Creative Programming in Scratch

CS4HS 2011

Hélène Martin

http://garfieldcs.com/static/2011cs4hsScratch

You might be wondering...

- Why is programming worth teaching?
- What kind of learning can Scratch support?
- What kinds of projects help students develop critical thinking skills?
- How can Scratch projects be assessed?
- How can Scratch support your course goals?

Programming?

- “Telling the computer what to do”
- Writing, testing and maintaining source code
- Creating original digital artifacts

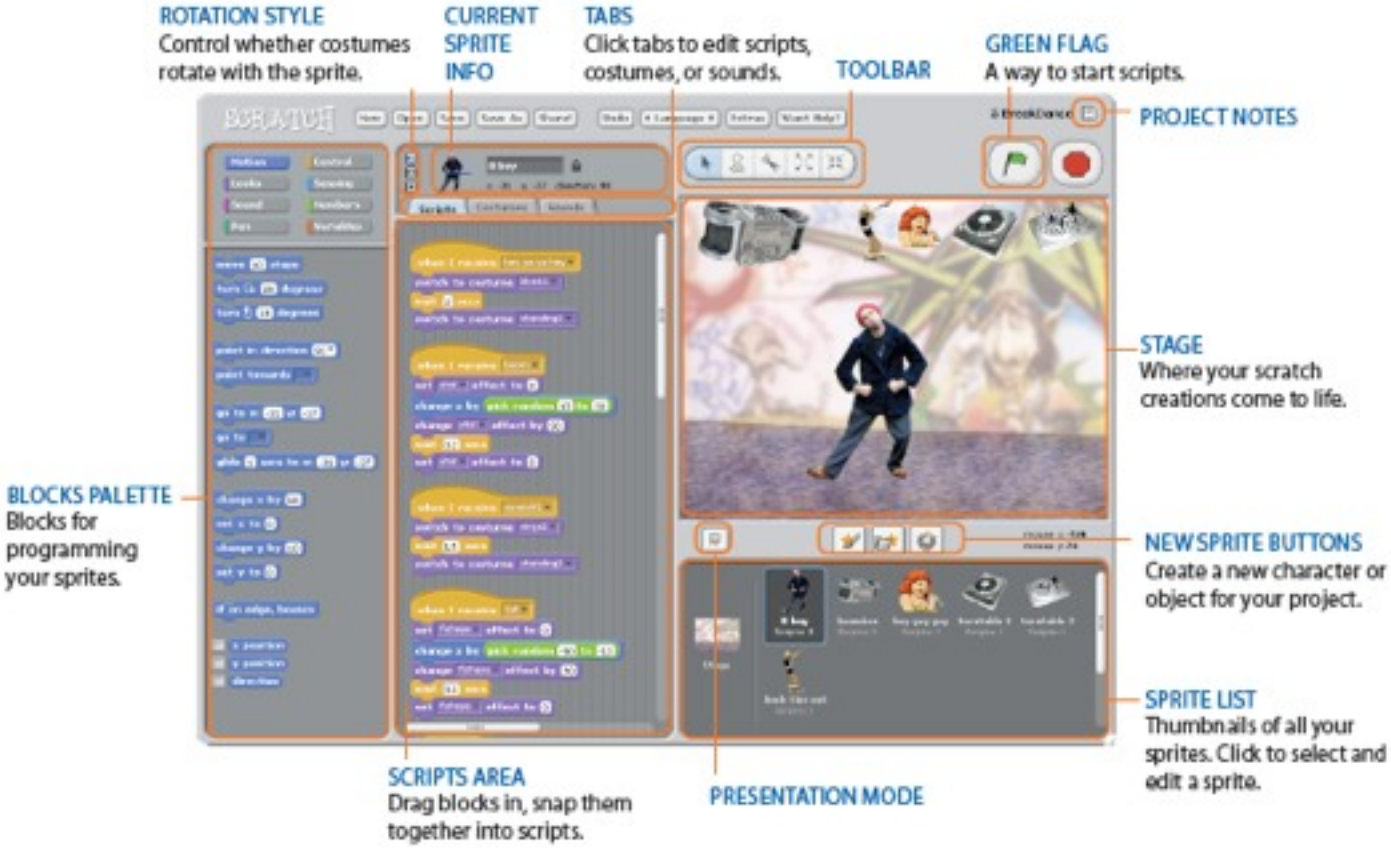
- Rule-based, detail-oriented
- Empowering!

Why teach programming?

- Increase problem-solving abilities
  - Algorithmic thinking
  - What computers can/can't do
- Reinforce learning from core subjects
  - Algebra
  - Geometry
  - Scientific method
- **It's concrete**

Scratch: visual programming

- http://scratch.mit.edu



ROTATION STYLE
Control whether costumes rotate with the sprite.

CURRENT SPRITE INFO

TABS
Click tabs to edit scripts, costumes, or sounds.

TOOLBAR

GREEN FLAG
A way to start scripts.

PROJECT NOTES

STAGE
Where your scratch creations come to life.

NEW SPRITE BUTTONS
Create a new character or object for your project.

SPRITE LIST
Thumbnails of all your sprites. Click to select and edit a sprite.

BLOCKS PALETTE
Blocks for programming your sprites.

SCRIPTS AREA
Drag blocks in, snap them together into scripts.

PRESENTATION MODE

Why Scratch?

- No syntax learning-curve
- Build intuitions about computing concepts relevant to most languages
- Immediate feedback
- Great social networking component
- It’s good enough for Harvard and Berkeley

Warning!

- It’s not algebraic
- Major concepts are missing - abstraction
  - See Snap (what Berkeley uses)
- Easy to slip into entirely unstructured play
  - Repetitive code
  - Lack of transfer

Check it out

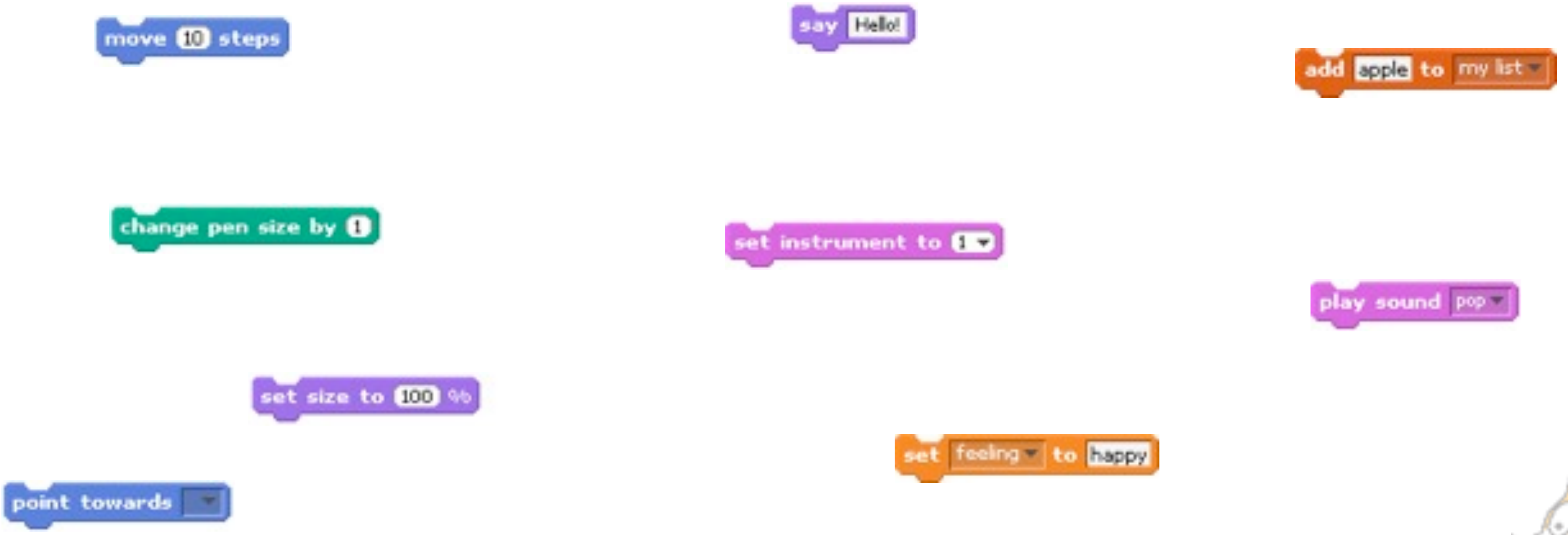
- Try out different projects on http://scratch.mit.edu/channel/featured (second link from workshop website)
- Look for projects relevant to your subject area

Scripts

- Formalized algorithms
- Combination of blocks
- Attached to sprites or the stage
- Blocks only fit where syntactically correct

Statements

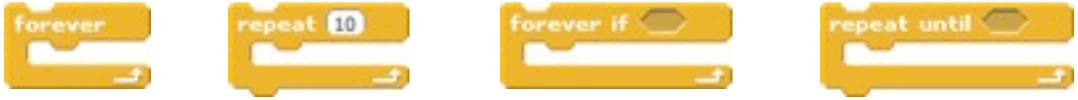
- Commands or instructions
- Simple statements are verbs



```
move 10 steps
say Hello!
add apple to my list
change pen size by 1
set instrument to 1
play sound pop
set size to 100 %
set feeling to happy
point towards
```

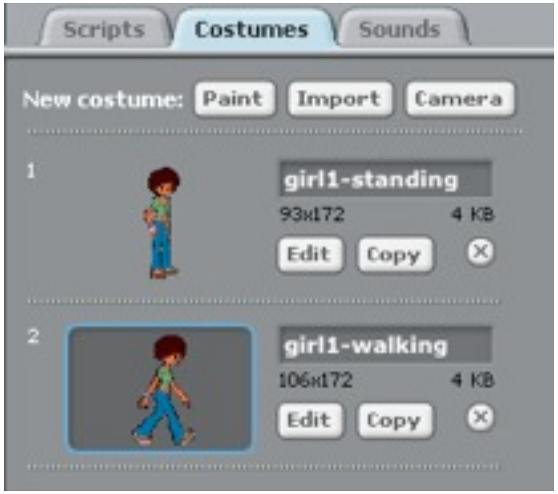
Repetition

- Loops are used to repeat a piece of code
  - Iteration is critical in CS but also math, science



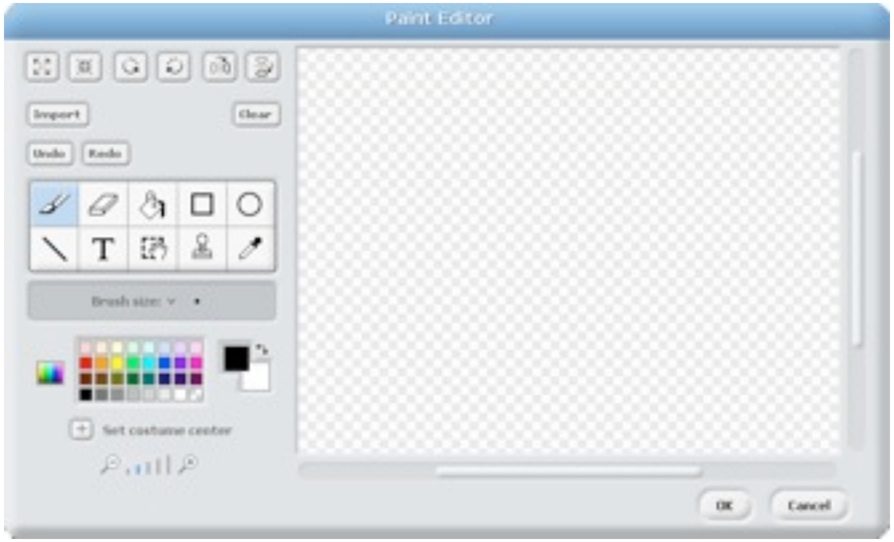
Visuals

- Sprites have costumes
- The stage has backgrounds



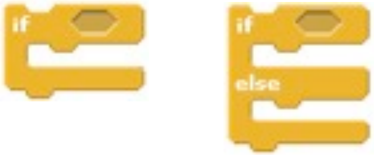
1 girl1-standing 93x172 4 KB

2 girl1-walking 106x172 4 KB

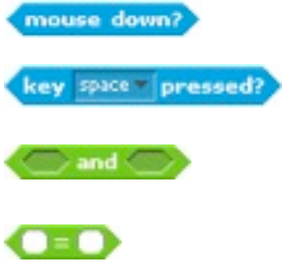


Conditionals

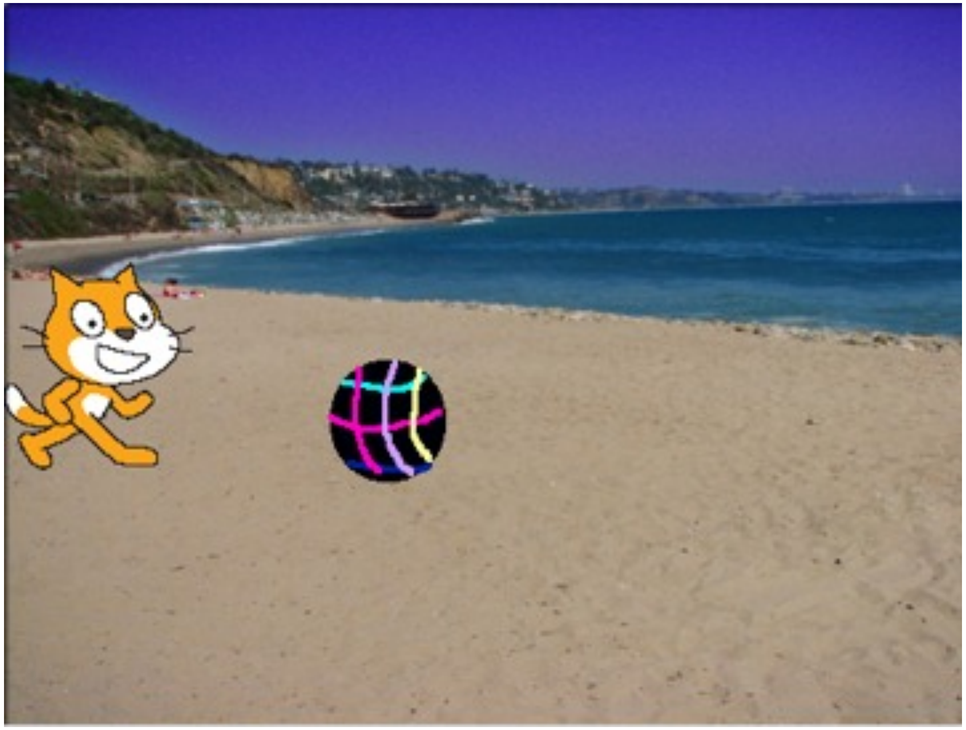
- Not all statements should be run all the time



- Boolean expressions define conditions
  - Evaluate to true or false

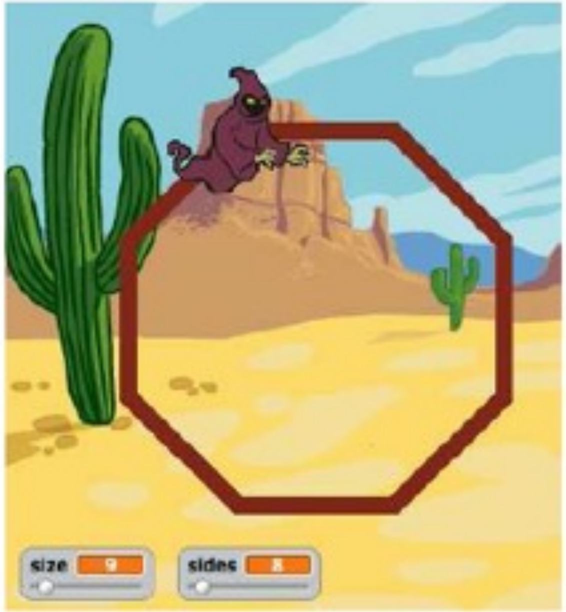


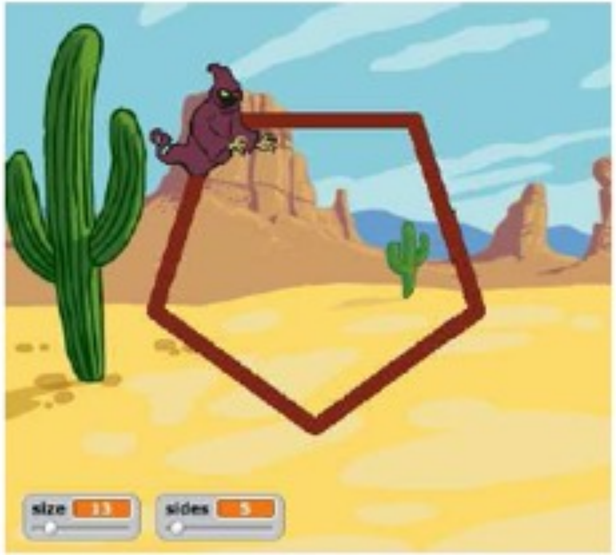
Bouncing ball on beach



Variables

- Placeholders for values
- Named by the user
- Your turn: resizable polygon





Rubric: Resizable polygon

- __/2 – has a size variable
- __/2 – has a sides variable
- __/2 – changing variables has desired effect
- __/1 – has a creative background
- __/2 – variable sliders on stage for user to set
- __/1 – clicking on green flag starts program

Threads

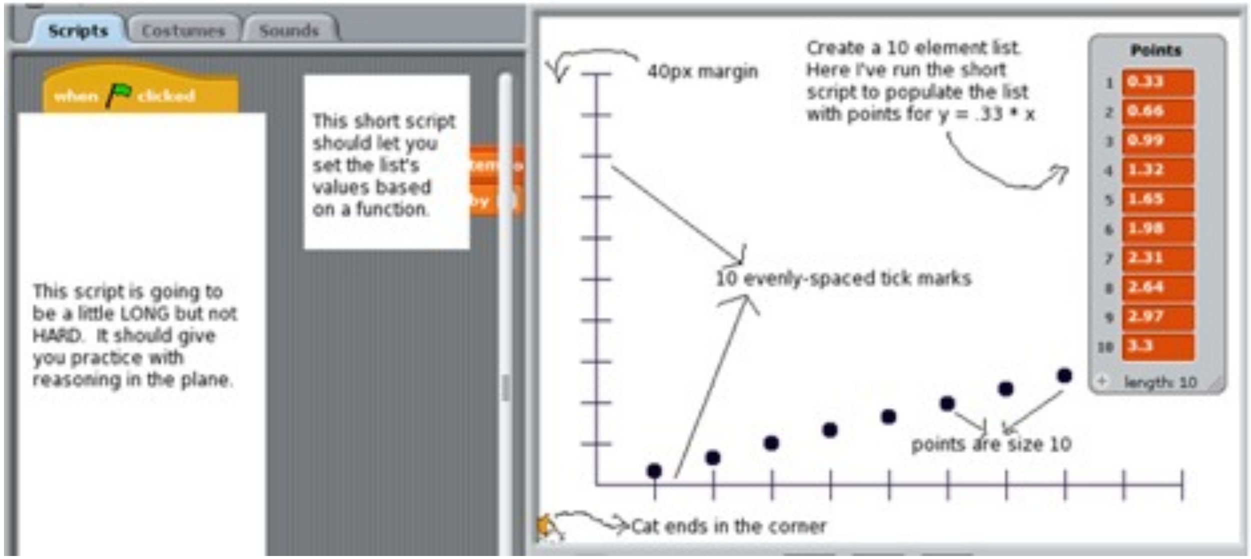
- ADVANCED programming topic
- Coordination of multiple things going on at once
- Any block beginning with ‘When’ starts a thread

Events

- Signal from one thread to another
- Broadcast blocks send events
- Your turn: LeBron James buttons project

Lists

- A type of variable that contains multiple related values
- See writeup for graphing assignment



This short script should let you set the list's values based on a function.

This script is going to be a little LONG but not HARD. It should give you practice with reasoning in the plane.

40px margin

Create a 10 element list. Here I've run the short script to populate the list with points for y = .33 * x

10 evenly-spaced tick marks

points are size 10

Cat ends in the corner

	Points
1	0.33
2	0.66
3	0.99
4	1.32
5	1.65
6	1.98
7	2.31
8	2.64
9	2.97
10	3.3

length: 10

Practicing problem solving

- Understanding problem statements
- Decomposing problems into solvable pieces
- Incremental design
- **Sample problem**: The user can set gridHeight, gridWidth and gridMargin with sliders. The cat should draw a grid of size gridHeight by gridWidth, gridMargin away from the edge of the screen.

Breaking down the grid problem

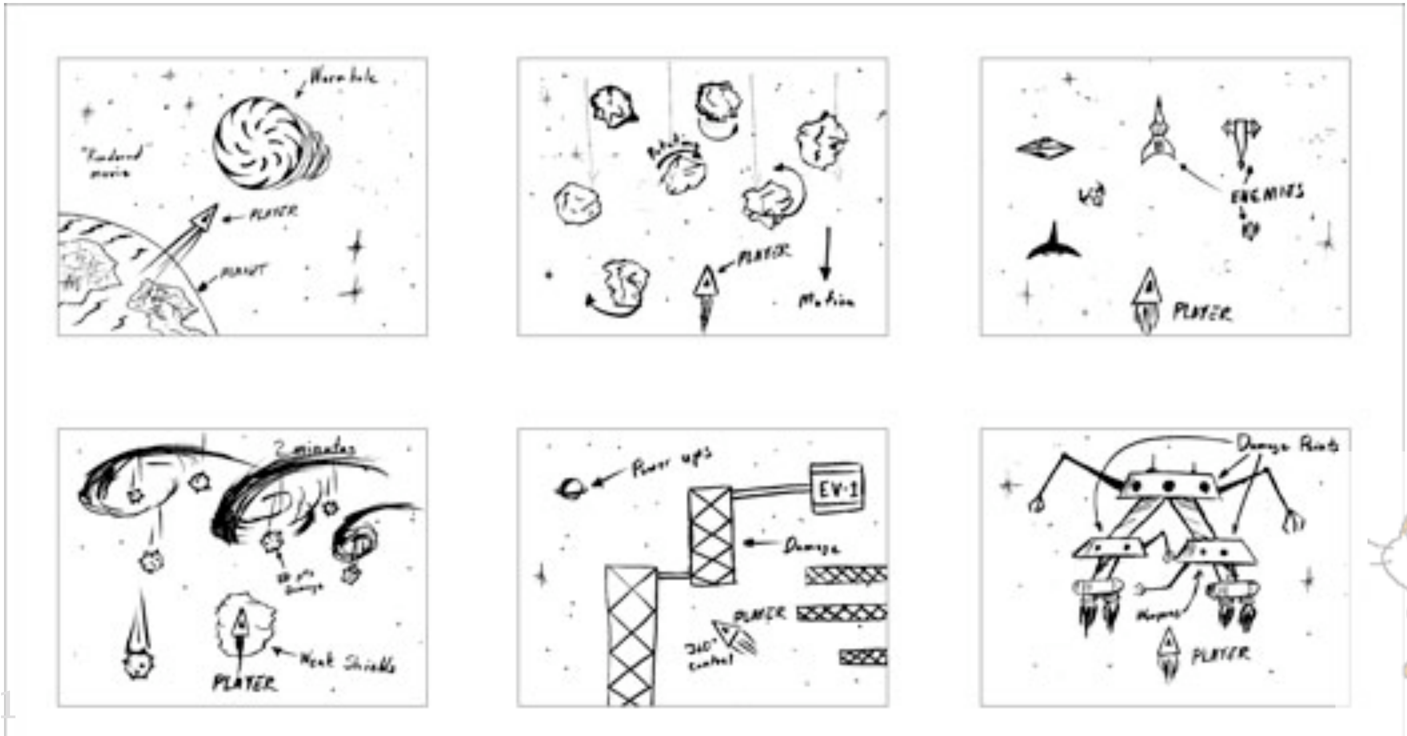
- (**Syntax**) How can we get Cat to draw a horizontal line 30 from the top of the screen?
- (**Geometry**) How can we get Cat to draw 5 evenly spaced horizontal lines?
- (**Syntax**) What kind of loop should we use?
- (**Algorithm**) How many times will it repeat?
- (**Algebra**) What if we want a margin of 30 on top and bottom?

Writing: proposal

- Students write formal project proposals
- Teach technical writing
- Bring in industry guest to hear elevator pitches

Design: storyboarding

- Encourage students to plan before acting
- Help students develop storytelling abilities
- Introduce a real-world technique



Parting words

- Encourage students to be tech producers
- Allow “guided play”
- Teach meaningful critical thinking skills
- “Hide” important lessons in a playful environment
- Leverage existing materials