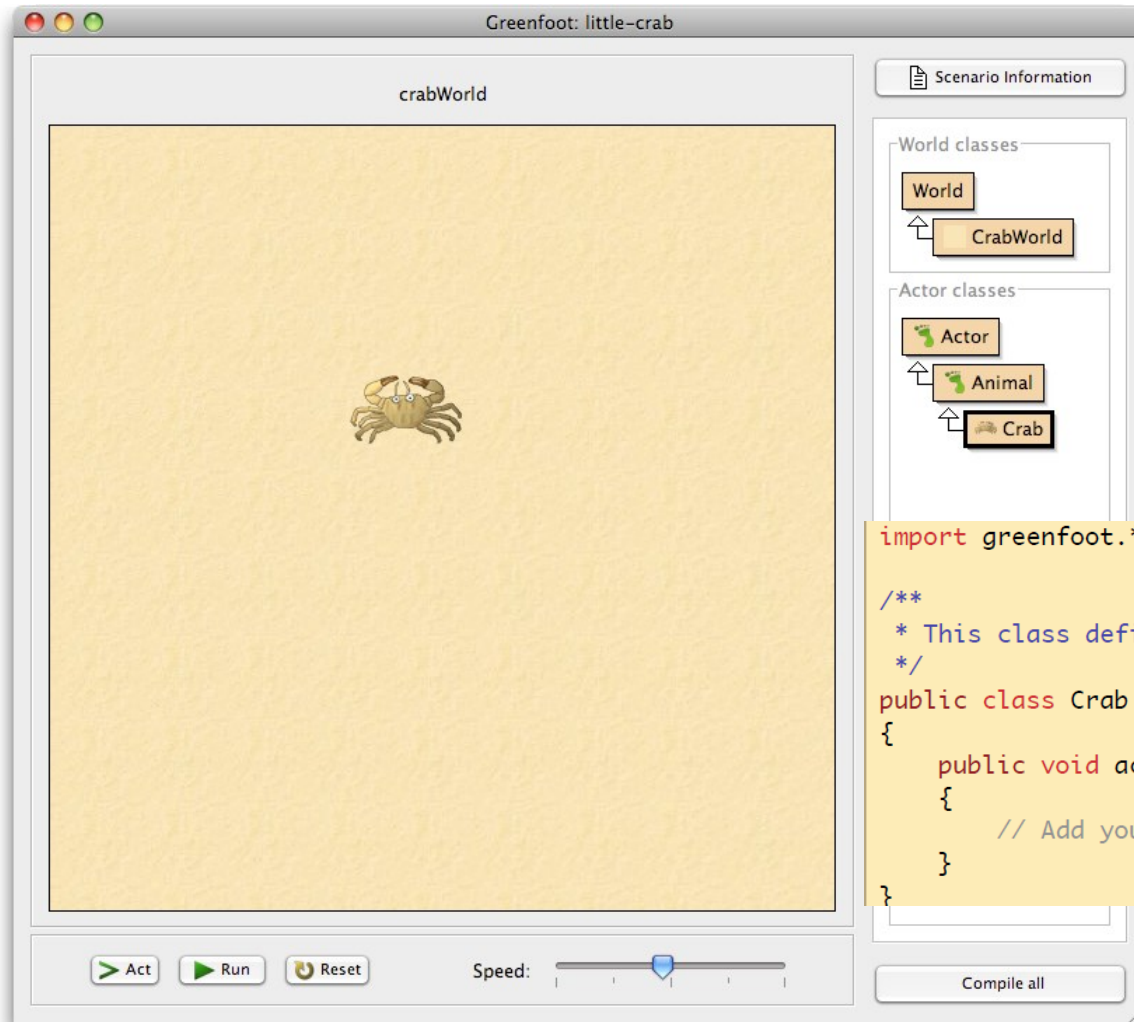


- Spinoff from Scratch
- Supports
 - Higher order functions
 - Recursion
 - Lists of lists

```

    mult list
    script variables helper break
    set helper to
    the script. Input names: sublist
    if empty? sublist
        report 1
    if item 1 of sublist = 0
        run break with inputs 0
    report item 1 of sublist * call helper with inputs all but first of sublist
    report
        the script. Input names: continuation
        call set break to continuation
        report call helper with inputs list
        with current continuation
    
```

Greenfoot

The screenshot shows the Greenfoot IDE interface. On the left, a window titled "Greenfoot: little-crab" displays a "crabWorld" with a single crab actor on a yellow background. On the right, a "Scenario Information" panel shows a class hierarchy:

- World classes:** World (base class), CrabWorld (subclass).
- Actor classes:** Actor (base class), Animal (subclass), Crab (subclass).

At the bottom of the IDE, there are control buttons for "Act", "Run", and "Reset", a "Speed" slider, and a "Compile all" button.

A gentle transition
into Java

```
import greenfoot.*; // (World, Actor, GreenfootImage, and Greenfoot)

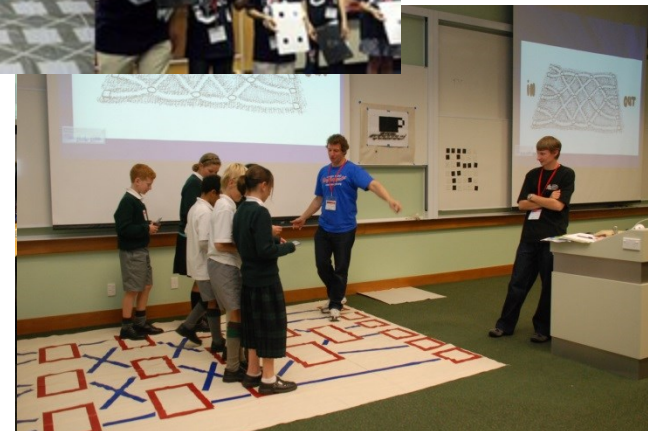
/**
 * This class defines a crab. Crabs live on the beach.
 */
public class Crab extends Animal
{
    public void act()
    {
        // Add your action code here.
    }
}
```



CS Unplugged

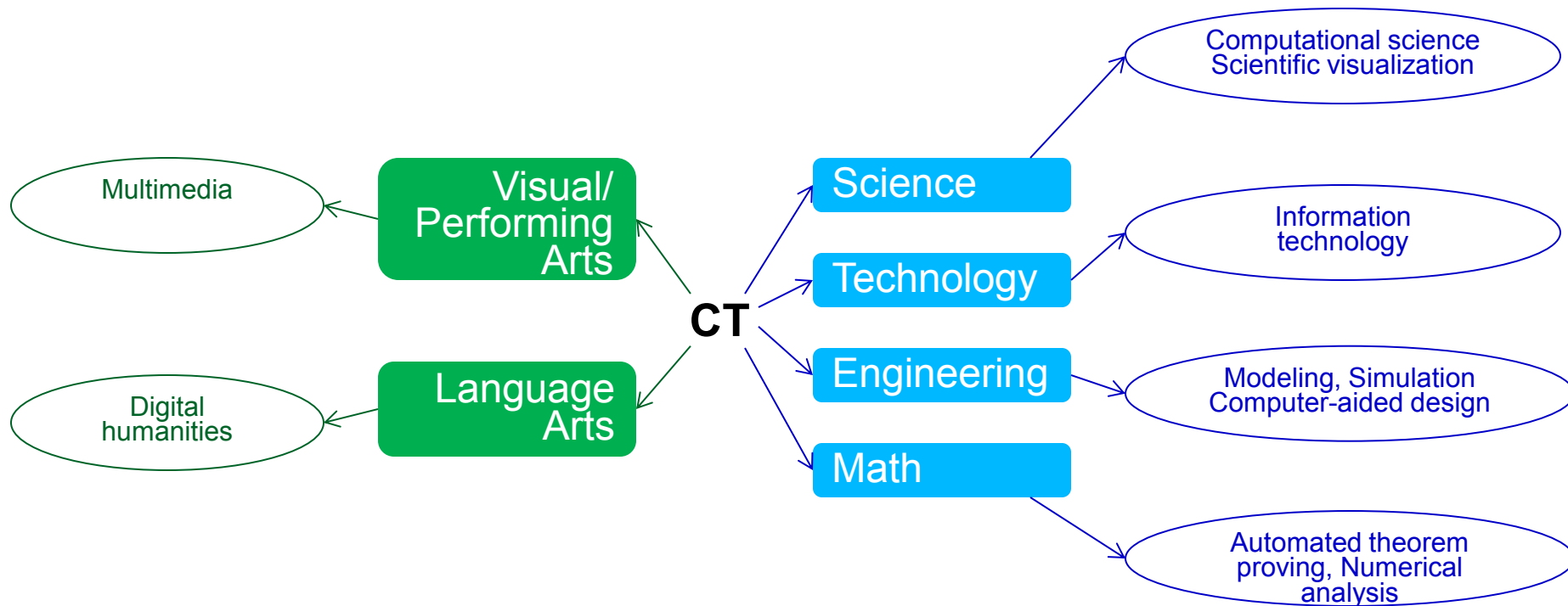


- <http://csunplugged.org>
- Computational thinking without computers!





Computational Thinking across the curriculum





Computational Thinking courses



Exploring Computer Science

<http://www.exploringcs.org/>

Follow @ExploringCS



Exploring
Computer
Science

Home About Curriculum Teacher Program News & Events Resources Contact



Exploring Computer Science

A K-12/University partnership committed to democratizing computer science.

ECS Intro Video



Computer Science Principles

<http://www.csprinciples.org/>

About the Project
Pilot Sites
Resources
Professional Development
People
Contact

Computer Science: Principles is a proposed AP course under development that seeks to broaden participation in computing and computer science. Development is being led by a team of computer science educators organized by the College Board and the National Science Foundation.

Welcome to the CS Principles website! Here you will find information [about CS Principles](#), [resources for teachers](#), and details about the [sponsored early adopters](#) currently piloting CS Principles.

[Sign up to join the team of early adopters!](#)

Like 120 people like this.



Some CS Resources



-
- J. Wing. Computational Thinking, *Communications of the ACM*, March 2006.
 - Computer Science Teachers Association, <http://csta.acm.org>
 - Google. Exploring Computational Thinking, <http://www.google.com/edu/computational-thinking>.
 - CS Unplugged, <http://csunplugged.org>.
 - Yadav. Computational Thinking in K-12 Education, http://cs4edu.cs.purdue.edu/_media/ct-in-k12_edps235.pdf.
 - See the list of resources near the end.



Scratch Resources



-
- Website: <http://scratch.mit.edu> (lots of examples)
 - Curriculum Guide:
<http://scratched.media.mit.edu/resources/scratch-curriculum-guide-draft>
 - After-school lessons:
<http://scratch.redware.com/lessonplan>