

Scratch Programming

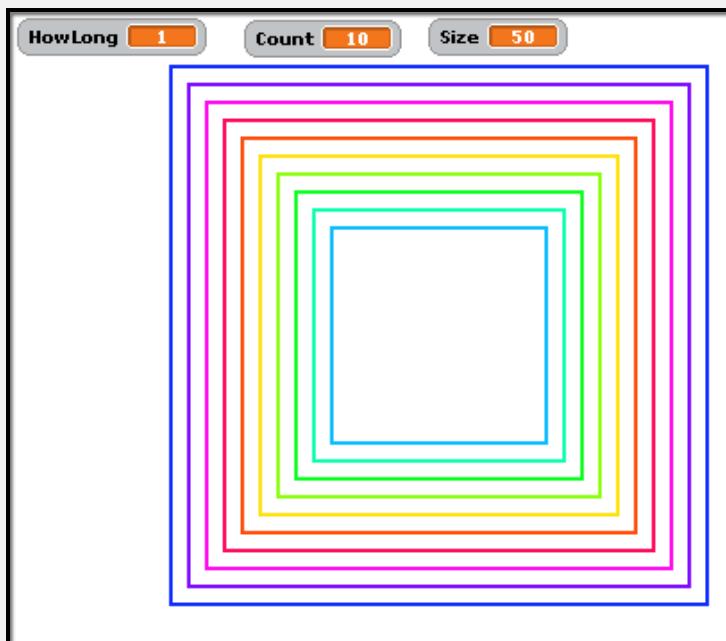
<http://cs.oregonstate.edu/~mjb/scratch>



Oregon State
University

Mike Bailey

mjb@cs.oregonstate.edu



Here's What These Notes Are Going to Cover ☺

2



Oregon State

University

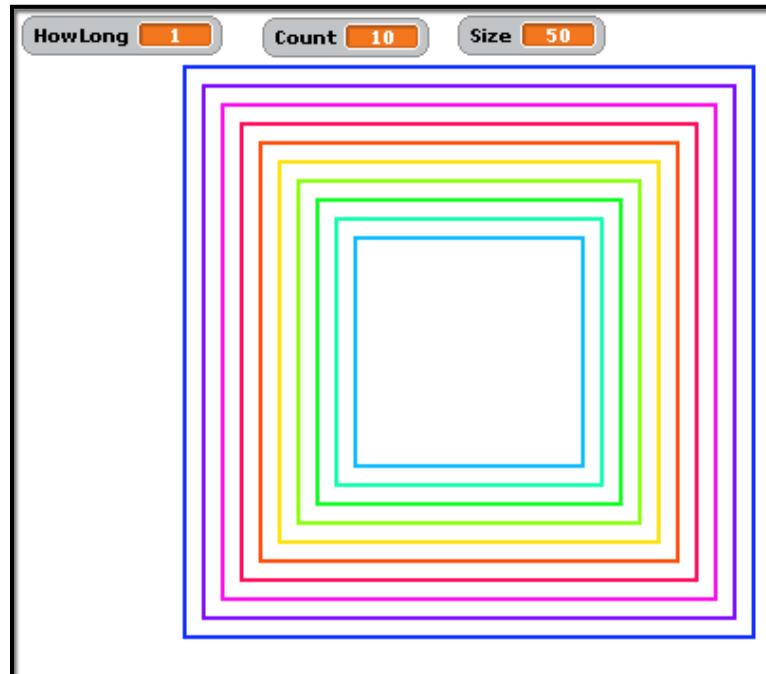
Computer Graphics

What is “Scratch”?

Scratch is a programming environment that lets you create your own artwork and animations. It's easy and it's fun! It also teaches you the fundamentals of programming, possibly without you realizing that's what you are doing.

Scratch was developed at MIT. It is free, and runs on Windows, Mac, and Linux machines. There is now also a web browser-only version.

In these notes, we will write a program that makes spiraling squares that change color. This is just a start -- you can do lots more than this!



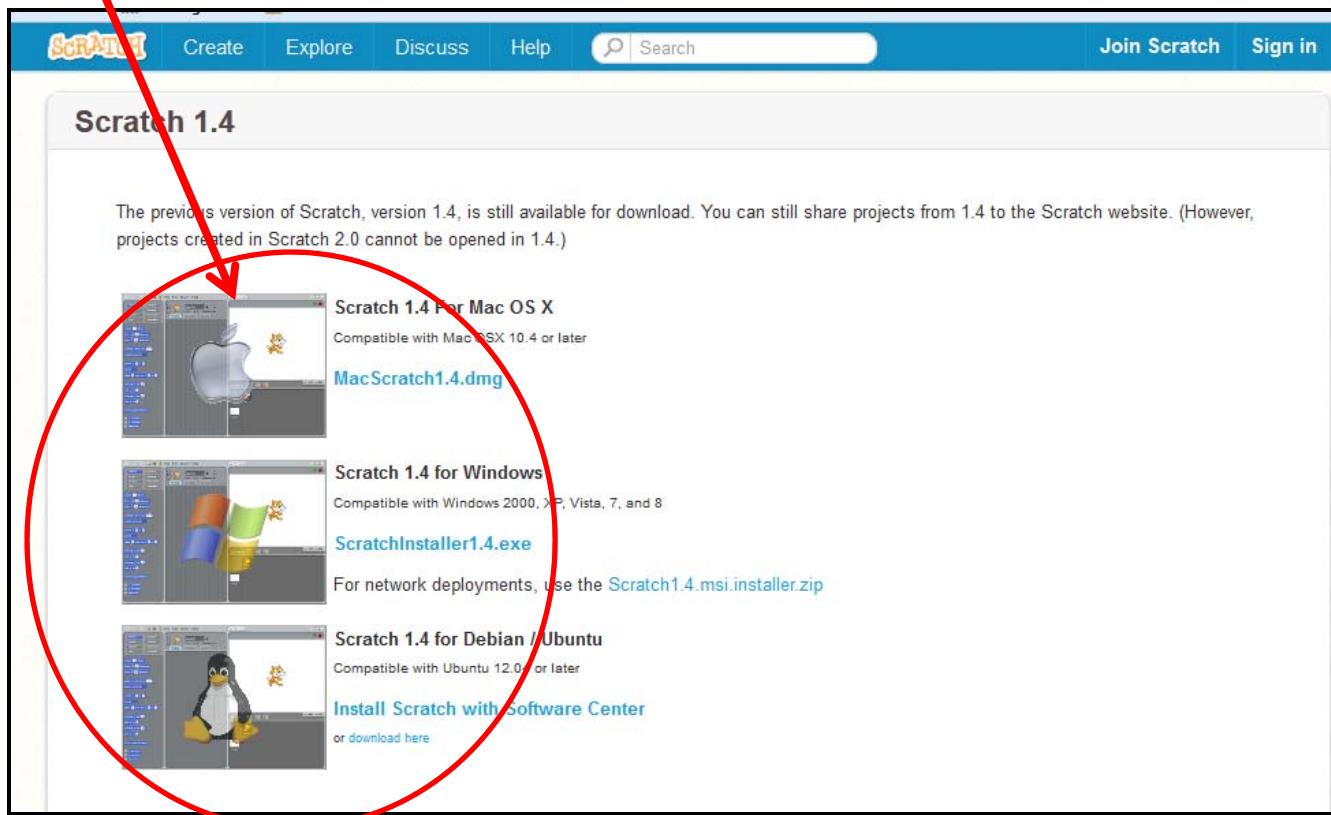
How Can You Get Scratch 1.4 for Free?

4

Scratch is free. To download it, go to the Scratch web site:

http://scratch.mit.edu/scratch_1.4/

and click here on one of these:



How Can You Get Scratch 2.0 for Free?

5

Scratch 1.4 is very stable and has the features you most want to use. However, there is a new Scratch 2.0 that has some new features you might have fun with (e.g., cloning, code blocks). Download it at:

<http://scratch.mit.edu/scratch2download/>

You can also get a web browser-based version of Scratch 2.0 at:

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

Explanations of some of the new Scratch 2.0 features are found later in these notes.



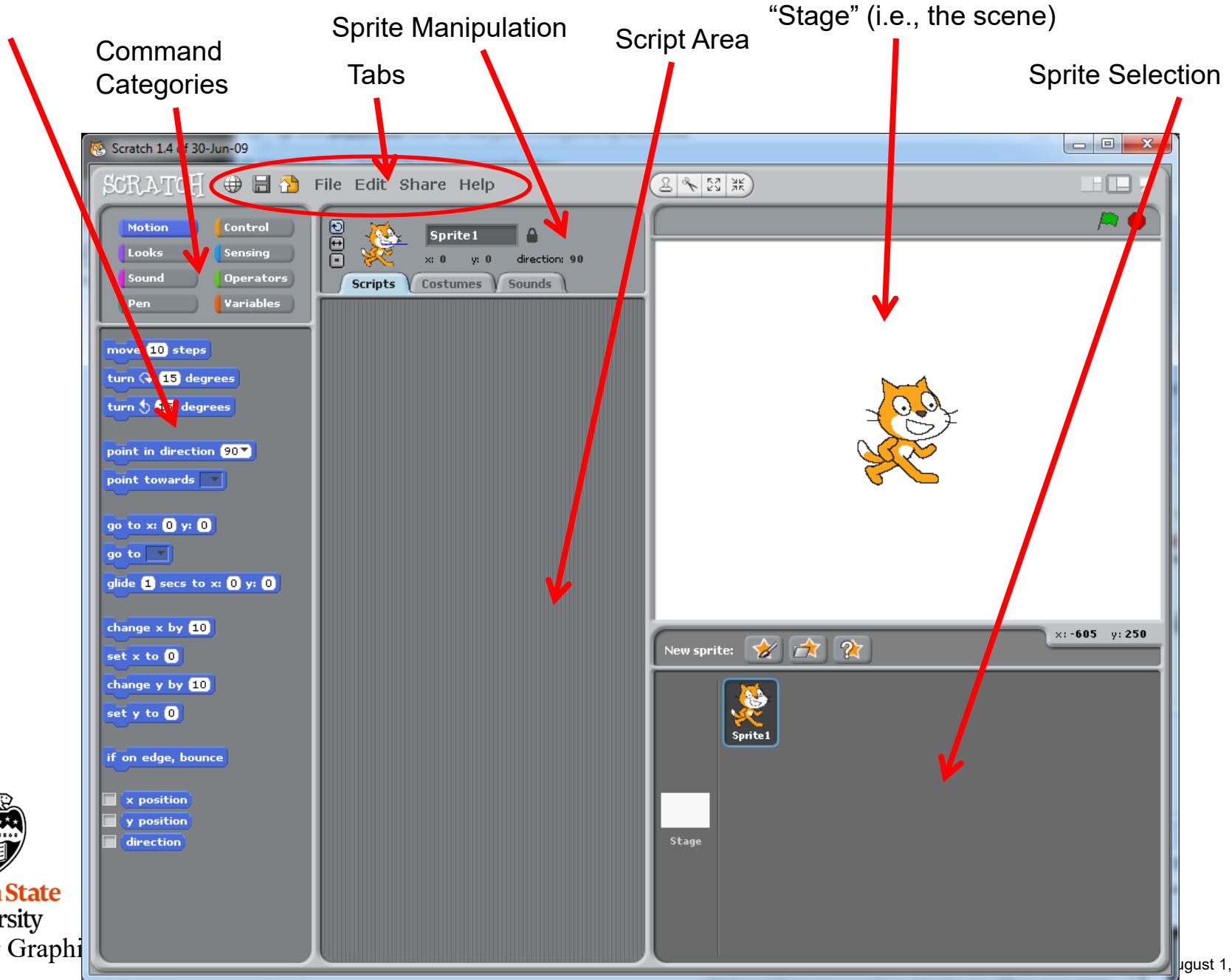
Oregon State
University

Computer Graphics

mjb – August 1, 2018

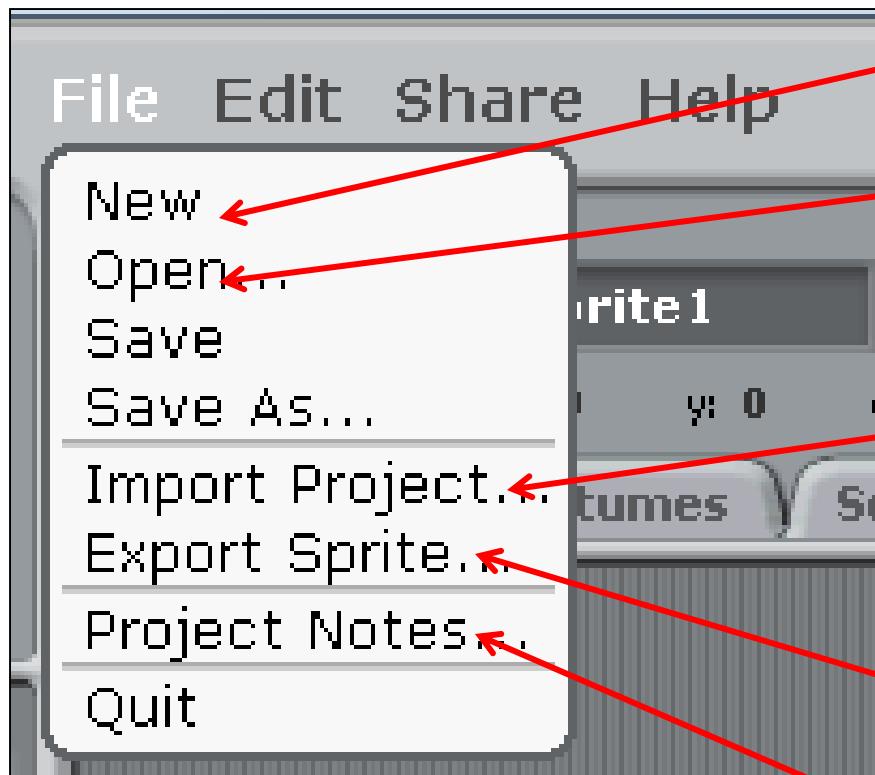
Scratch Screen Layout

Commands in a Selected Menu



Oregon State
University
Computer Graphics

Menu Tabs -- File



Abandon the current project and start a new one

Abandon the current project and open an existing one

Without abandoning the current project, bring an existing one in and assign all its properties (including its scripts) to a new sprite

Write out all of the information about the current sprite

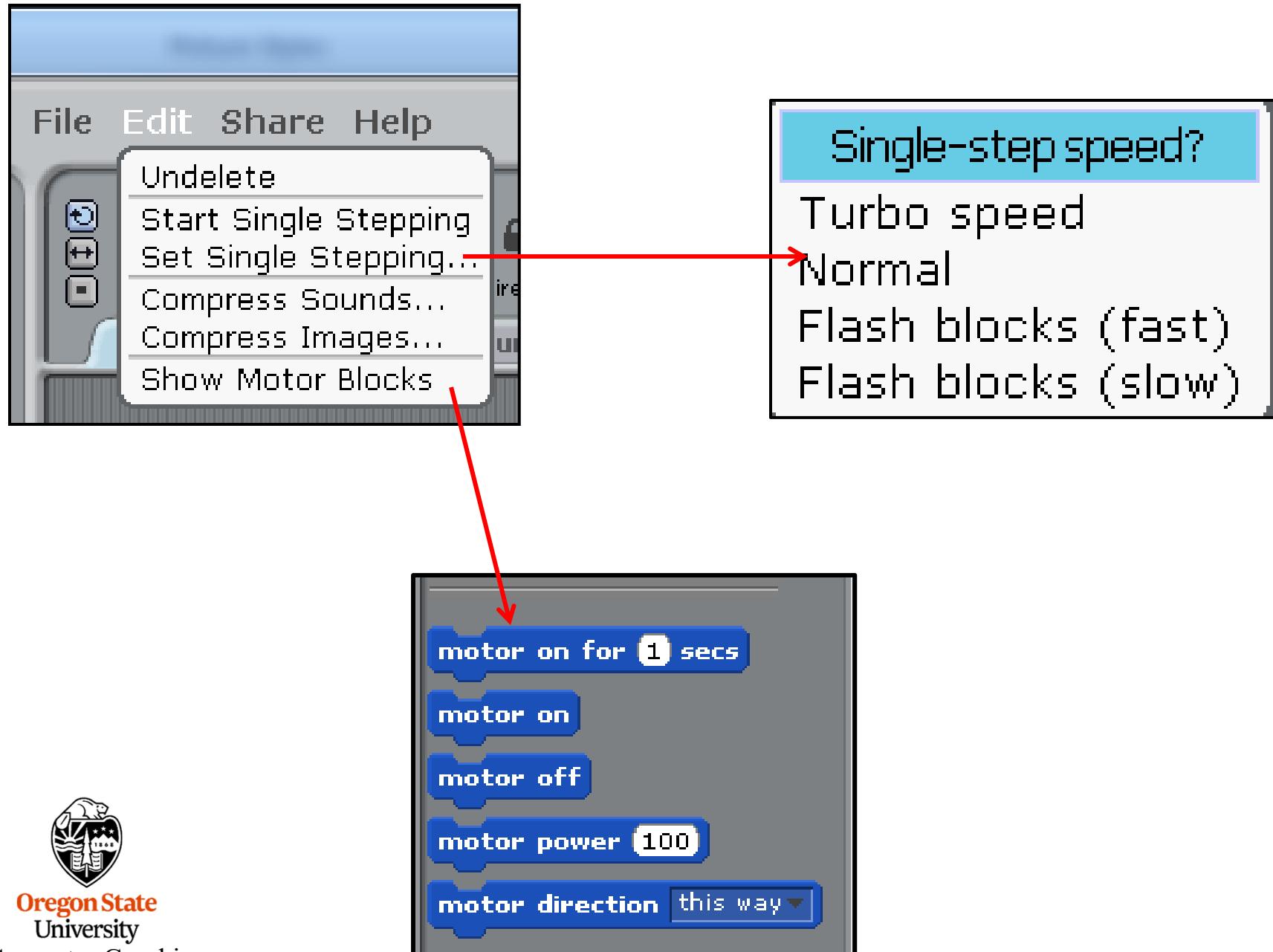
Brings up a notepad for you to record information about this project



Oregon State
University

Computer Graphics

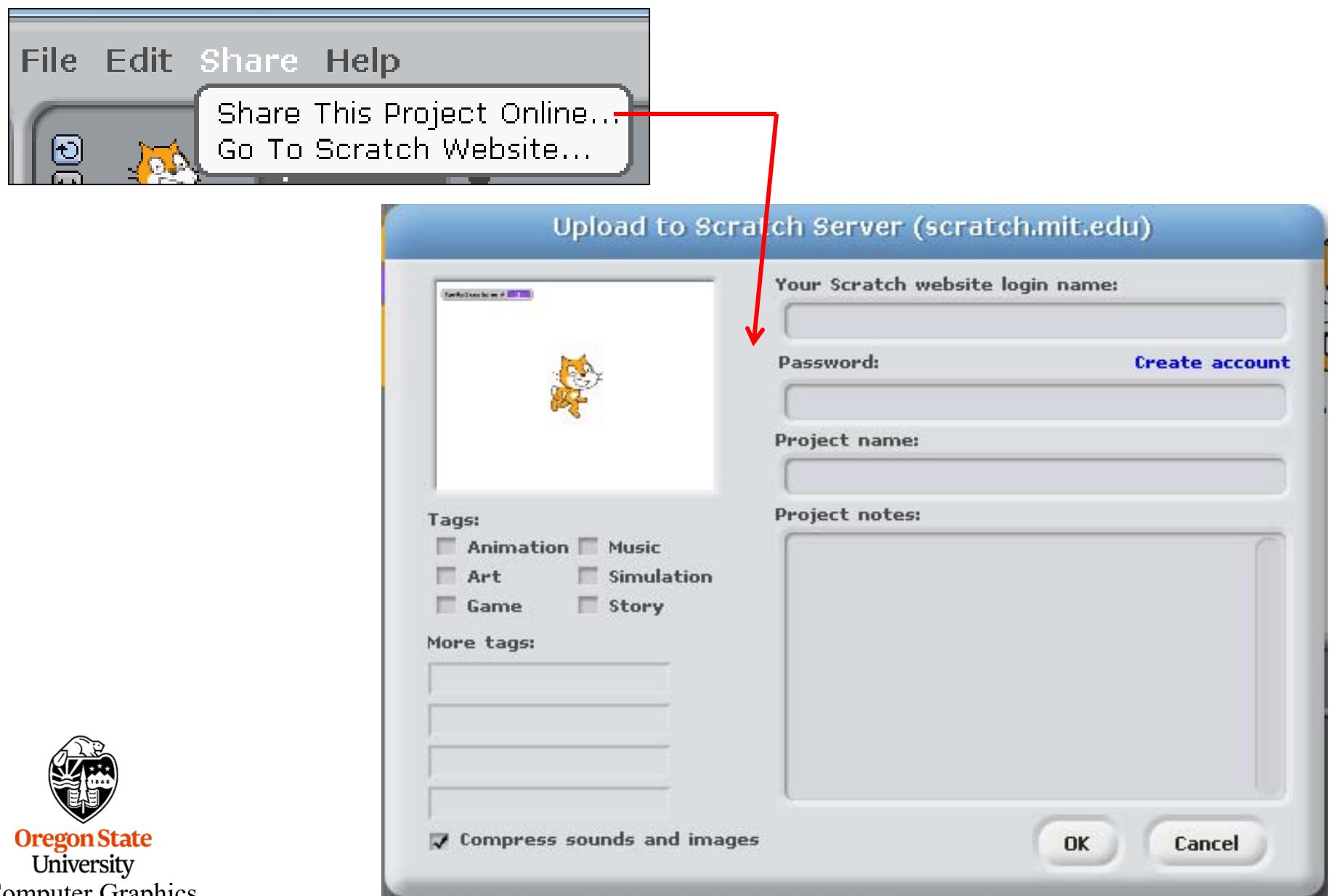
Menu Tabs -- Edit



Oregon State
University

Computer Graphics

Menu Tabs -- Share



Menu Tabs -- Help

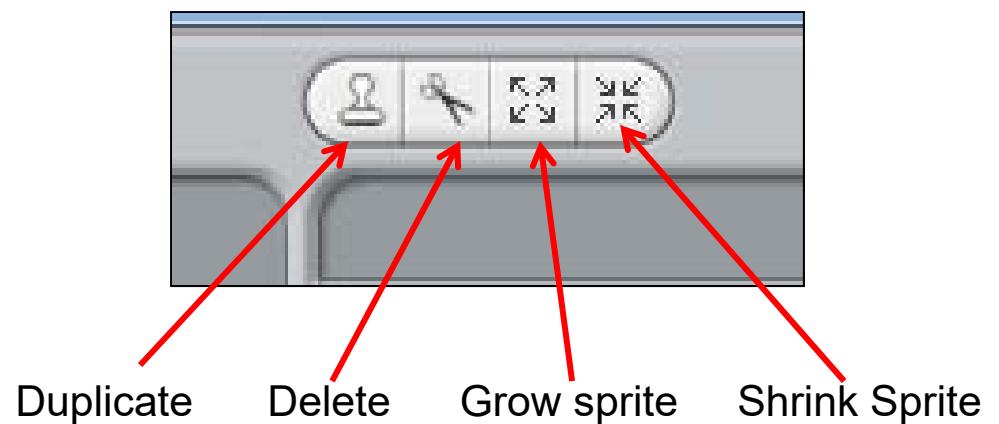
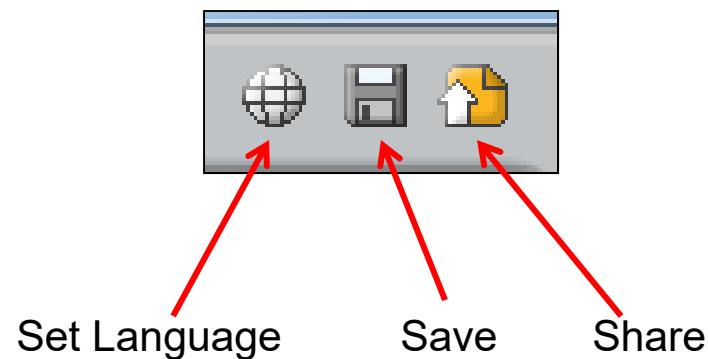
10



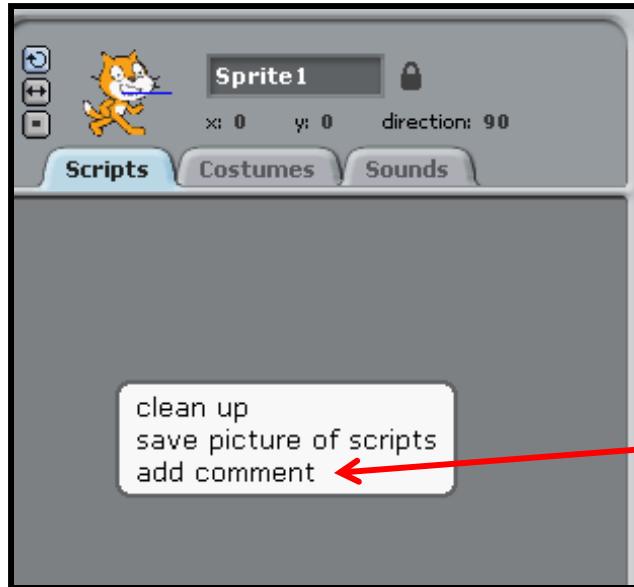
Oregon State
University
Computer Graphics

mjb – August 1, 2018

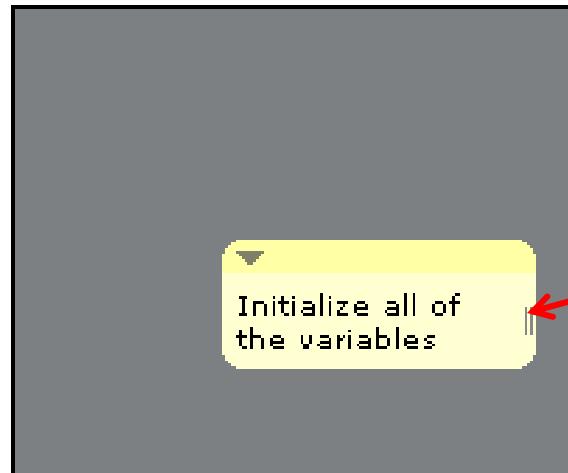
Icons



A Good Way to Start Any Program – A Collection of Comments Describing What You Are Trying To Do

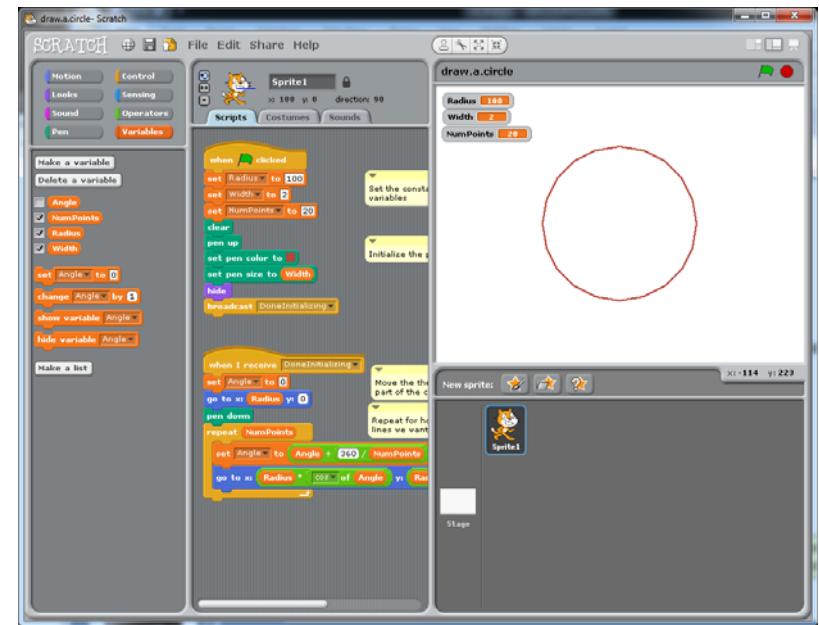
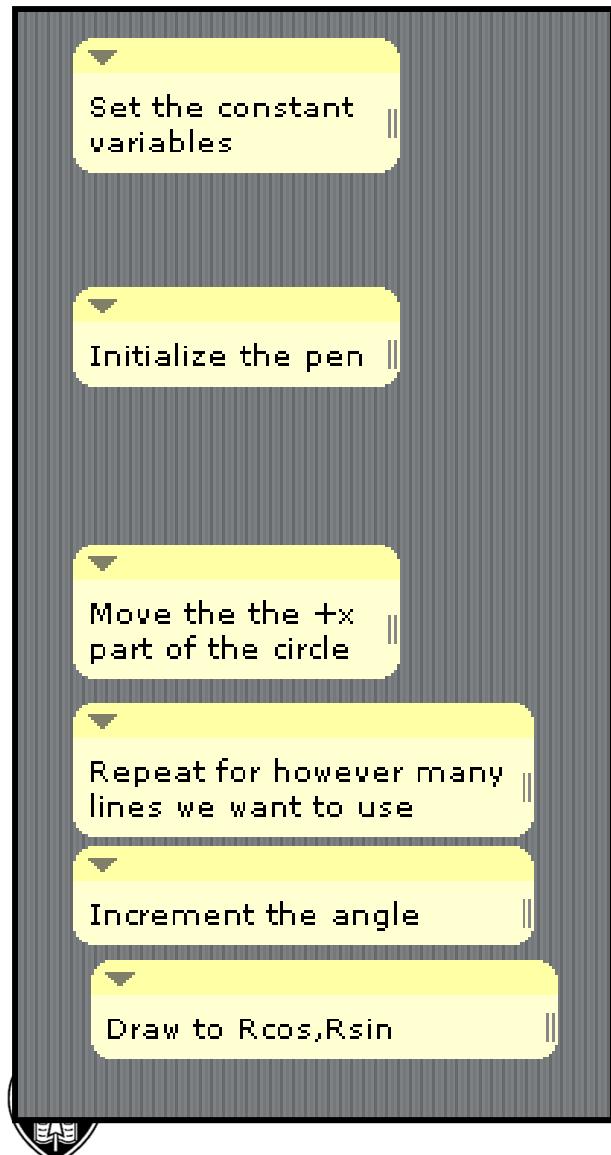


Right-click in the Script Area and select **add comment**



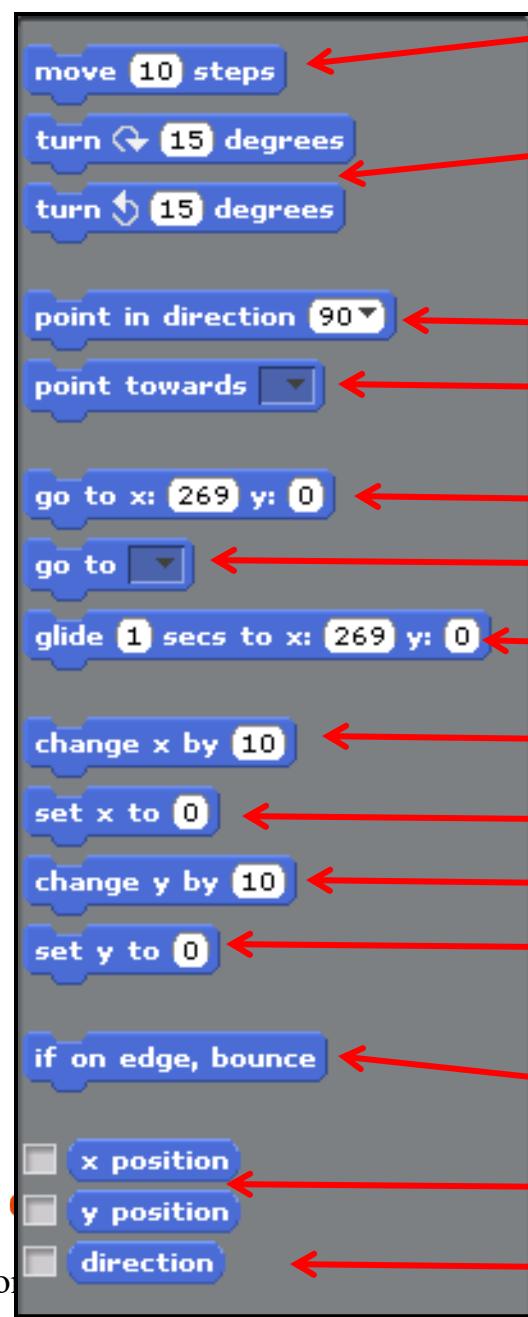
Then type your comment

A set of English-readable statements are a better way to start than just entering code.



A set of English-readable statements are a better way to start than just entering code.

Motion Menu



Move the sprite a certain number of steps (pixels) in the direction it is pointing

Change the pointing direction clockwise or counterclockwise



Set the pointing compass direction

Point towards something in particular (e.g., mouse-pointer)

Move the sprite to a particular location

Move the sprite somewhere in particular (e.g., mouse-pointer)

Animate the sprite somewhere in particular

Change the sprite's x location by a certain amount (+ or -)

Set the sprite's x location to a certain amount

Change the sprite's y location by a certain amount (+ or -)

Set the sprite's y location to a certain amount

If this sprite is on a scene edge, turn it around so it can move in the other direction

Display the x and y locations in the scene

Display the direction angle in the scene

Costumes

Each sprite can have multiple ways it can look. Each of these is called a **costume**.



These can be changed during the course of an animation to give the appearance that the subject is moving or has simply changed its “look”.



Oregon State

University

Computer Graphics

Looks Menu

Change this sprite to look like its designated “costume”

Change this sprite to look like its next costume

Display the costume number in the scene

Put a talking balloon on this sprite.

Put a thinking balloon on this sprite

Change one of these effects

Set one of these effects

Clear all of these effects

Change the sprite size (%)

Set the sprite size (units are percent of original size)

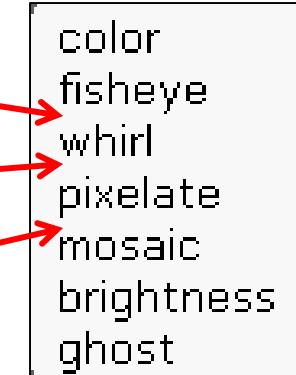
Display the sprite size in the scene

Show this sprite

Don't display this sprite

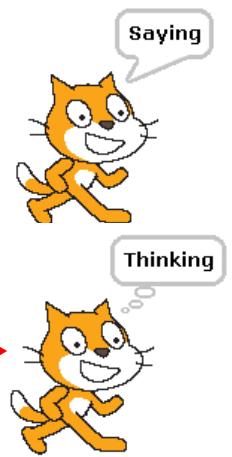
Move this sprite in front of all the others

Move this sprite behind one layer of sprites



The graphic effect options listed in the callout box are:

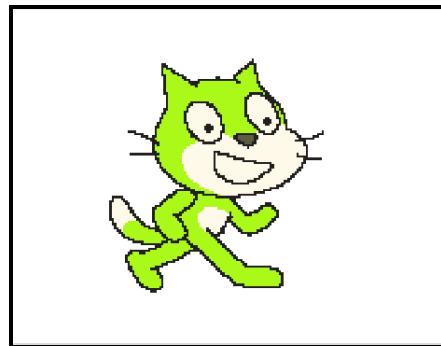
- color
- fisheye
- whirl
- pixelate
- mosaic
- brightness
- ghost



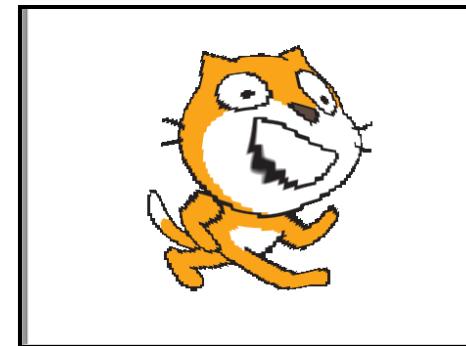
Effects in the Looks Menu

17

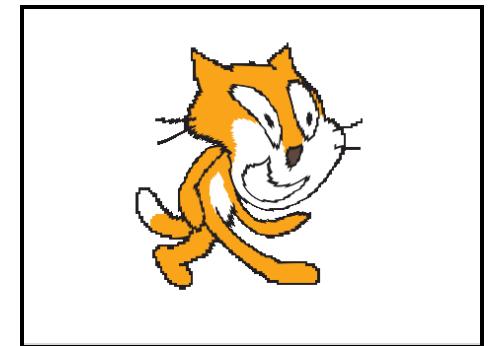
color
fisheye
whirl
pixelate
mosaic
brightness
ghost



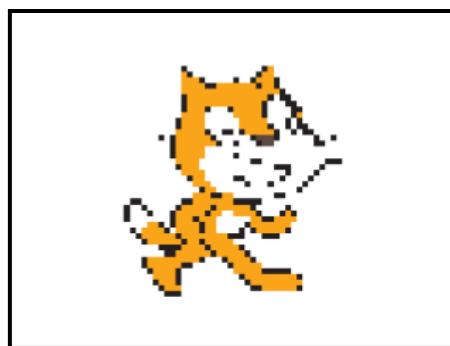
color



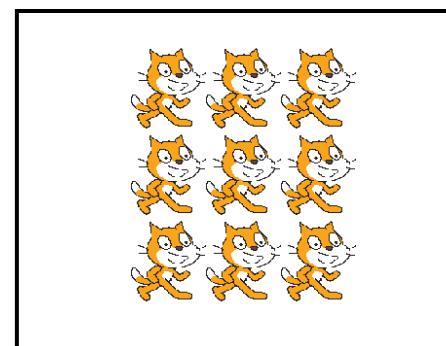
fisheye



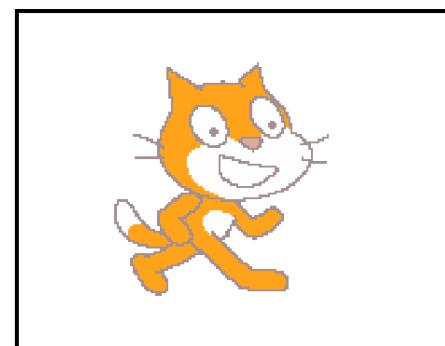
whirl



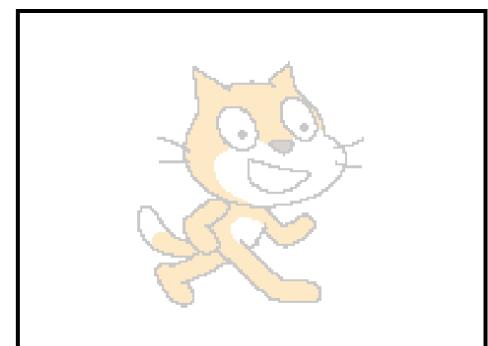
pixelate



mosaic

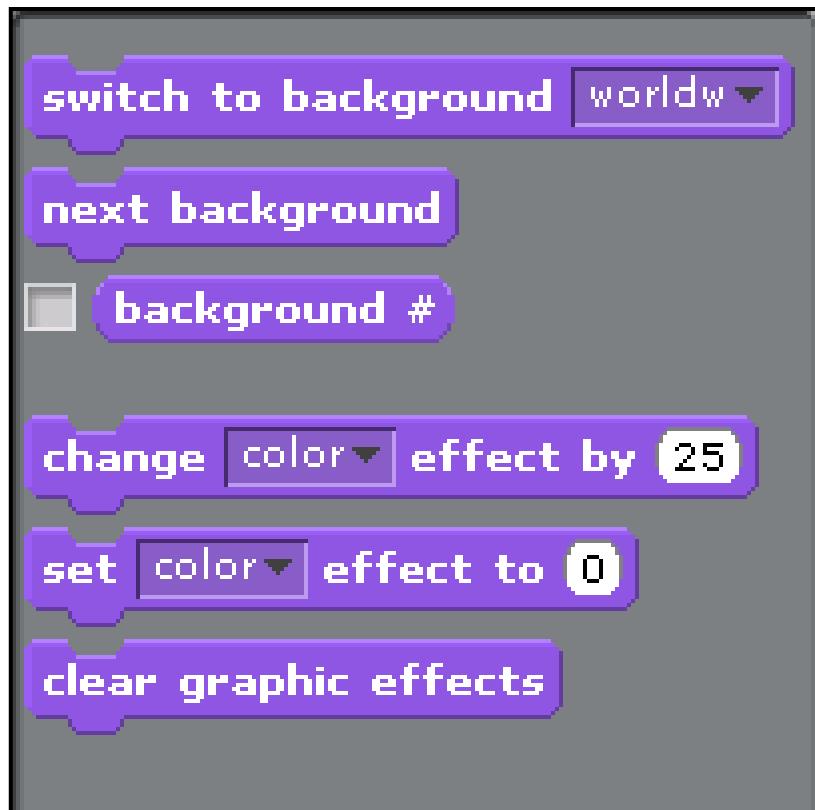


brightness



ghost

If You Have Selected a Background,
the Looks Menu Appears Like This



Oregon State

University

Computer Graphics

Sound Menu



Play a particular sound. Keep executing commands while the sound plays.

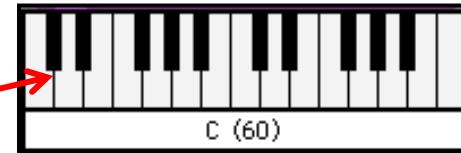
Play a particular sound. Pause until the sound is done playing.

Stop playing all sounds

Play a particular drum sound (35-81)

Stop playing the drum for a moment

Play a particular note



Set the instrument number 1-128

Change the volume (percent)

Set the volume (units are percent)

Display the volume in the scene

Change the tempo (+ or -, units are beats per minute)

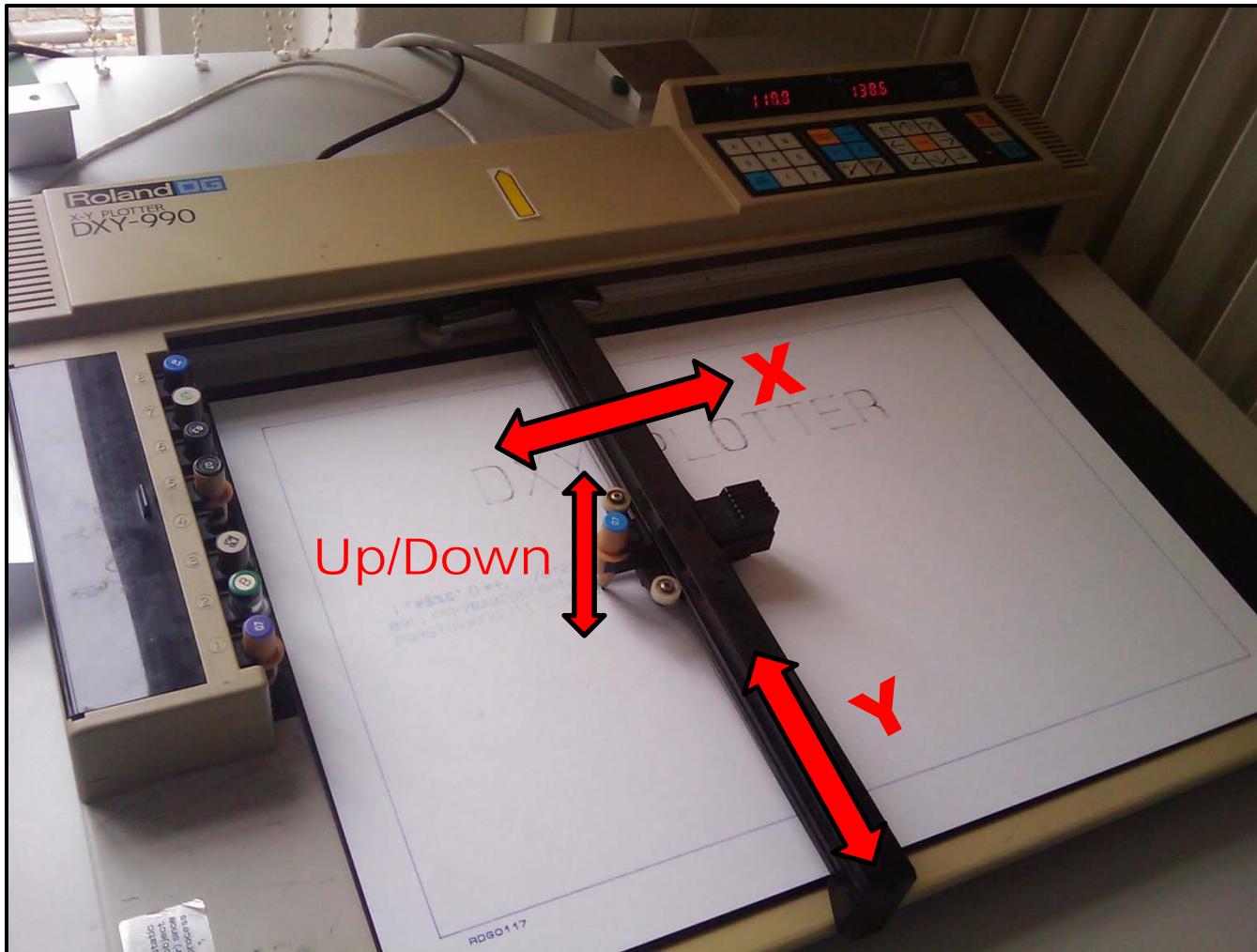
Set the tempo (units are beats per minute)

Display the tempo in the scene

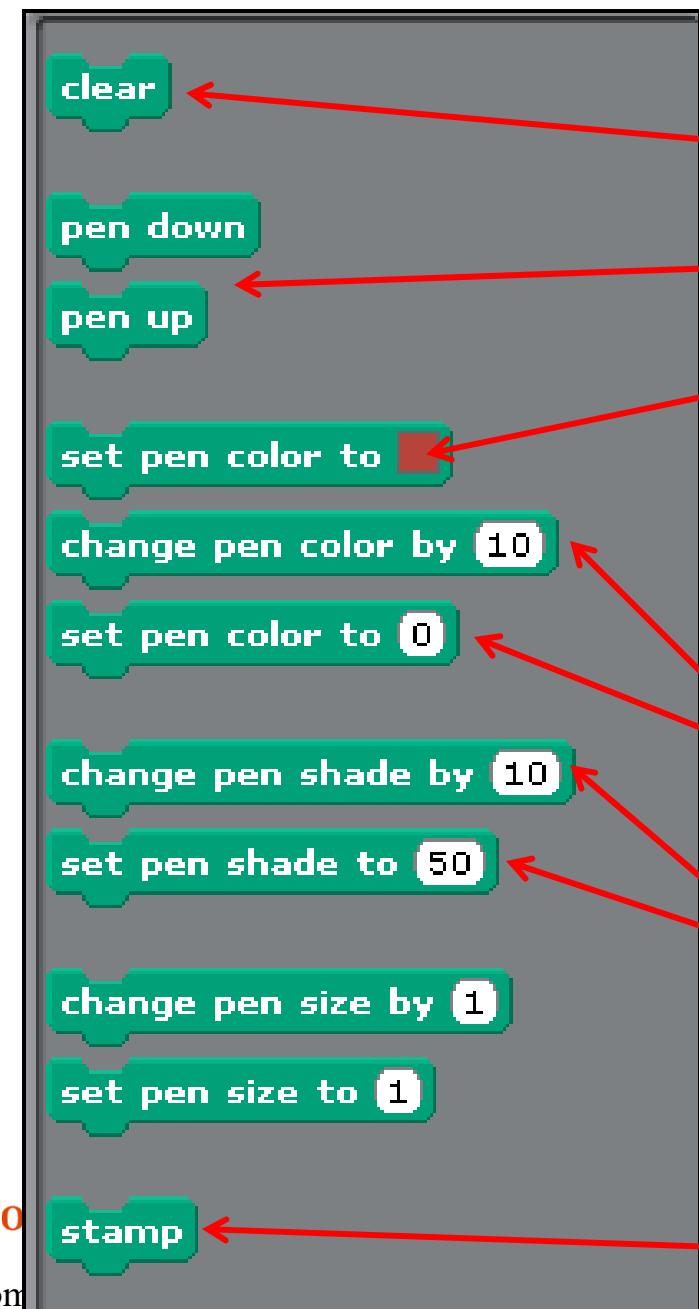
(35) Acoustic Bass Drum
(36) Bass Drum 1
(37) Side Stick
(38) Acoustic Snare
(39) Hand Clap
(40) Electric Snare
(41) Low Floor Tom
(42) Closed Hi-Hat
(43) High Floor Tom
(44) Pedal Hi-Hat
(45) Low Tom
(46) Open Hi-Hat
(47) Low-Mid Tom
(48) Hi-Mid Tom
(49) Crash Cymbal 1
(50) High Tom
(51) Ride Cymbal 1
(52) Chinese Cymbal
(53) Ride Bell
(54) Tambourine
(55) Splash Cymbal
(56) Cowbell
(57) Crash Cymbal 2
(58) Vibraphon
(59) Ride Cymbal 2
(60) Hi Bongo
(61) Low Bongo
(62) Mute Hi Conga
(63) Open Hi Conga
(64) Low Conga
(65) High Timbale
(66) Low Timbale
(67) High Agogo
more...

(1) Acoustic Grand
(2) Bright Acoustic
(3) Electric Grand
(4) Honky-Tonk
(5) Electric Piano 1
(6) Electric Piano 2
(7) Harpsichord
(8) Clavinet
(9) Celesta
(10) Glockenspiel
(11) Music Box
(12) Vibraphone
(13) Marimba
(14) Xylophone
(15) Tubular Bells
(16) Dulcimer
(17) Drawbar Organ
(18) Percussive Organ
(19) Rock Organ
(20) Church Organ
(21) Reed Organ
(22) Accordion
(23) Harmonica
(24) Tango Accordion
(25) Nylon String Guitar
(26) Steel String Guitar
(27) Electric Jazz Guitar
(28) Electric Clean Guitar
(29) Electric Muted Guitar
(30) Overdriven Guitar
(31) Distortion Guitar
(32) Guitar Harmonics
(33) Acoustic Bass
more...

A Real Pen Plotter



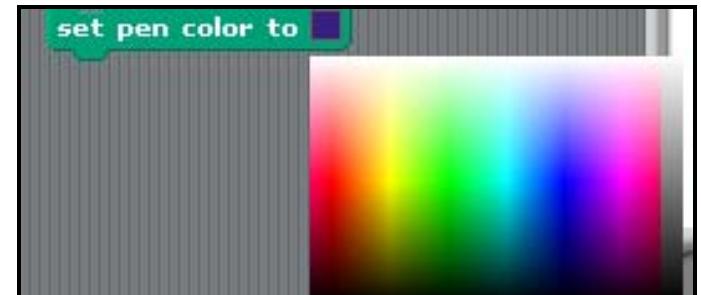
Pen Menu



Erase all previous plotter linework

Will the next pen movement leave a trace or not?

Click here to bring up a color palette to choose from



(See the next slide)

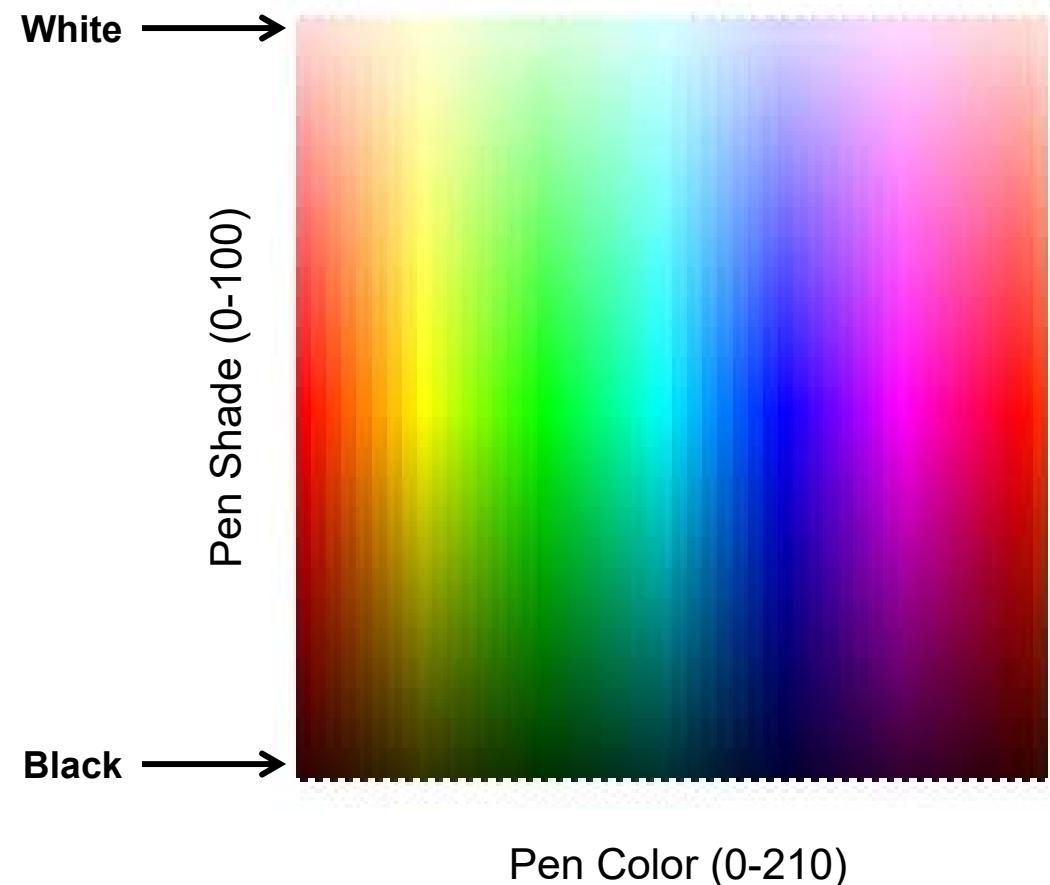
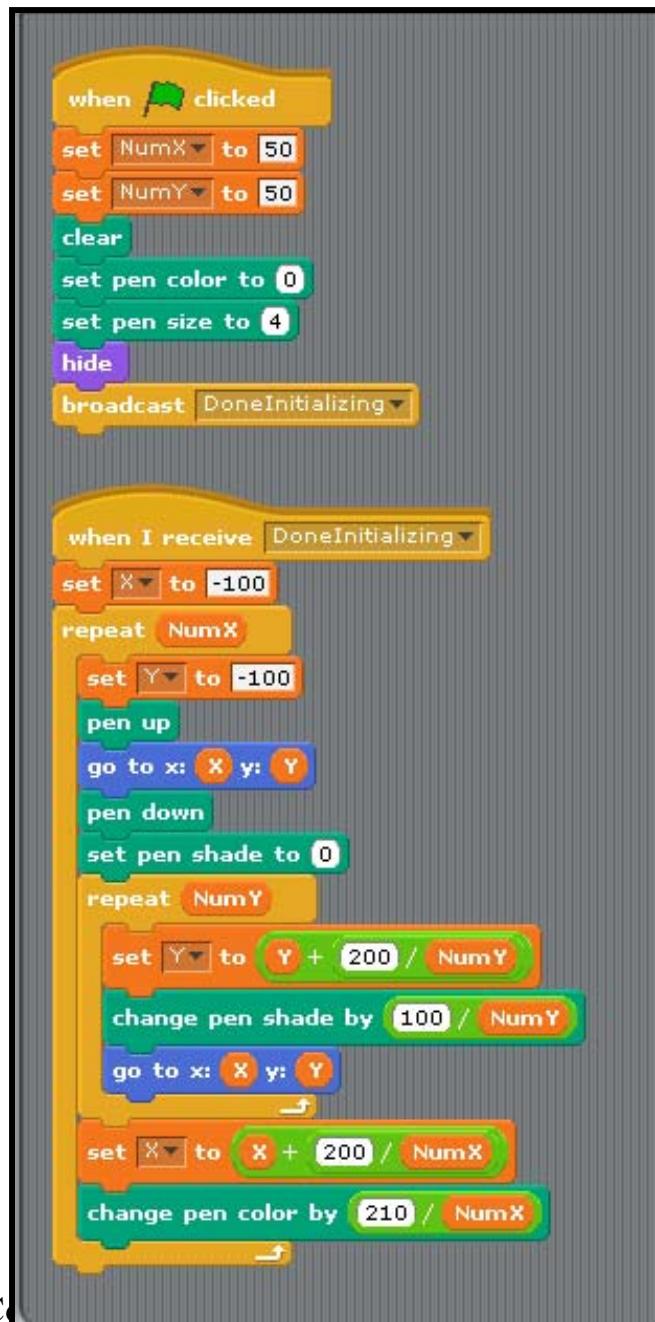
(See the next slide)

Control the width of the trace

Stamp a permanent copy of the sprite

The Meanings of “Pen Color” and “Pen Shade”

22

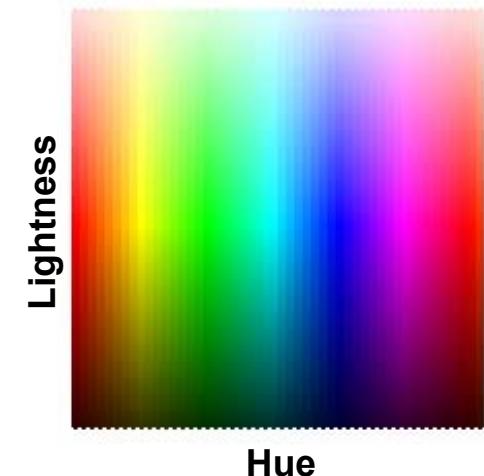
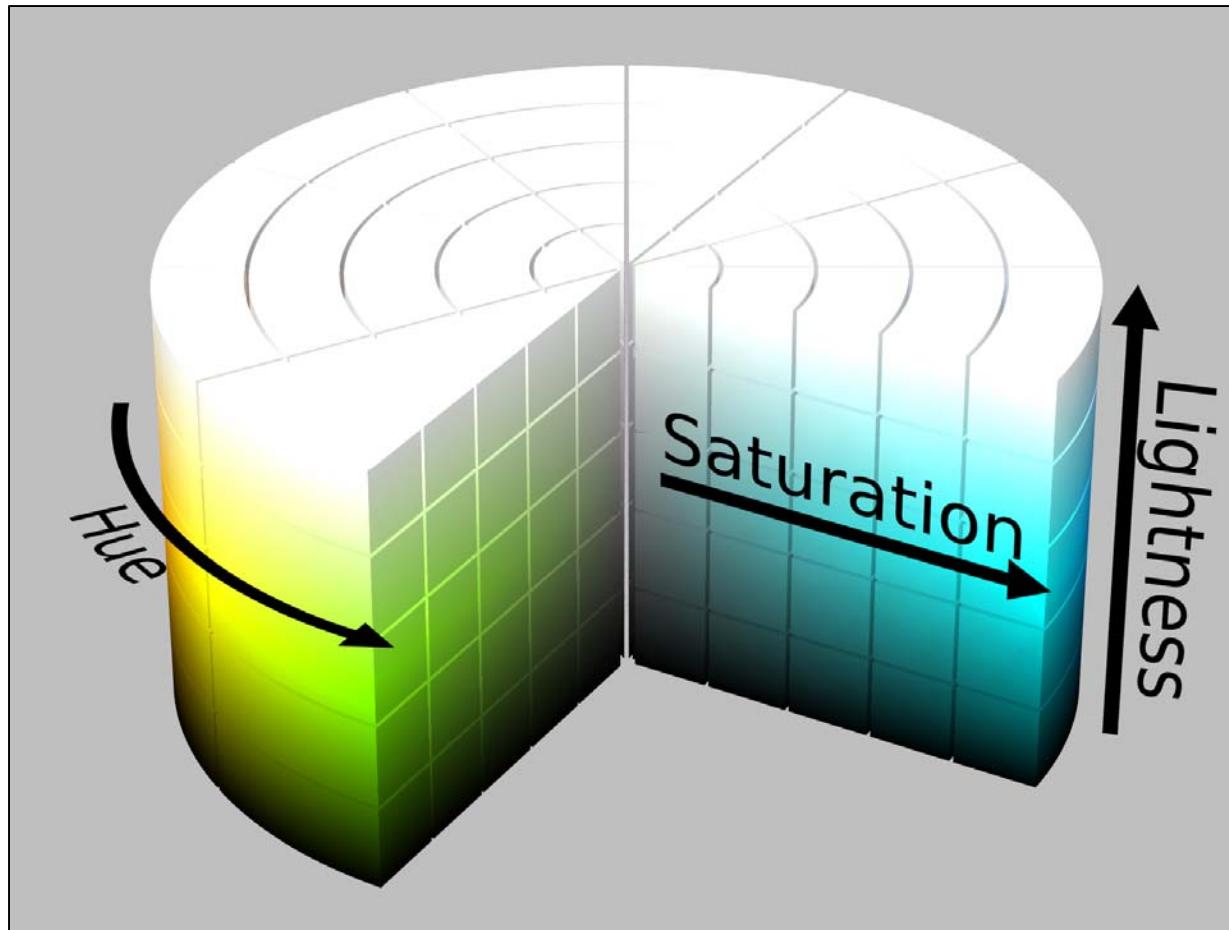


C

The Meanings of “Pen Color” and “Pen Shade”

23

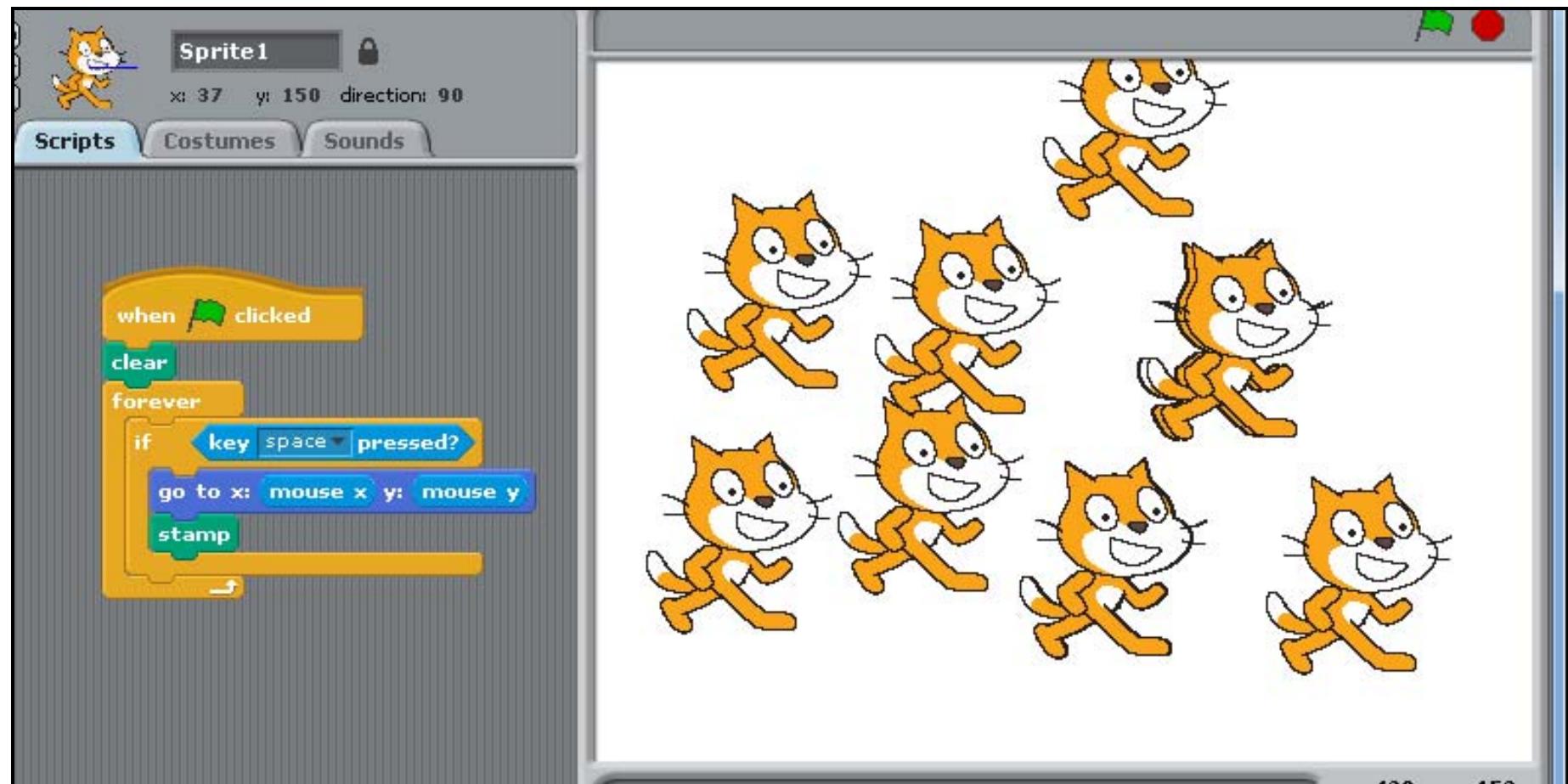
In computer graphics, this sort of color representation is referred to as Hue-Lightness. In the diagram below, the Scratch pen colors occupy the outer skin of the cylinder. Full color representation has a third component, Saturation, which Scratch doesn't give you access to.



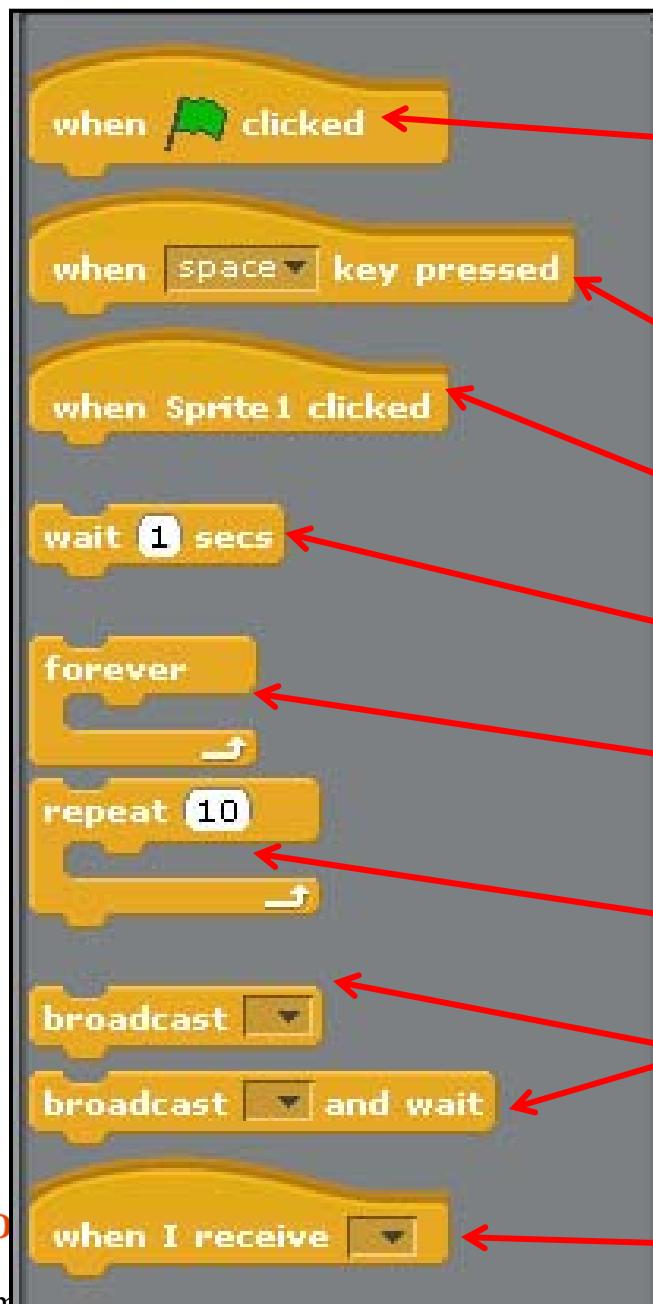
“Pen Color” = Hue.
“Pen Shade” = Lightness

The Meaning of “Stamp”

24



Controls Menu, I



When the green flag at the top of the screen is clicked, this part of the program will start executing. You can have as many of these as you want.

When a particular key is pressed, this part of the program will start executing. You can have as many of these as you want.

When this sprite is clicked, this part of the program will start executing.

Pause

Loop forever

Loop a particular number of times

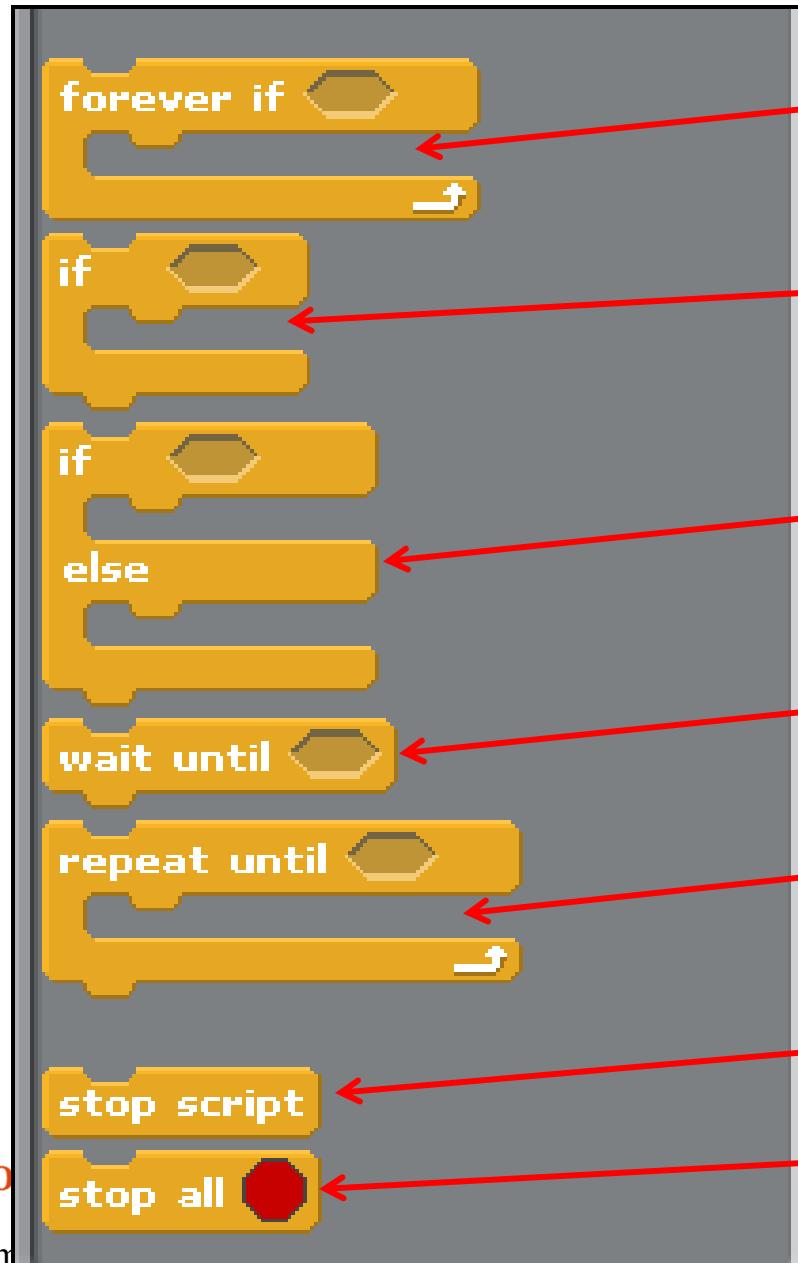
Send a signal out to everyone. You get to pick the name. The “wait” option will wait until everyone has actually received the signal.

When this signal is broadcast, this part of the program will start executing. You can have as many of these as you want.



Controls Menu, II

26



Loop forever if the given condition is **true**

Execute this block of statements if the given condition is **true**

Execute the first block of statements if the given condition is **true**, otherwise execute the second

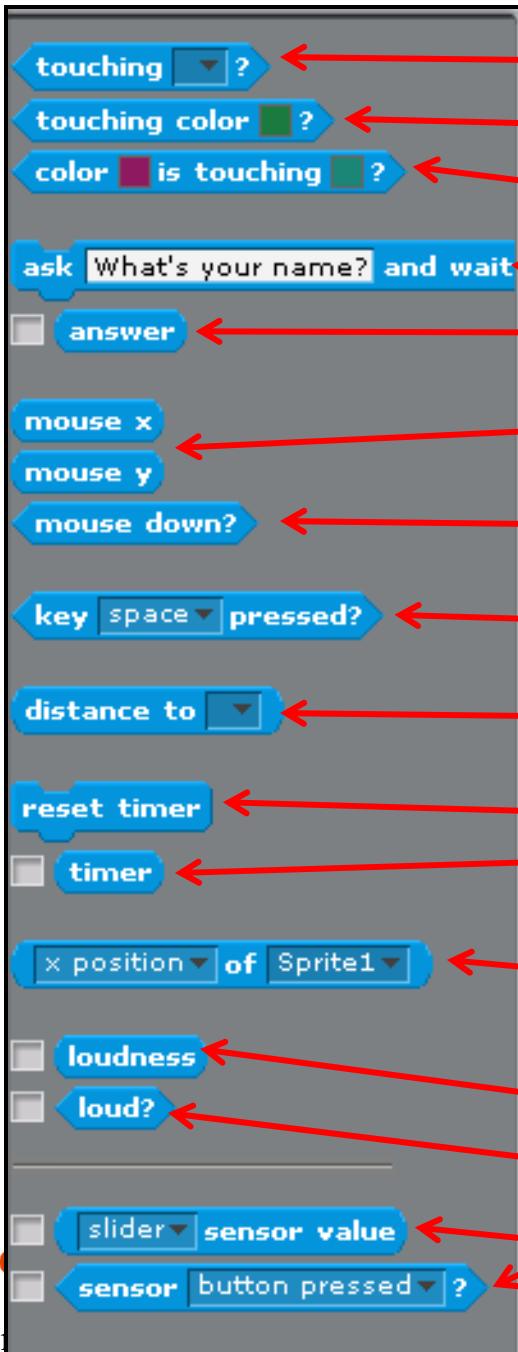
Pause until the given condition is **true**

Loop until the given condition is **true**

Stop this script

Stop all scripts

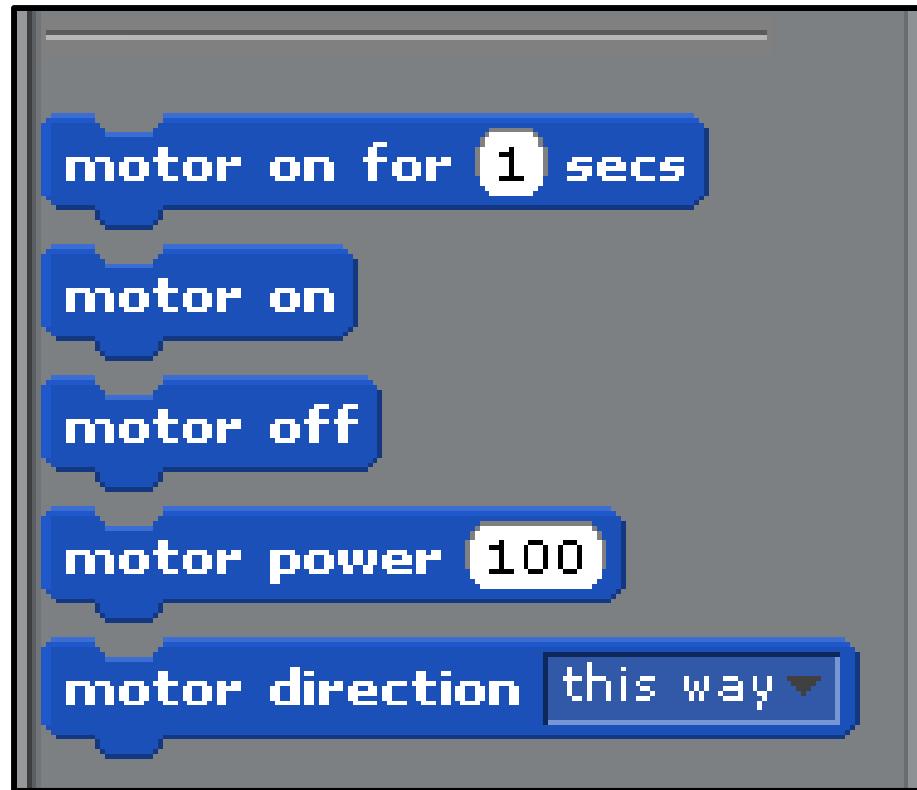
Sensing Menu



Co

If the Motor Controls are Turned On, You Also get...

28



Oregon State

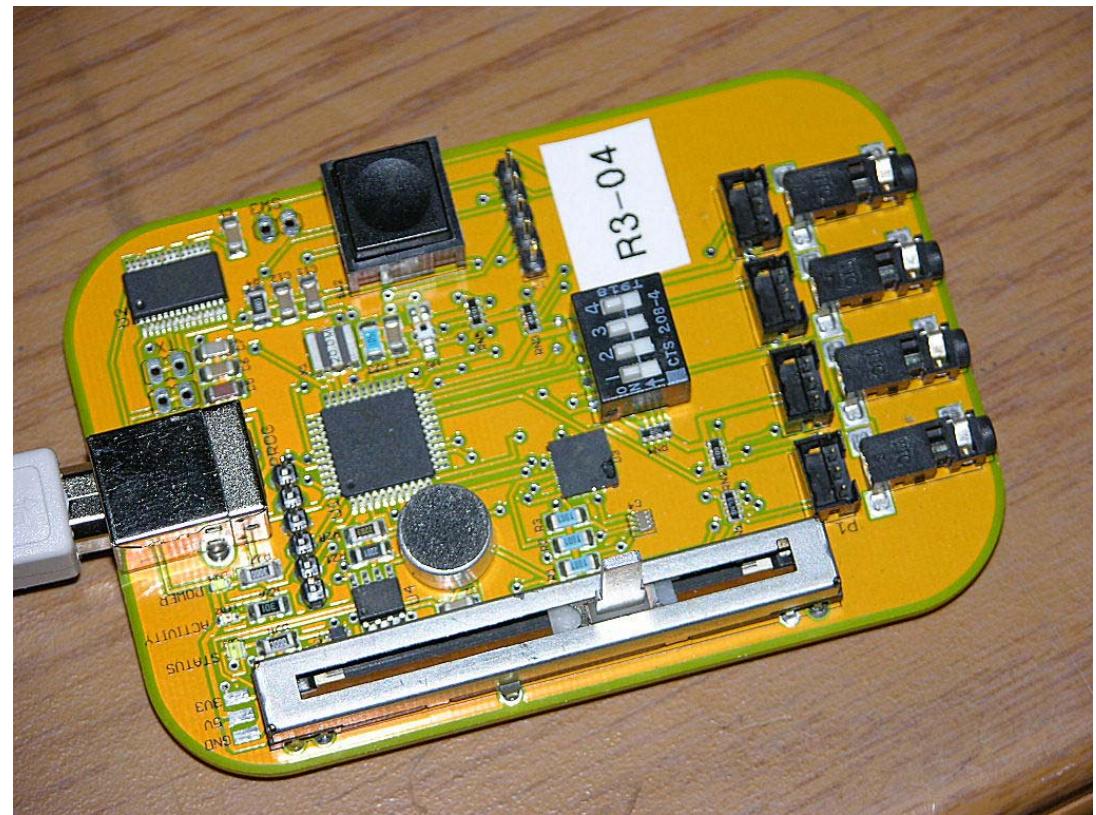
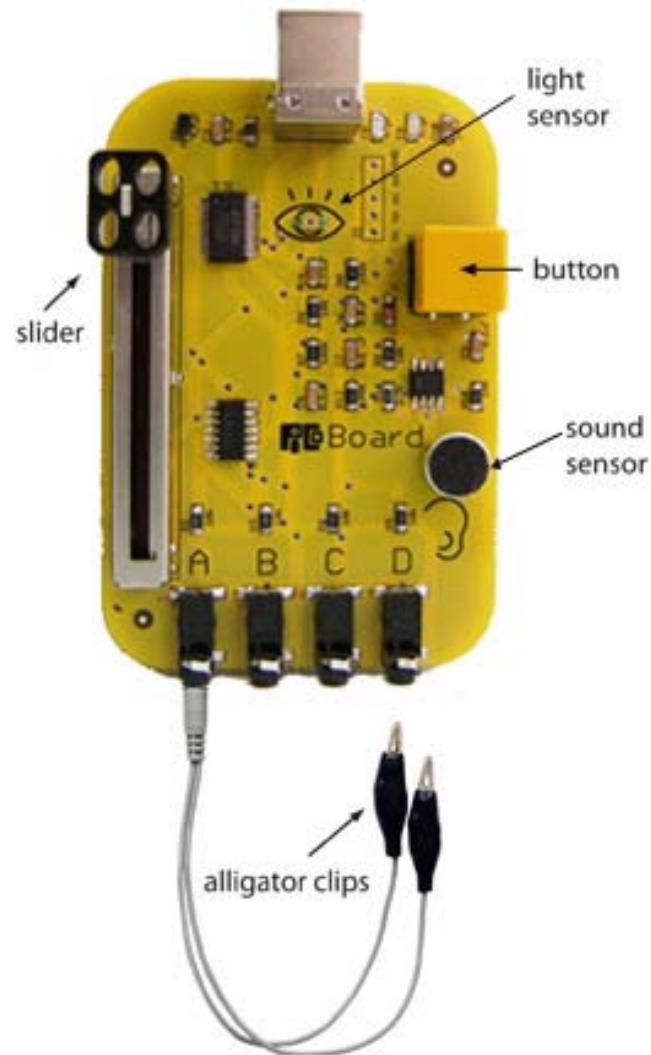
University

Computer Graphics

mjb – August 1, 2018

The Scratch Board

29



The Lego WeDo Construction Set

30



<http://education.lego.com/en-gb/lego-education-product-database/wedo/9580-lego-education-wedo-construction-set>

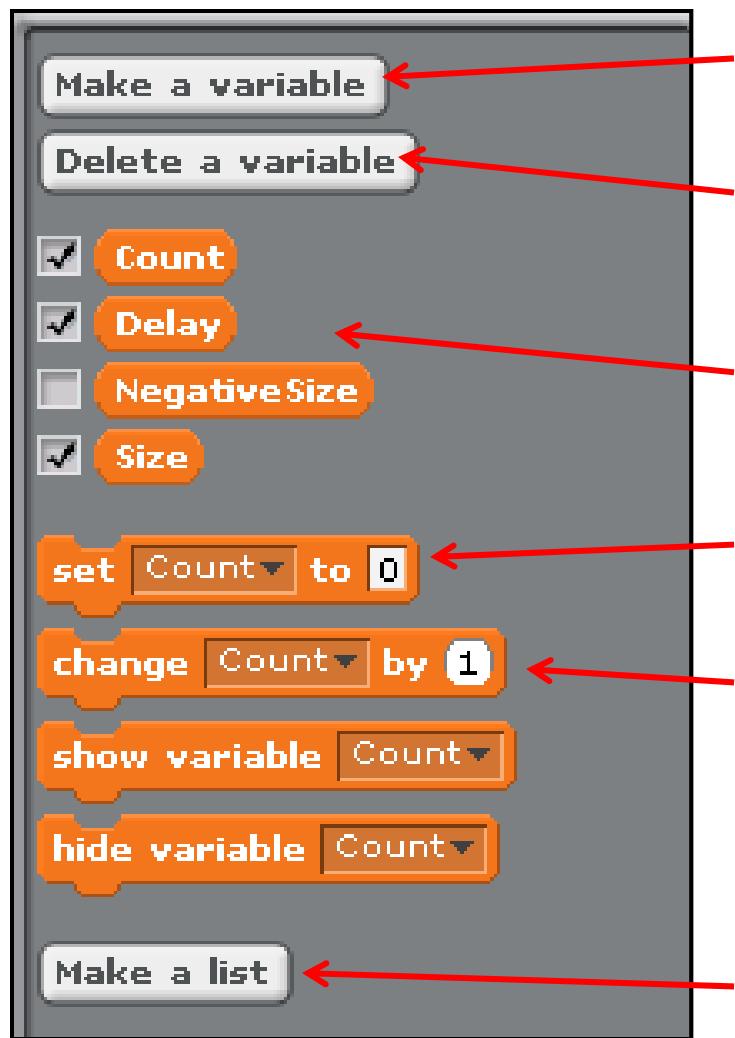


Oregon State
University

Computer Graphics

mjb – August 1, 2018

Variables Menu



Create a new variable – you will be asked to give it a name

Choose which variable to delete from a list of variables

These are variables I had created at the time.
The checkboxes say whether or not you want them to be displayed in the scene.

Used to set the variable to an initial value

Change the variable's value by a certain amount (plus or minus)

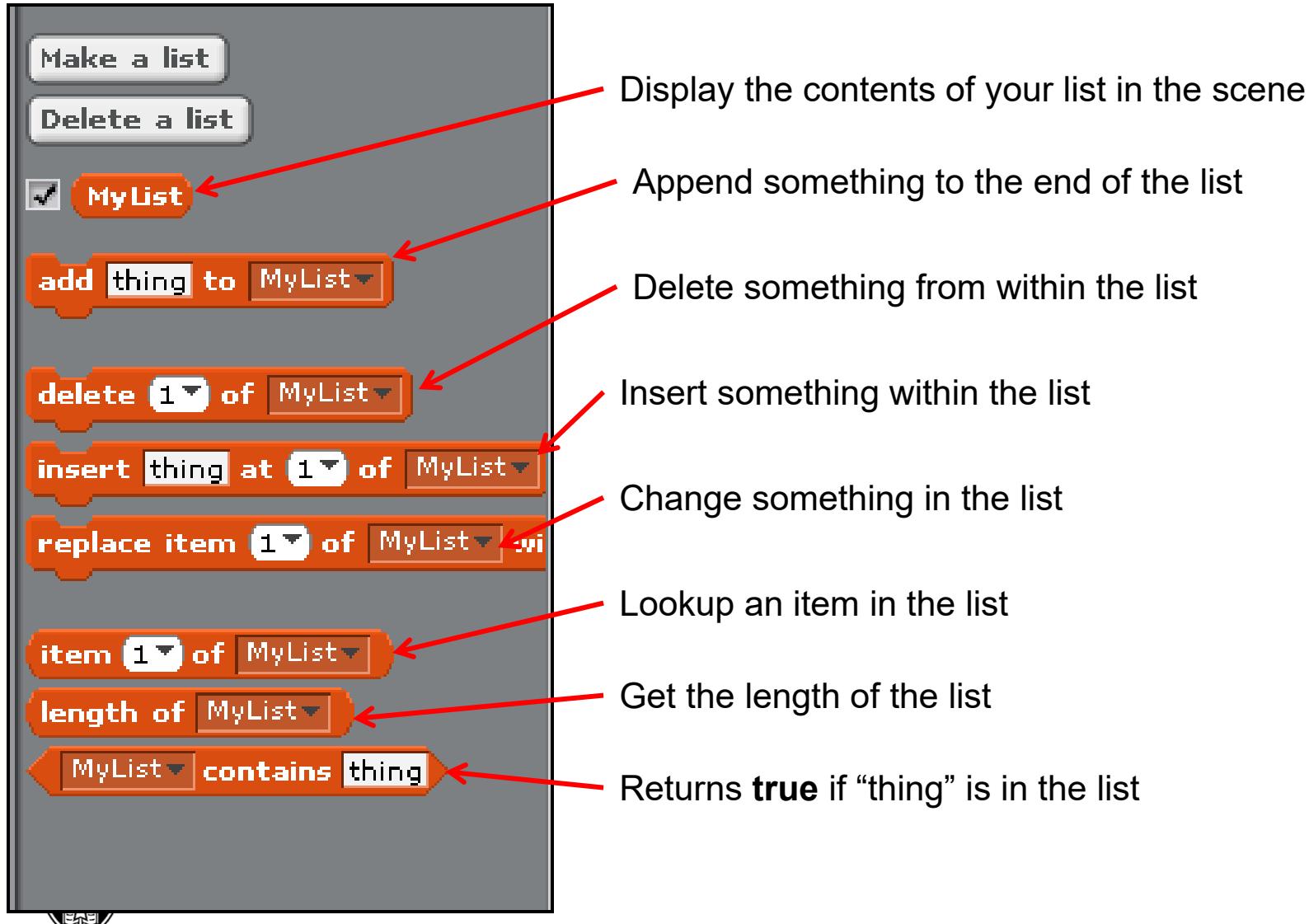
Start a list. A list is a multi-value variable (like an array in C and Java).



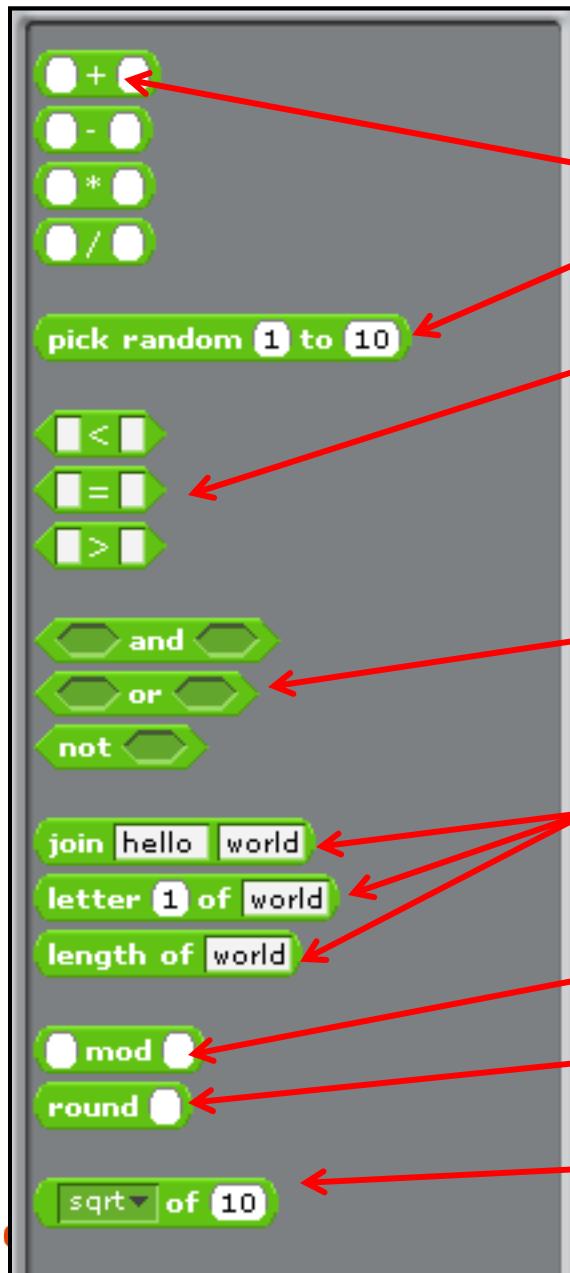
Oregon State

University

Computer Graphics



Operators Menu



Placeholders like these can be numbers or variable names

Random number generator. You select the range.

These return **true** or **false**, and are typically used in **if**, **forever-if**, and **repeat-until** statements

These are used to link multiple **true/false**-producing statements together

String manipulation (good for creating strings for the sprite to say or think)

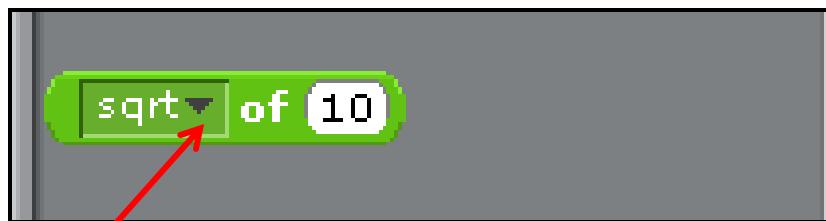
Modulo function (i.e., the remainder after a division is done)

Round to the nearest integer

(See the next slide)

Operators Menu: Math Functions

34



Click here to select from a list of math functions:



Absolute value

Square root

sin

cos Angles are all in *degrees!*

tan

asin

acos

atan

ln

Natural log

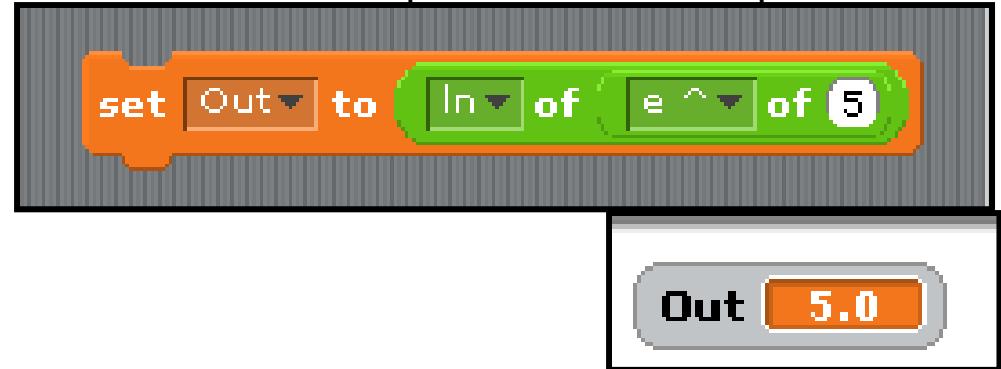
log

Base 10 log

$e^$

$10^$

You can embed an operator within an operator:

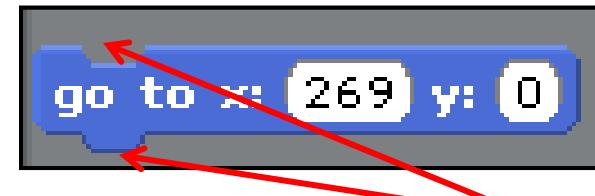
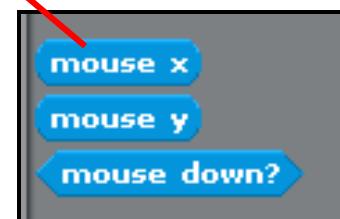
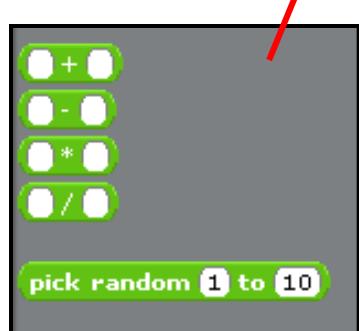


Shape Encoding within Scratch

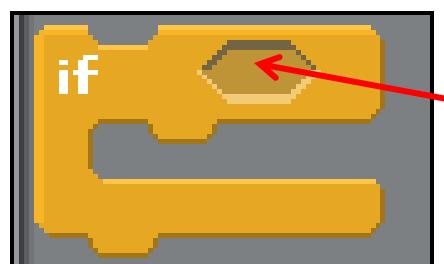
35



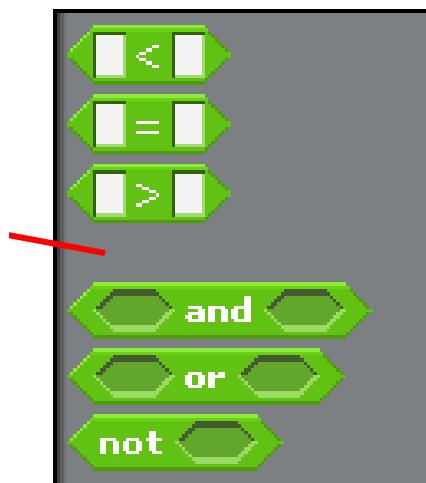
Ovals are a place to put numeric constants and variables



Puzzle piece interlocks on the top and bottom show where connections can be made. A command without an interlock on the top is an **Event**.



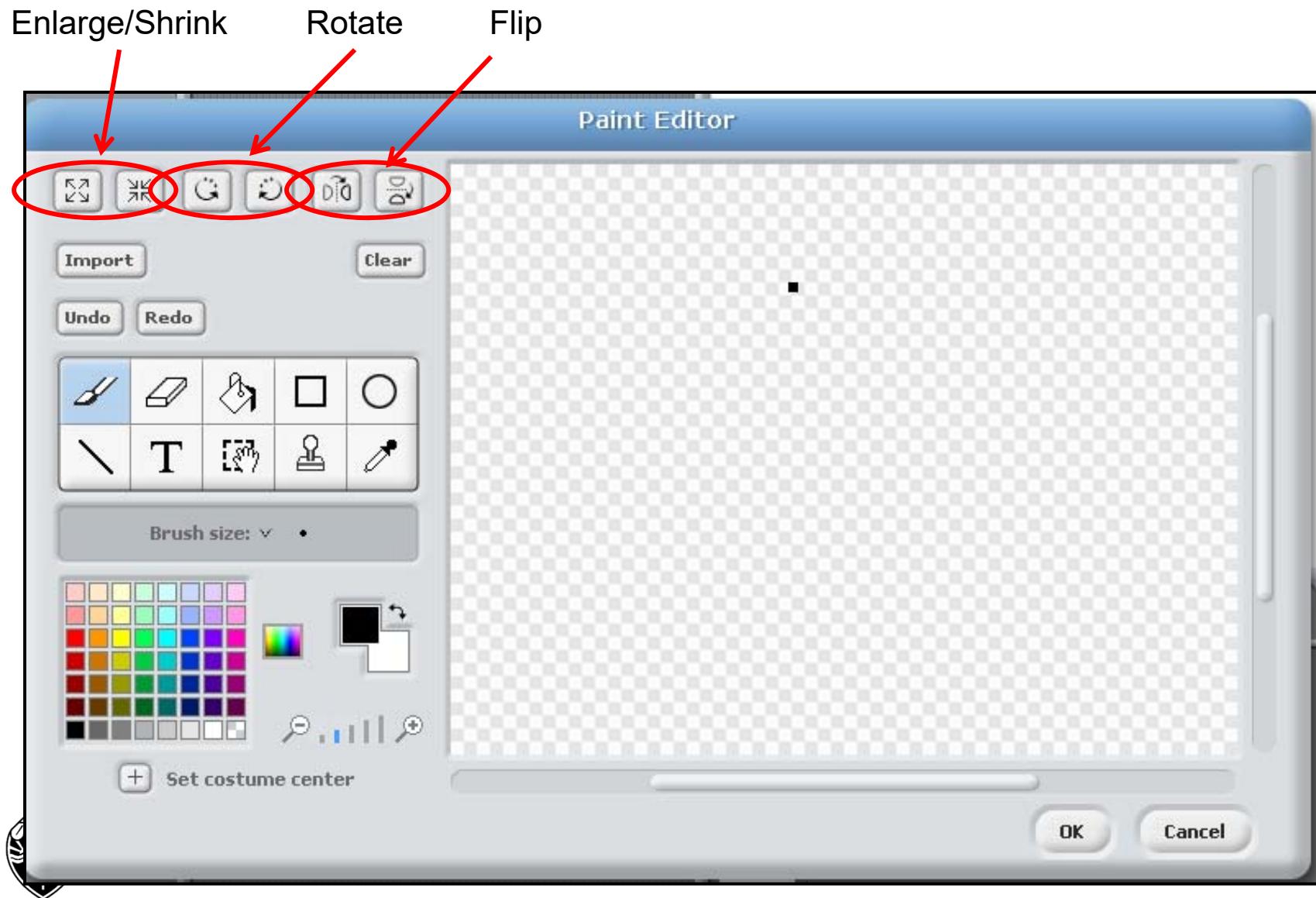
Hexagons are a place to put Boolean values (i.e., values that can be either **true** or **false**)



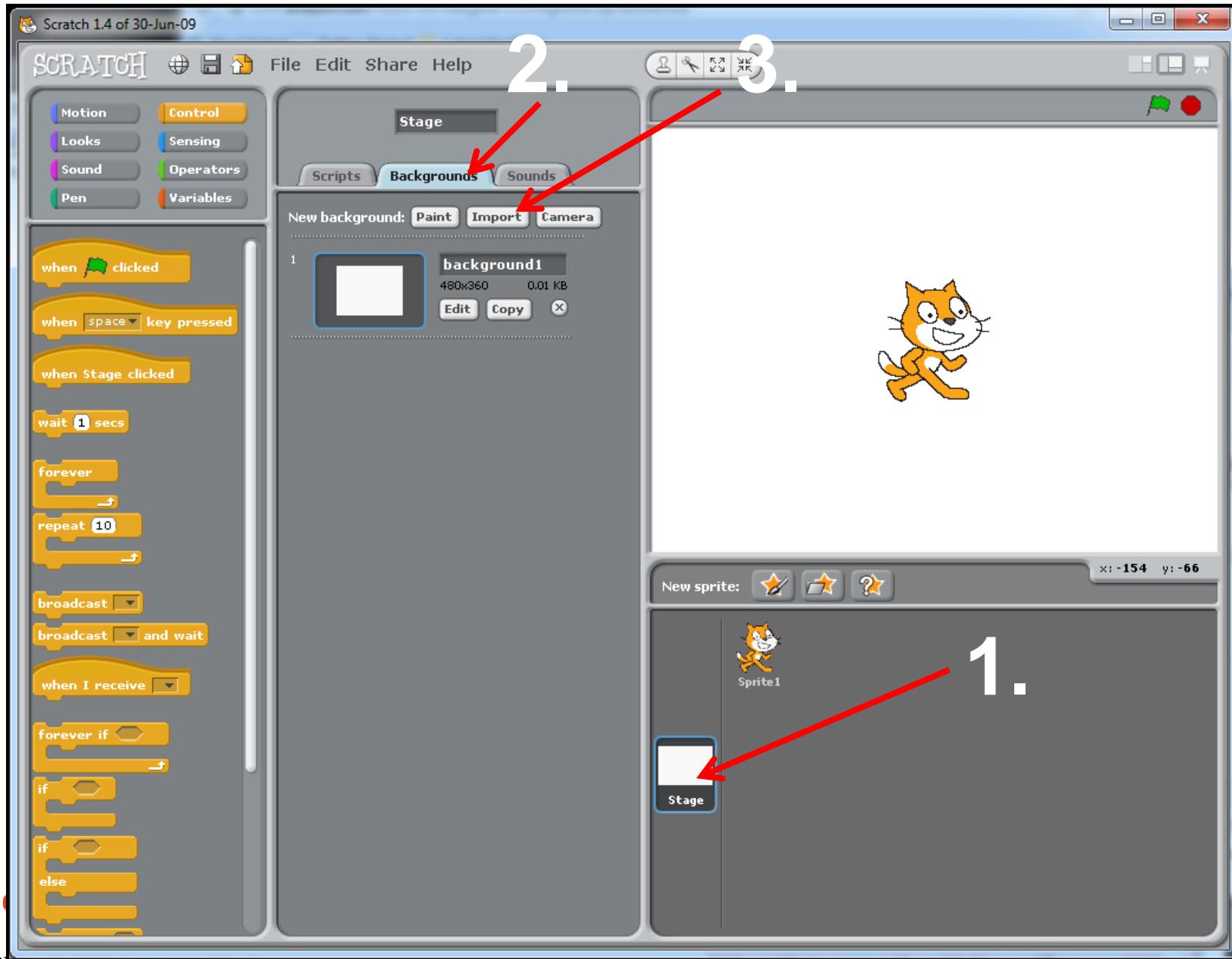
Oregon State
University

Computer Graphics

The Paint Editor

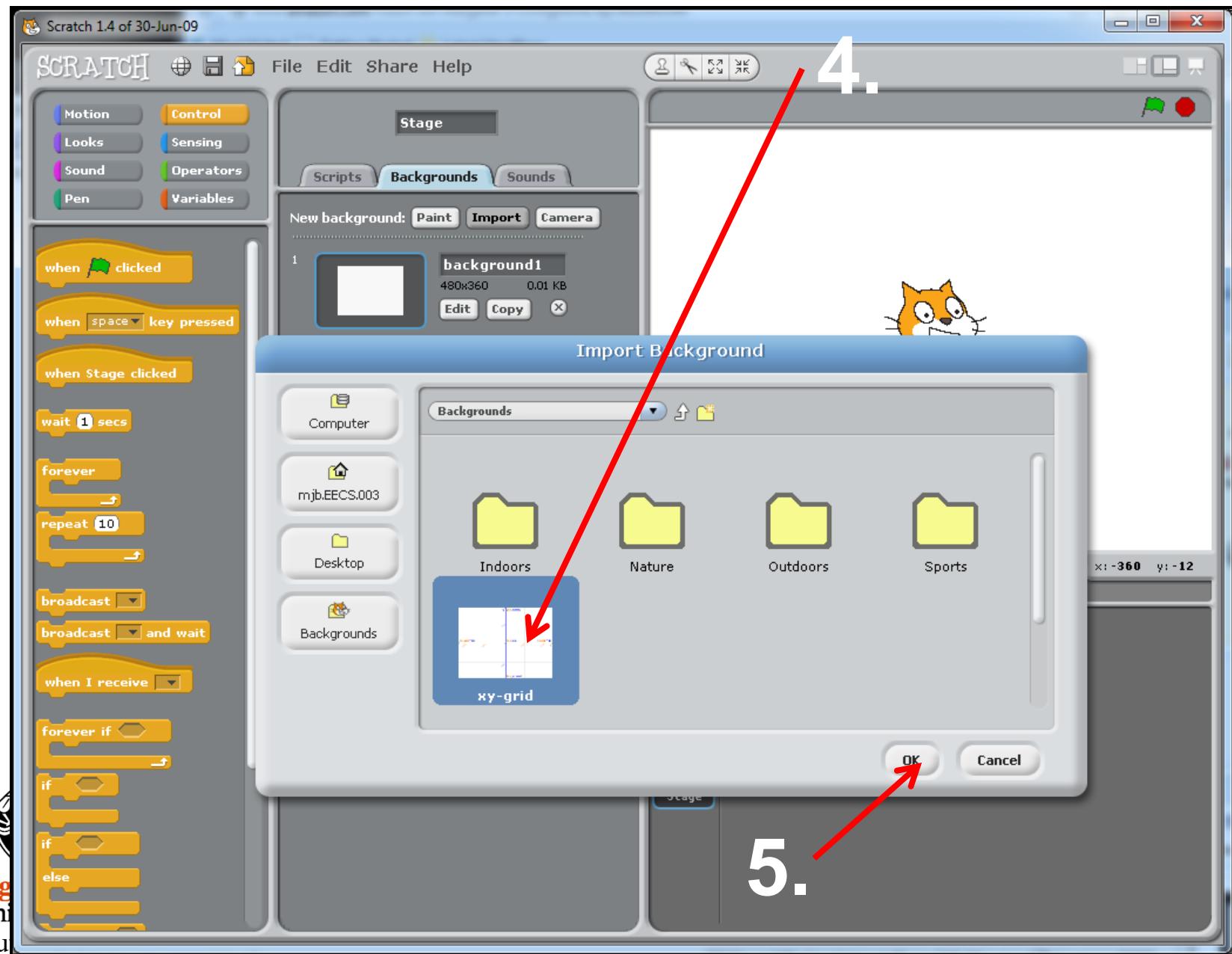


Setting the Background to a Handy X-Y Grid

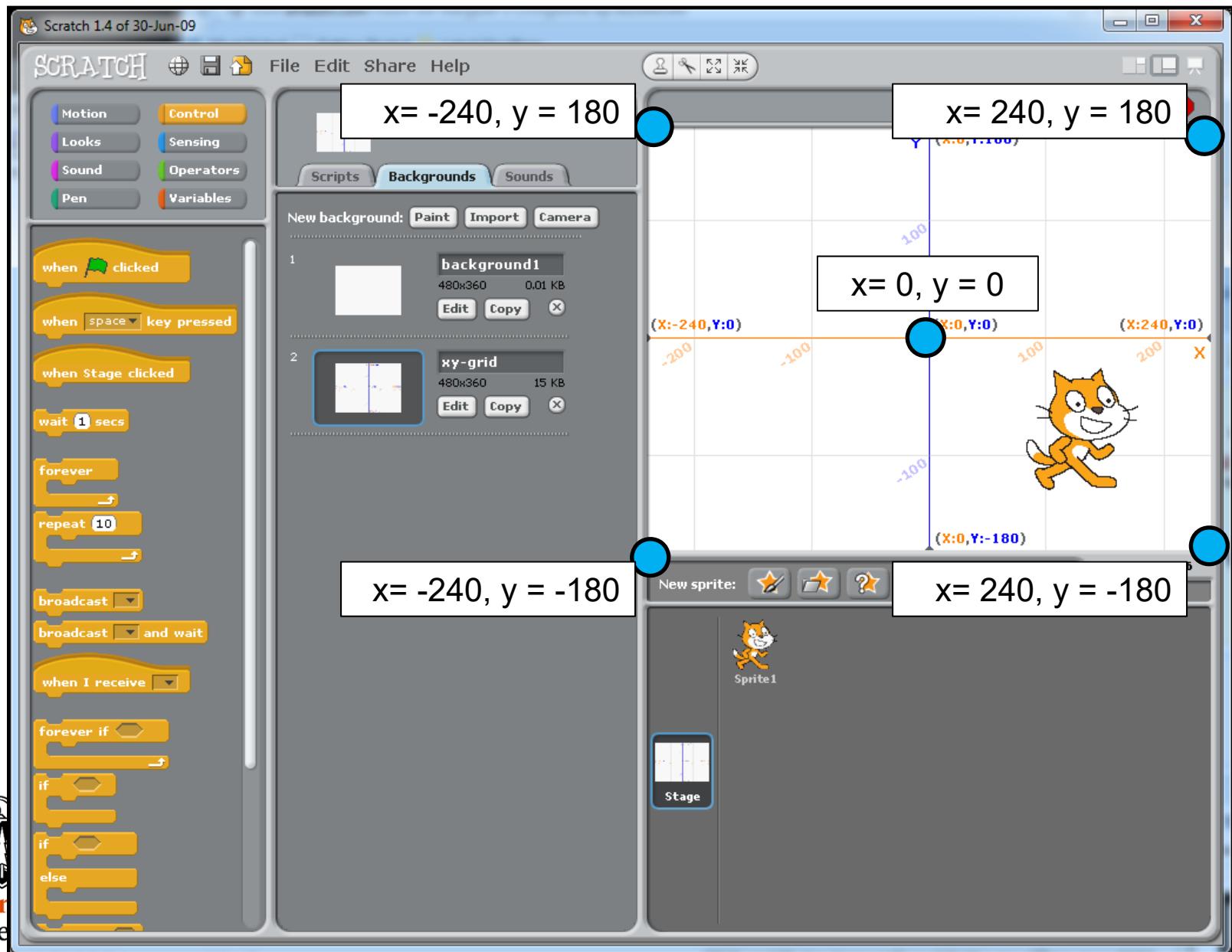


Setting the Background to a Handy X-Y Grid

38

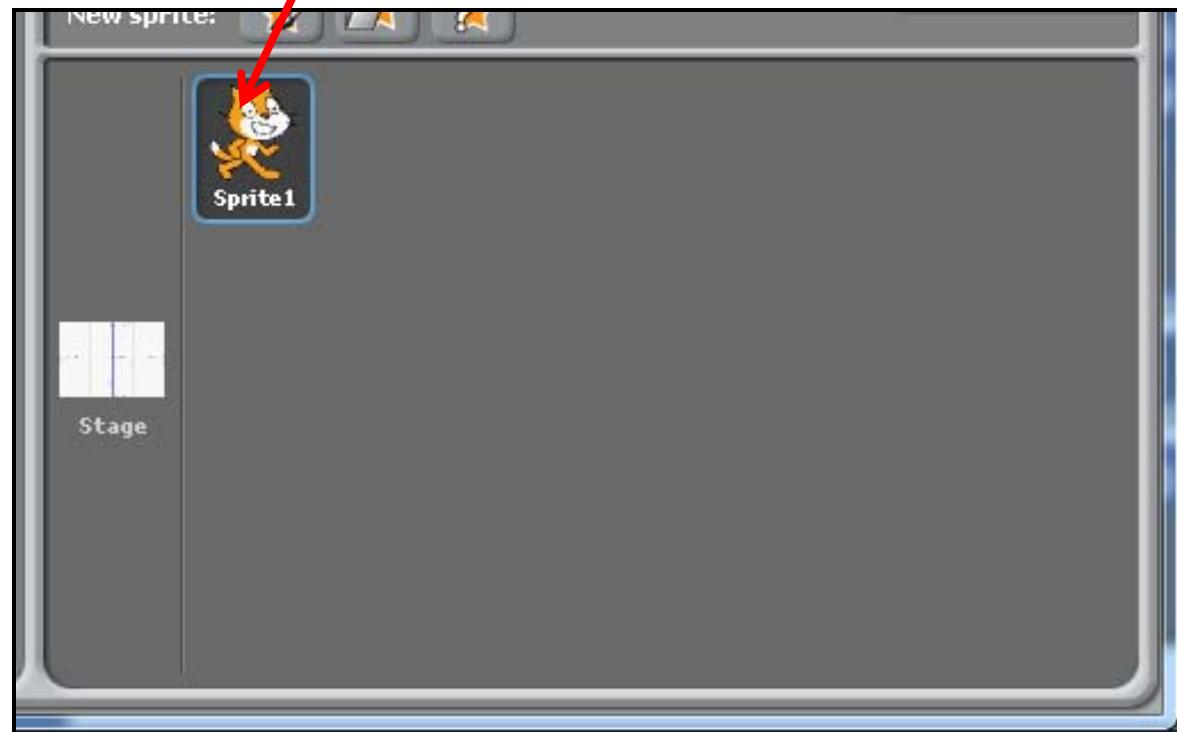


This is What the X-Y Grid Looks Like – It is Useful When you are Creating your Animation



Work with the Cat Sprite Again

Click on the Cat sprite to get back to working with it



Oregon State

University

Computer Graphics

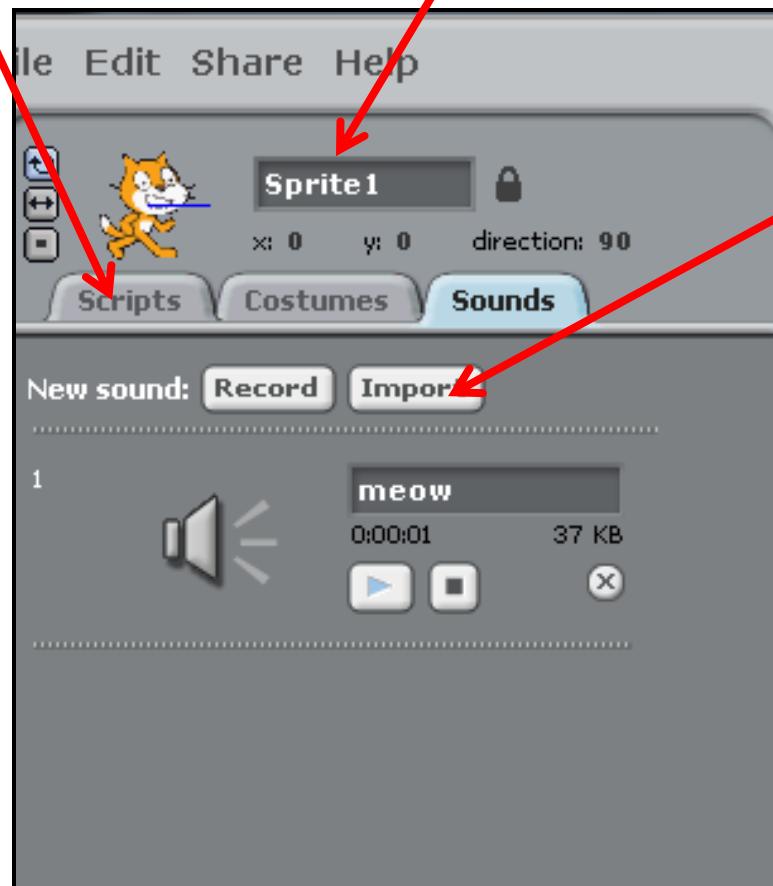
We Want a Sound to Confirm When Our Program is Done

41

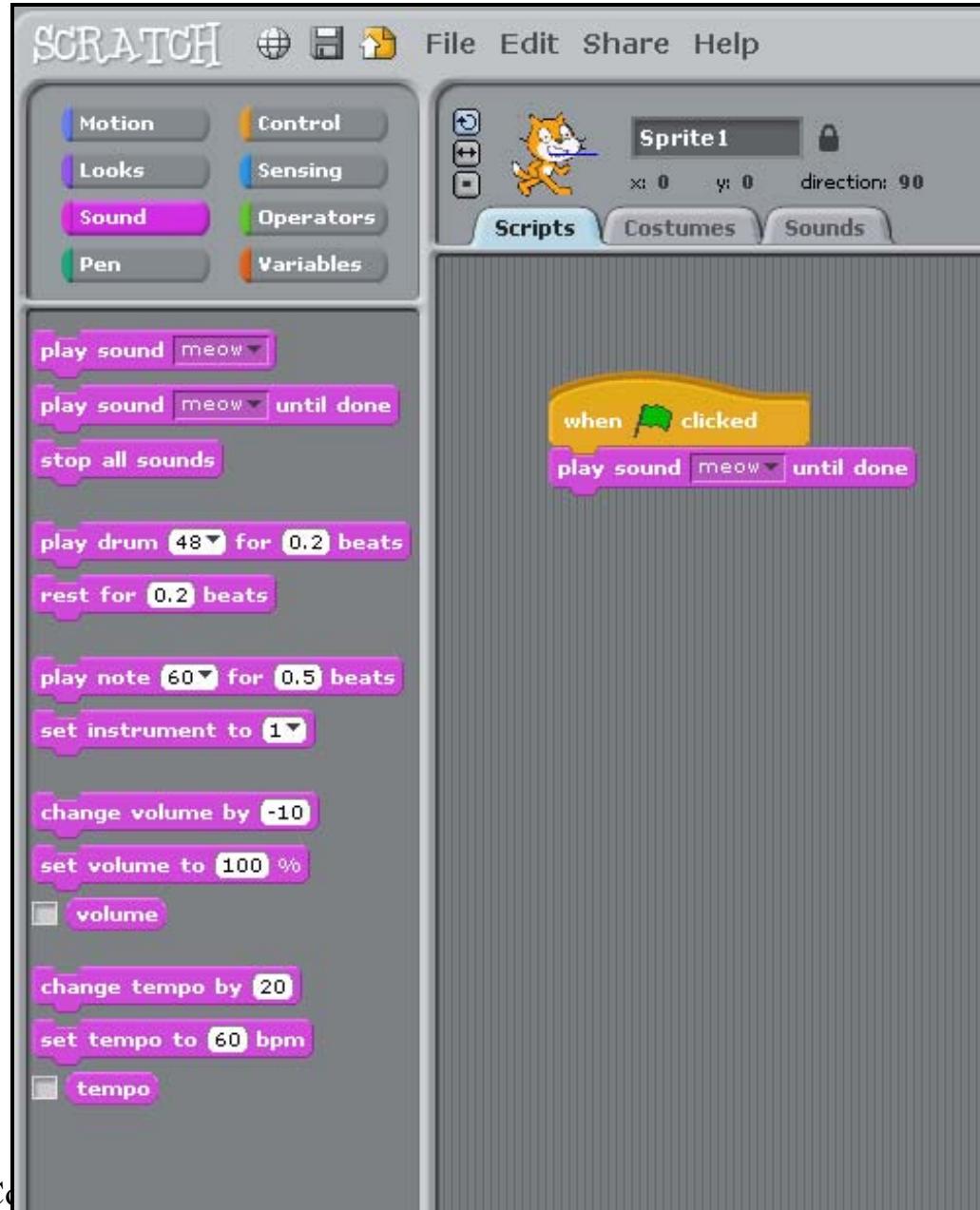
3. Click on the Scripts tab to go back to creating your program.

1. Click on the Sounds tab and verify that there is already a sound named “meow”.

2. If there isn’t, click on the Import button and load the sound.



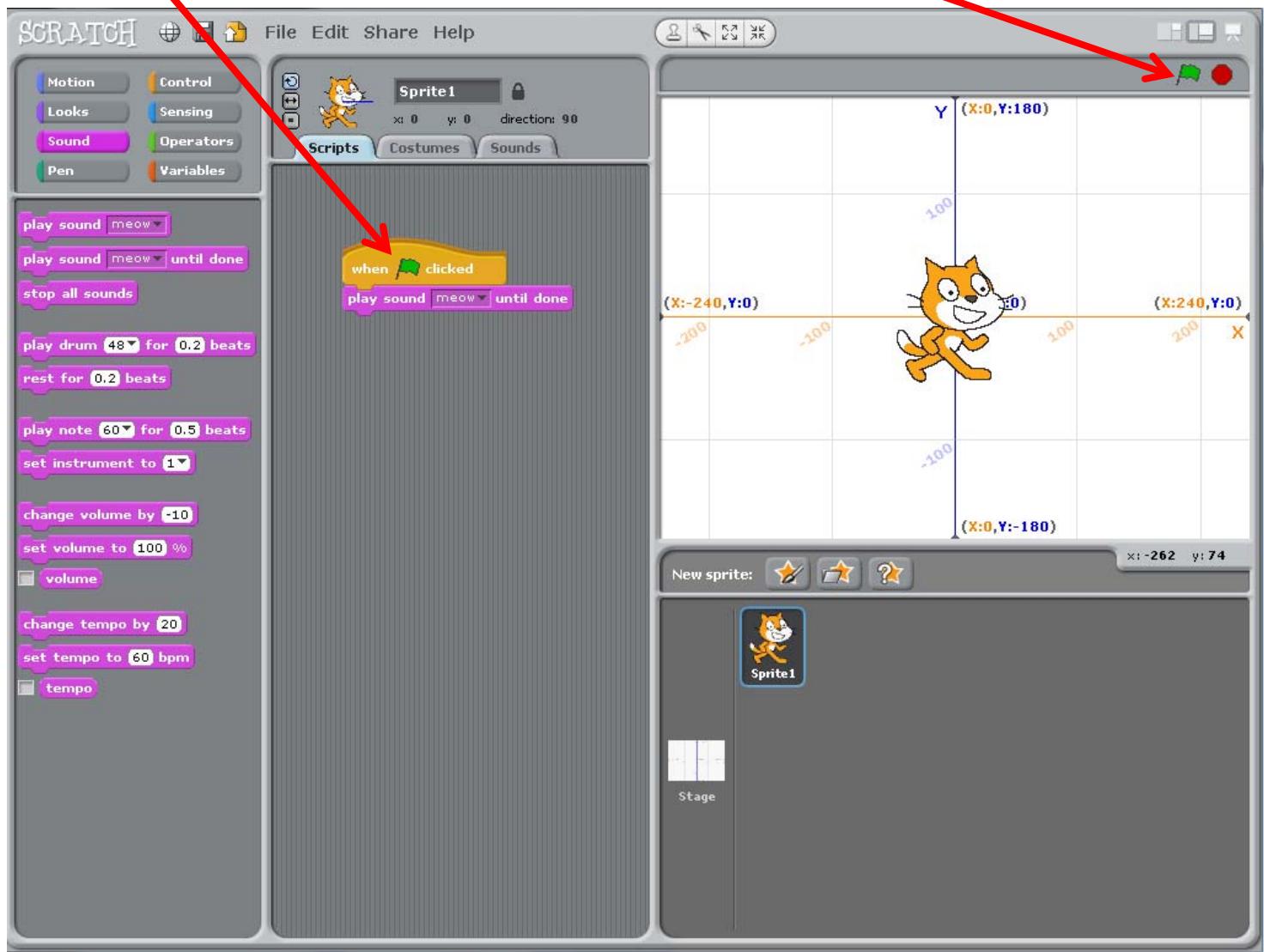
Drag Each of These Commands From Its Menu and Hook Them Together⁴²



Running Your Program

2. . . triggers this, and everything below it

1. Clicking here . . .



Oregon State
University

Computer Graphics

Bring in the Sound-playing Command, and then Drag Each of These 44 Commands From Its Menu and Hook Them Together

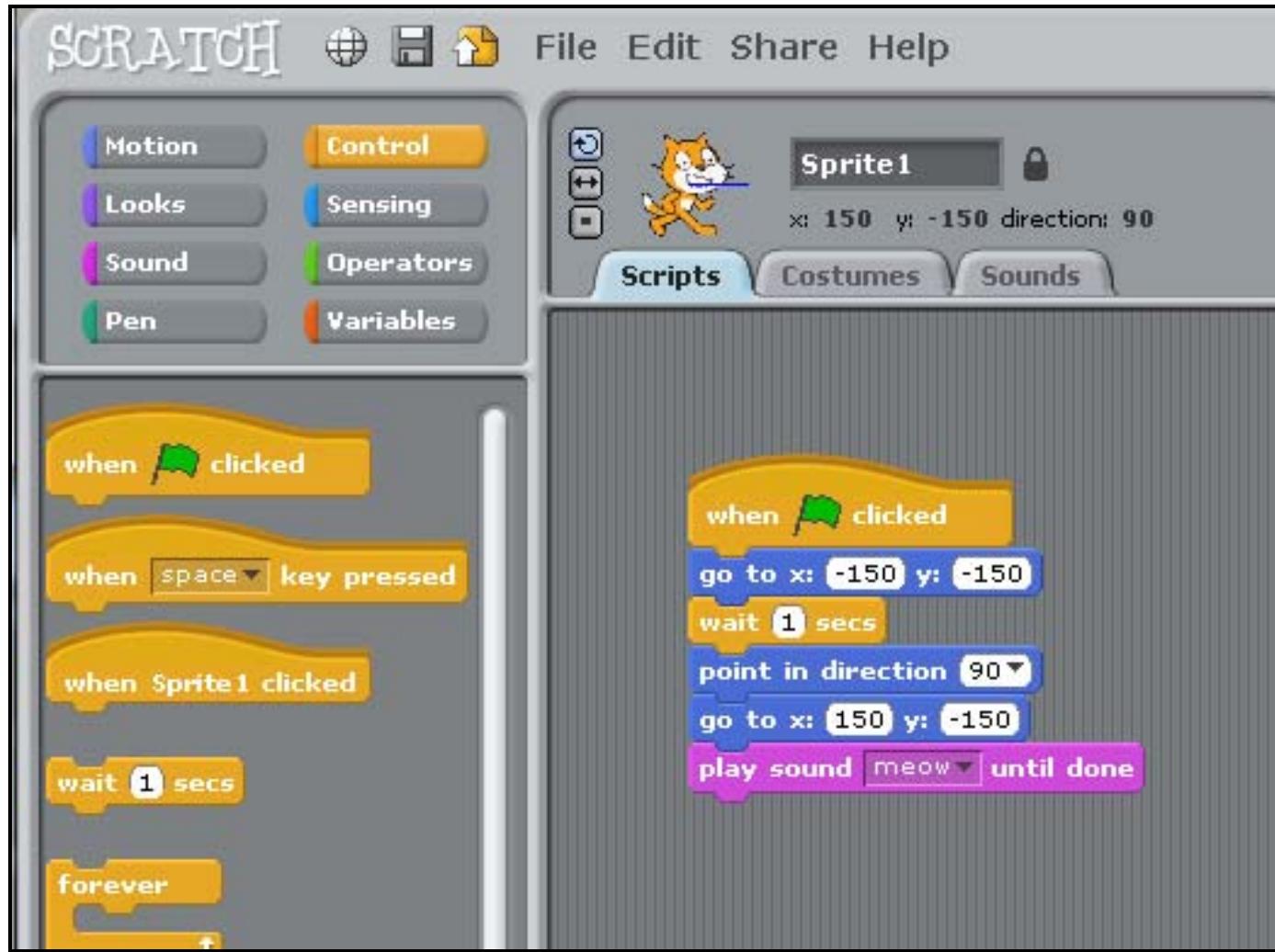


Oregon State
University
Computer Graphics

Run this program! What happens? It's probably not what you were hoping for. Why? (Hint: because you told it to.)

Add a “wait” after the first “go to”

45



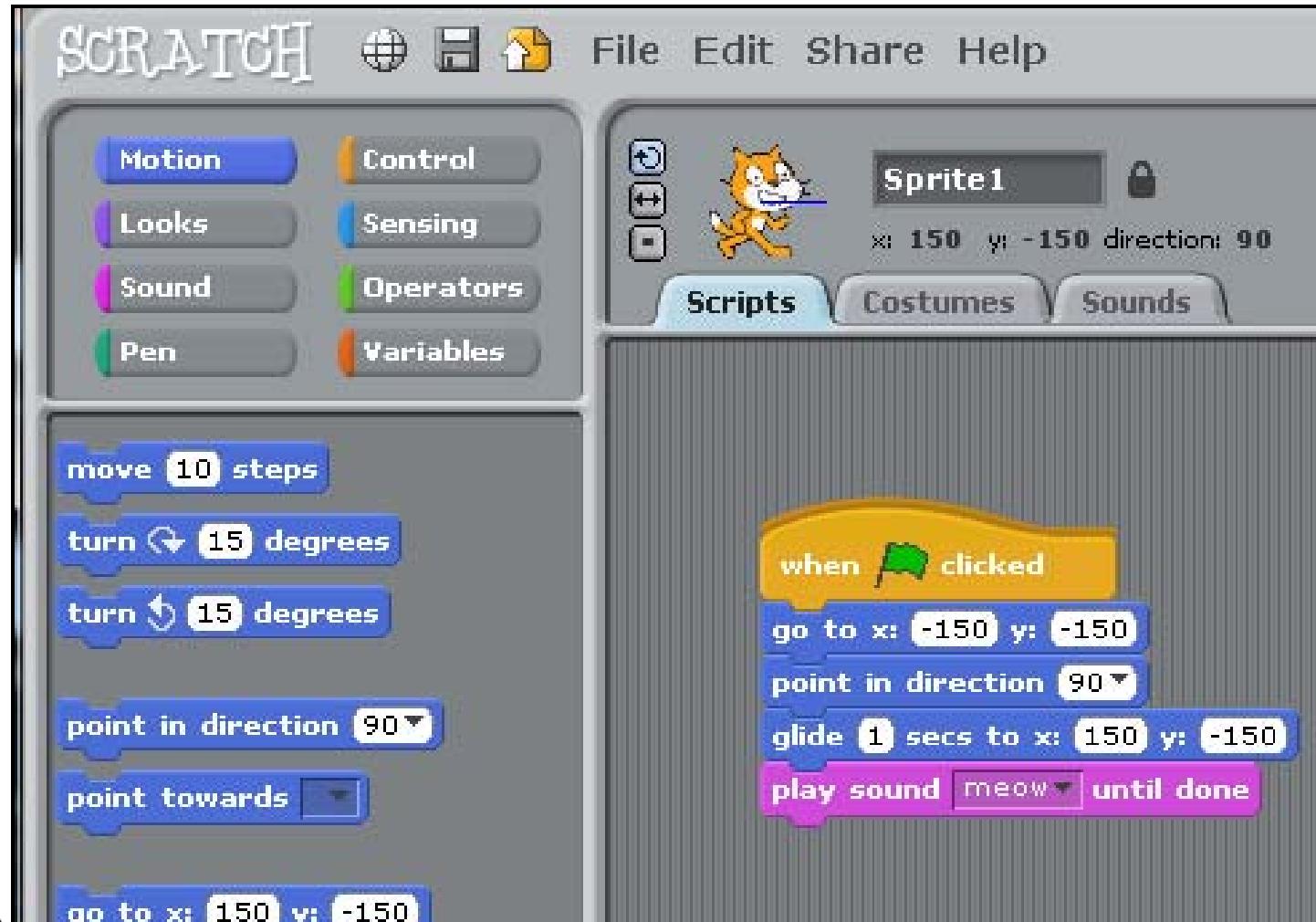
Oregon State
University

Computer Graphics

Now what happens? It's still probably not what you were hoping for. Why?

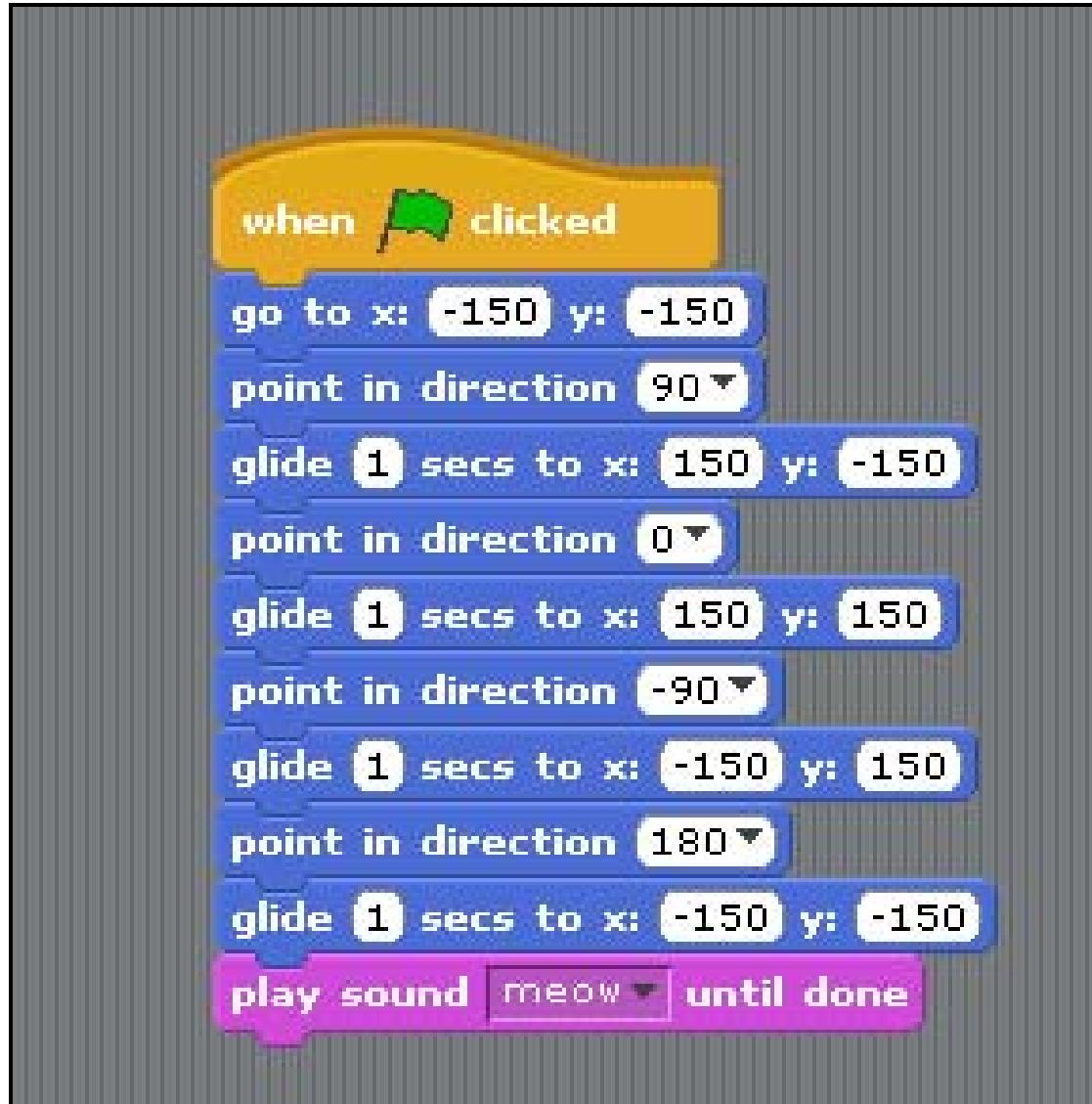
mjb – August 1, 2018

Get rid of the “wait” and change the second “go to” to a “glide” 46



Now what happens? This is probably the effect you were hoping for.

Have the Cat Run a Lap



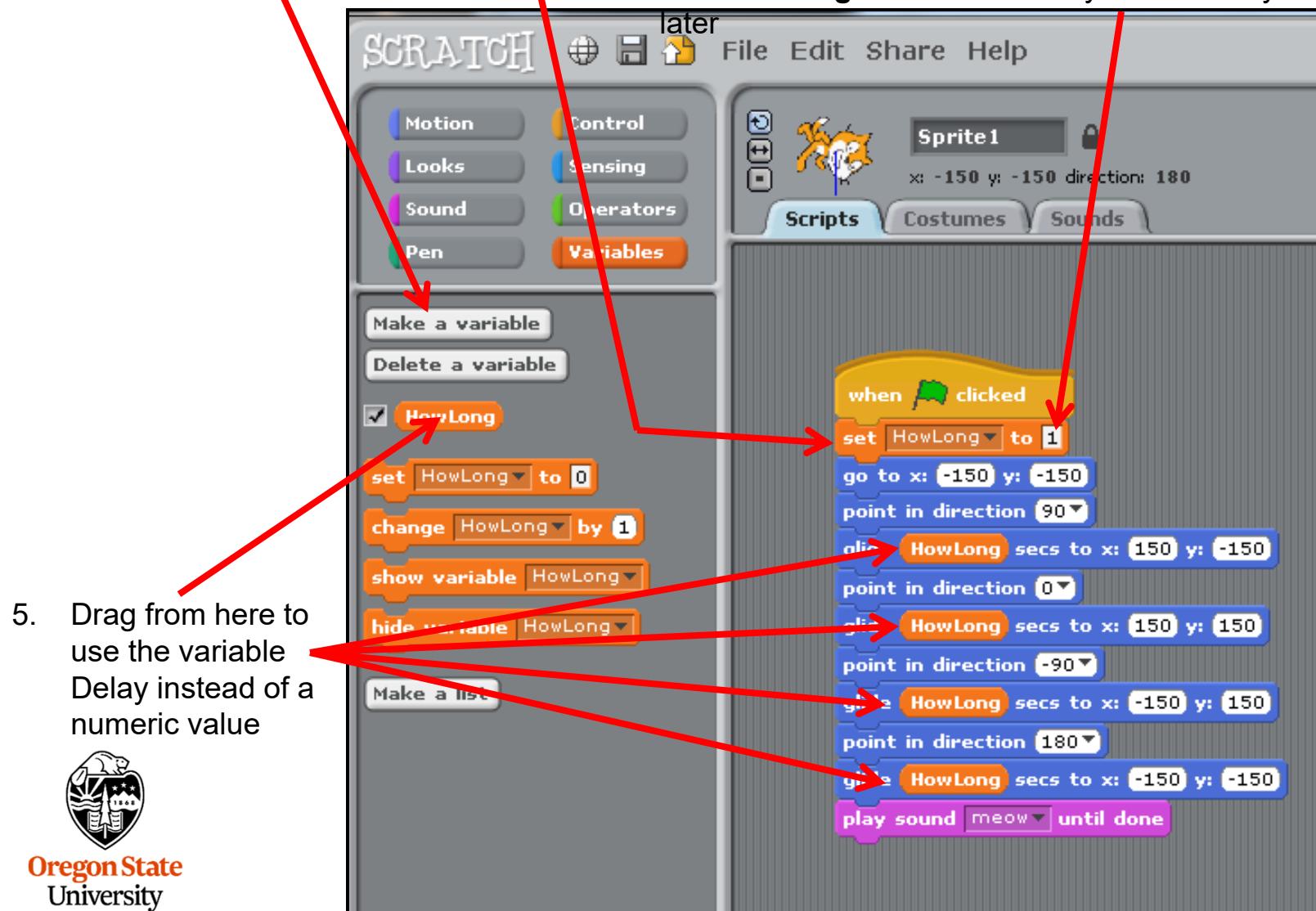
Oregon State
University
Computer
Sciences

What if we want to change how long it takes the cat to get from one place to another?

Changing the Glide Time in Four Places is a Pain, So Do This

48

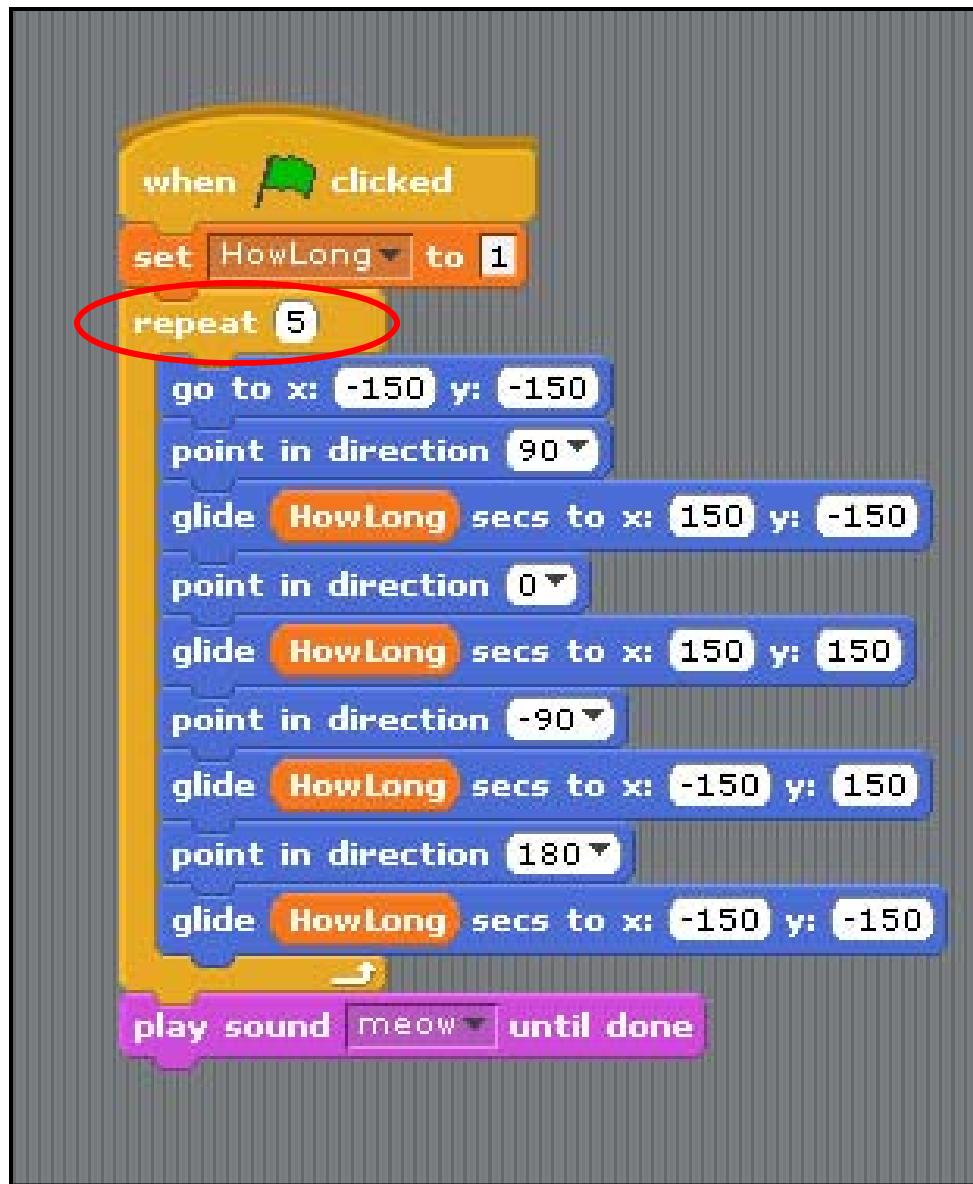
1. Make a variable
2. Call it **HowLong**
3. Put this command in here
4. Set **HowLong** to some value – you can always try other values



Oregon State
University

Computer Graphics

And Let's Repeat It Multiple Times



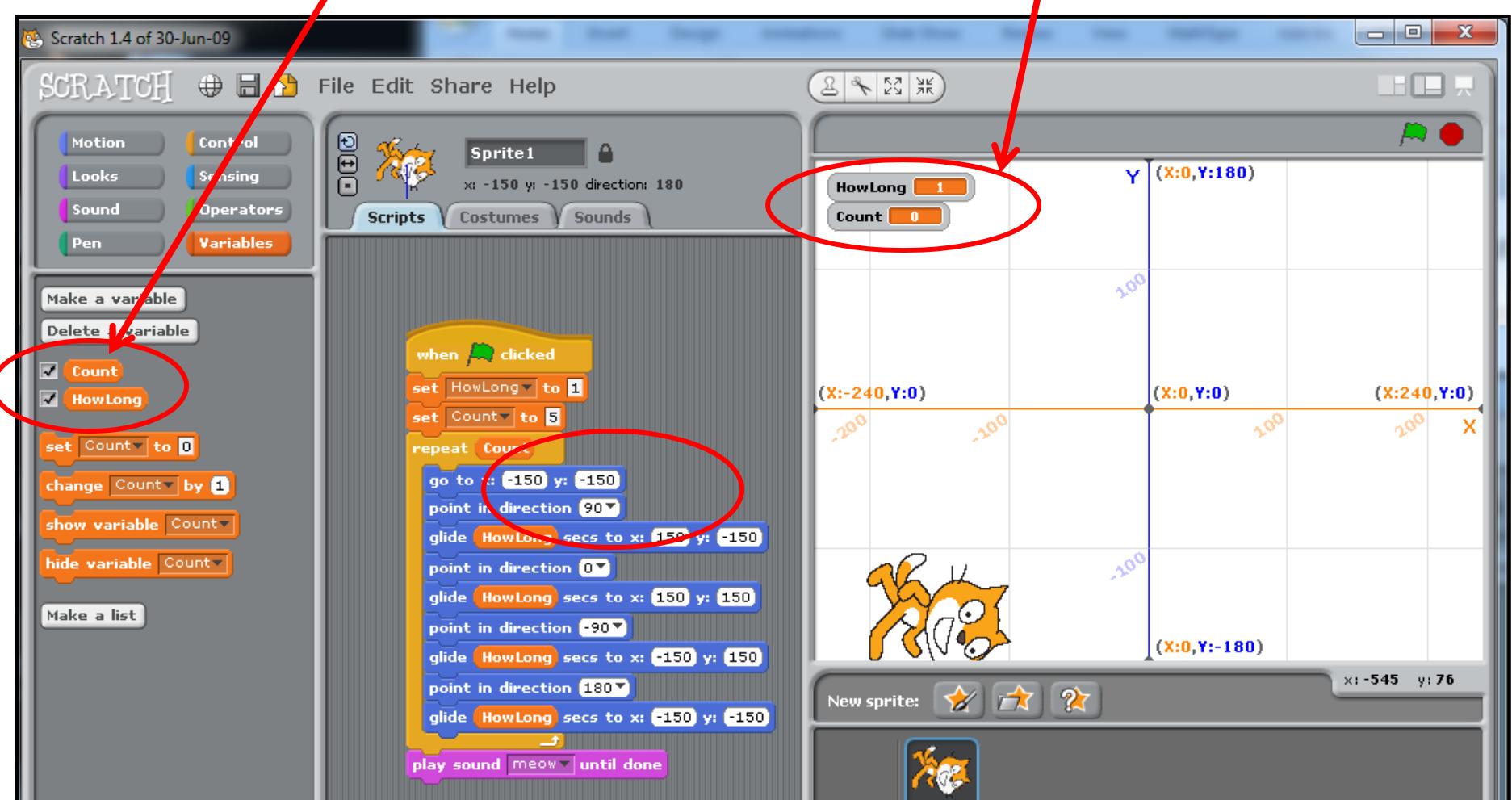
Oregon State
University

Computer Graphics

Putting the Repeat Count into a Variable

50

Clicking these checkboxes causes your variables to display on the Stage



Oregon State

University

Computer Graphics

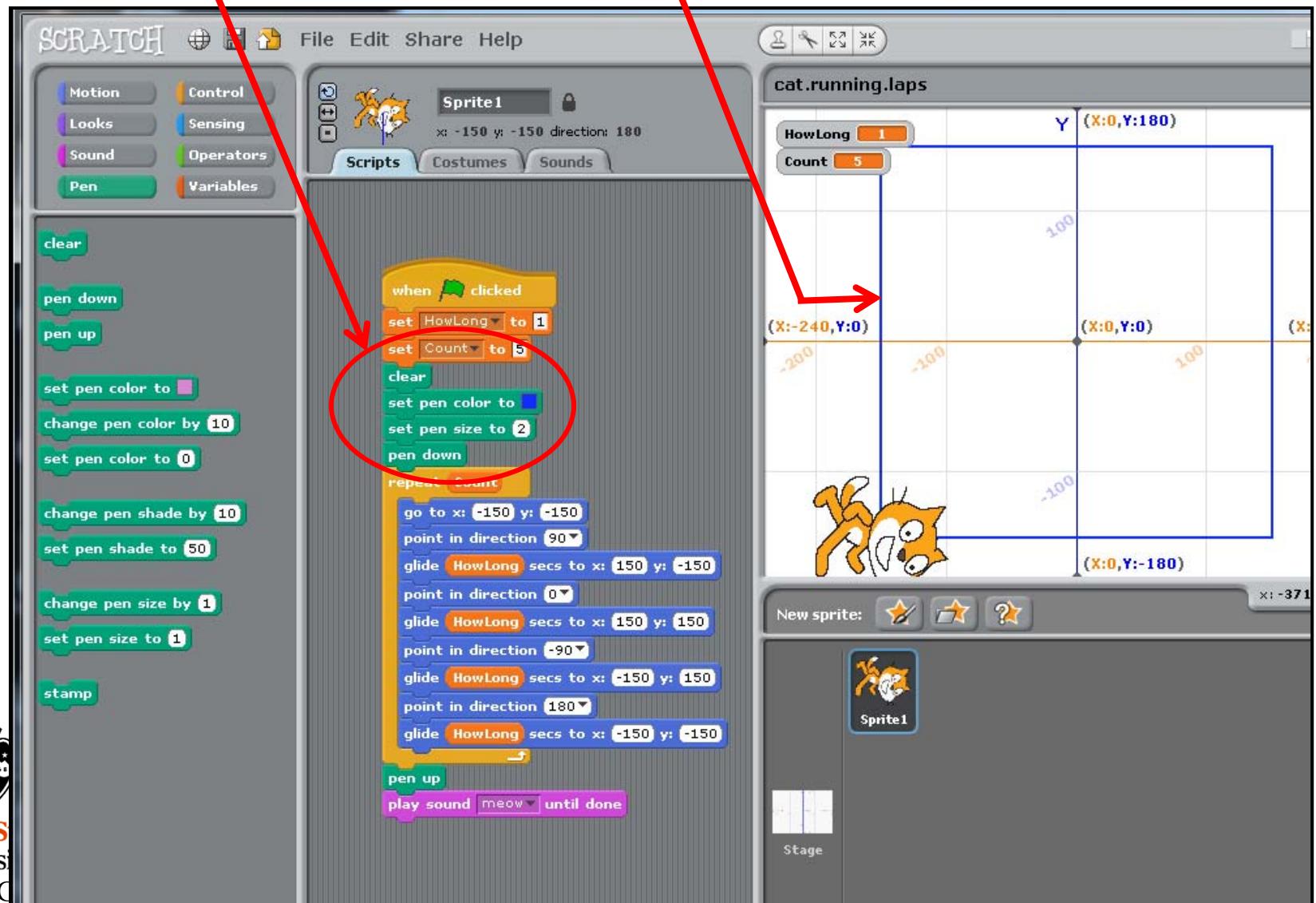
mjb – August 1, 2018

Let's Tie a Colored Pen to the Moving Cat

51

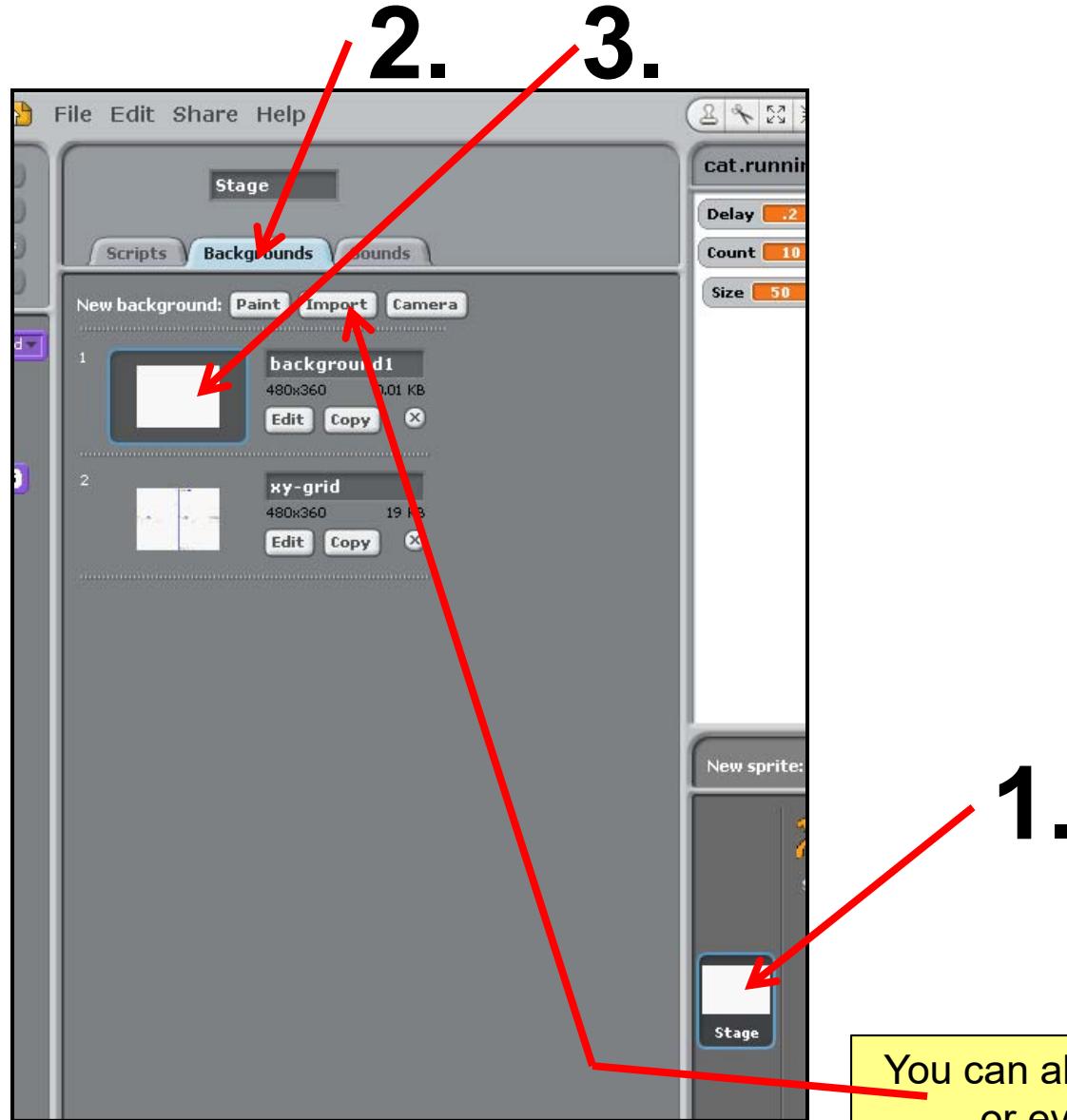
Clear the screen, Set the pen color, Set the pen width, Put the pen down (touching the page)

This draws this square



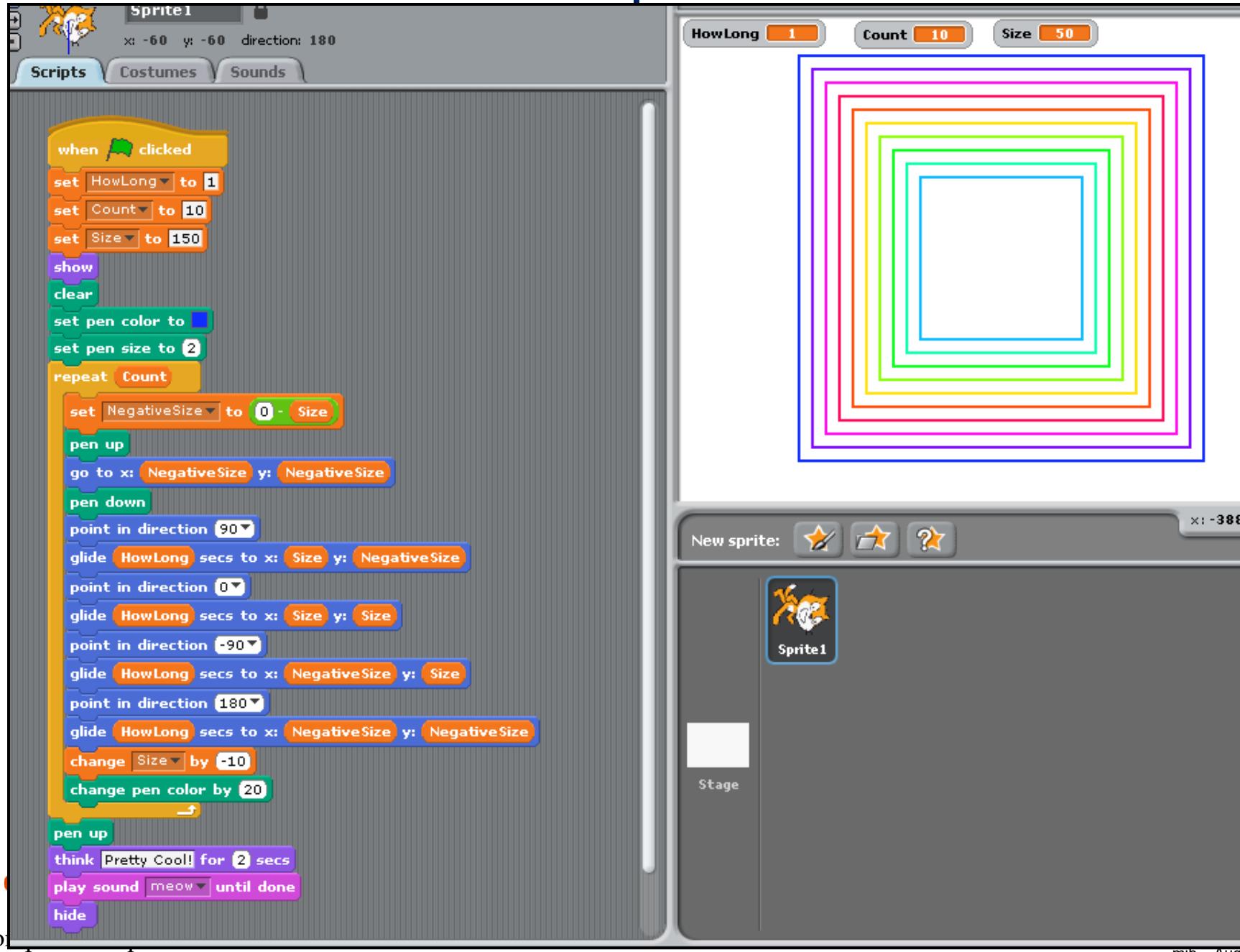
Time to Change the Background to Something Else

52



Even trickier – Change the Size and Color of Each Square

53

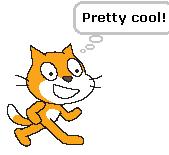


Co

mjb – August 1, 2018

Even trickier – Change the Size and Color of Each Square !

54



1. Define a variable called Size
2. Define another one called NegativeSize

3. Set Size to 150

4. Set NegativeSize to “0-Size” (i.e., -Size)

5. Lift the pen up, move to the start location, then put the pen down

6. Use Size and NegativeSize to give the coordinates to glide to

7. Change the size after each Repeat. Change the pen color too

8. After the Repeat is over, make the cat think something, meow, and then disappear

The Scratch script starts with a 'when green flag clicked' hat. It initializes variables: 'HowLong' to 1, 'Count' to 10, and 'Size' to 150. It then performs a series of actions: 'show', 'clear', 'set pen color to blue', 'set pen size to 2'. It enters a 'repeat Count' loop. Inside the loop, it sets 'NegativeSize' to 0 minus 'Size', lifts the pen, moves to the start location (x: NegativeSize, y: NegativeSize), puts the pen down, points in direction 90, and glides 'HowLong' seconds to those coordinates. It then changes direction by 90 degrees and repeats the gliding and direction-changing sequence for directions 0, -90, and 180. After the repeat loop, it changes 'Size' by -10, changes 'pen color' by 20, lifts the pen, thinks 'Pretty Cool!' for 2 seconds, plays a 'meow' sound until done, and finally hides.

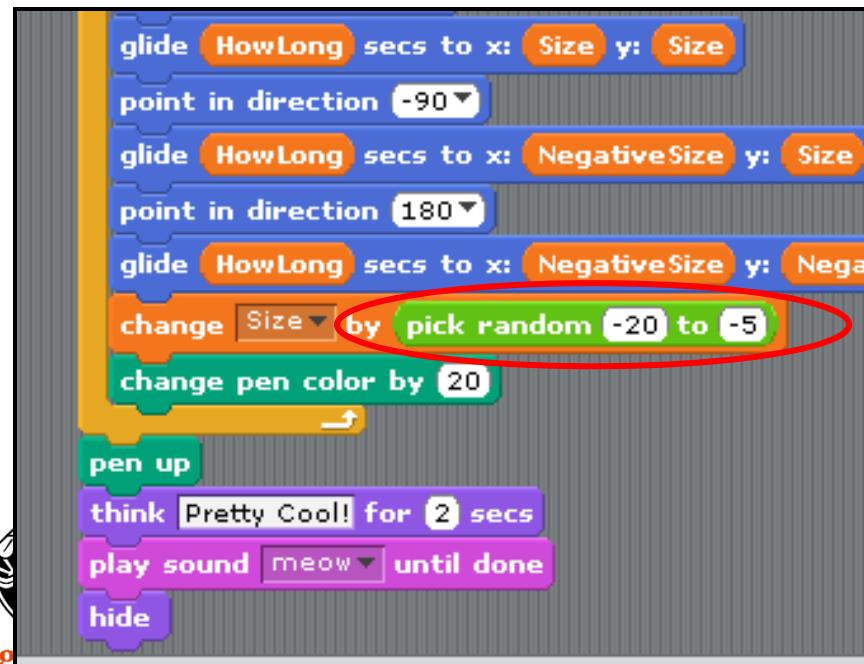
Adding Randomness to Your Animation

55

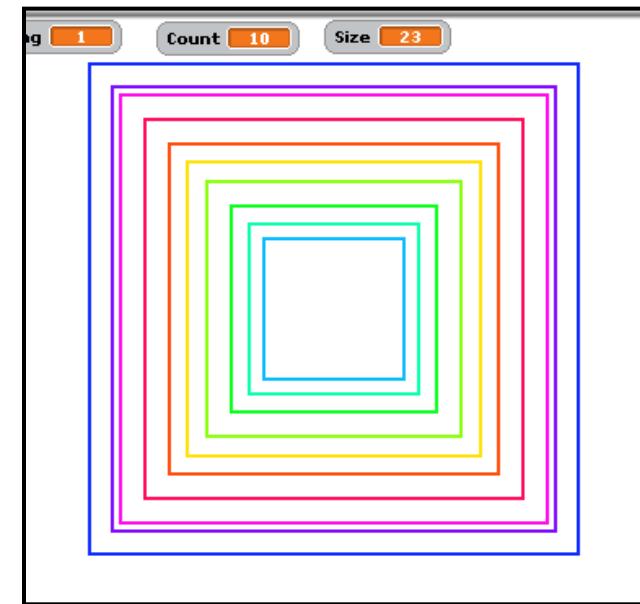
Sometimes these animations are more exciting when you can't predict exactly what will happen. This command will pick a random number between the two numbers that you give it:



You can drag it into place anywhere you would have otherwise used an actual number. For example, doing this ...

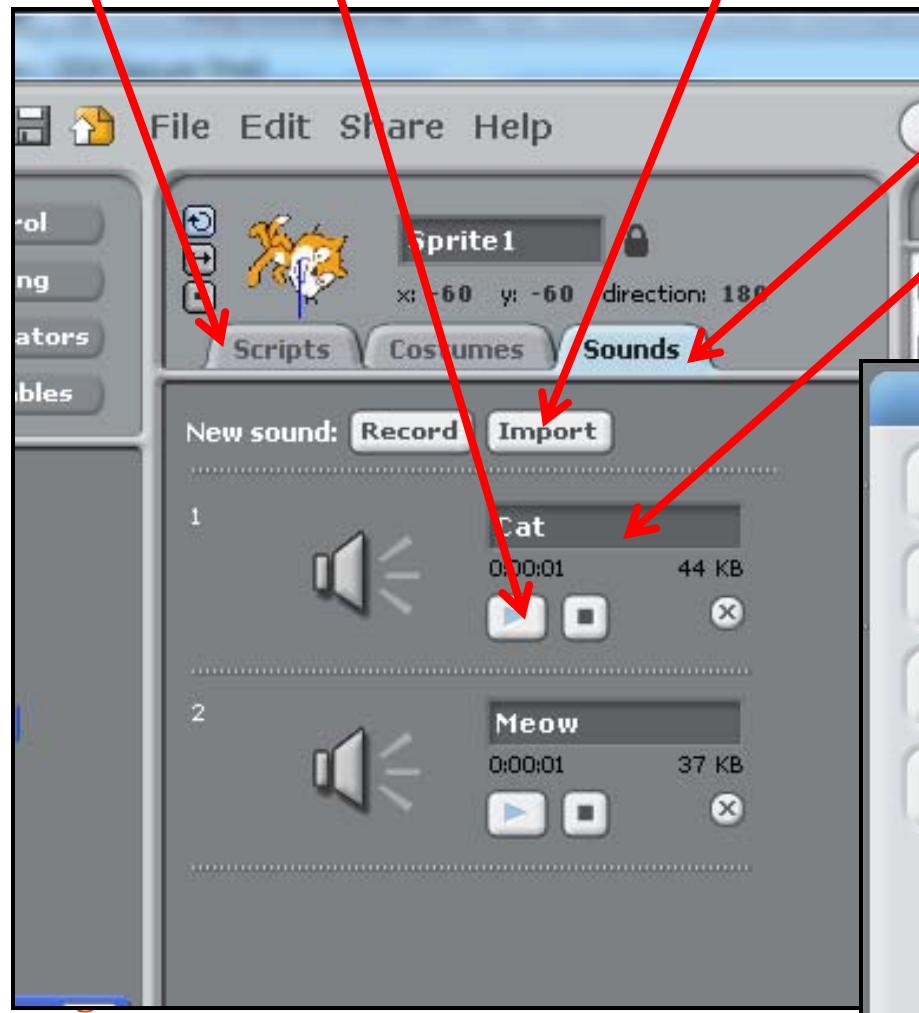


... could produce something like this:

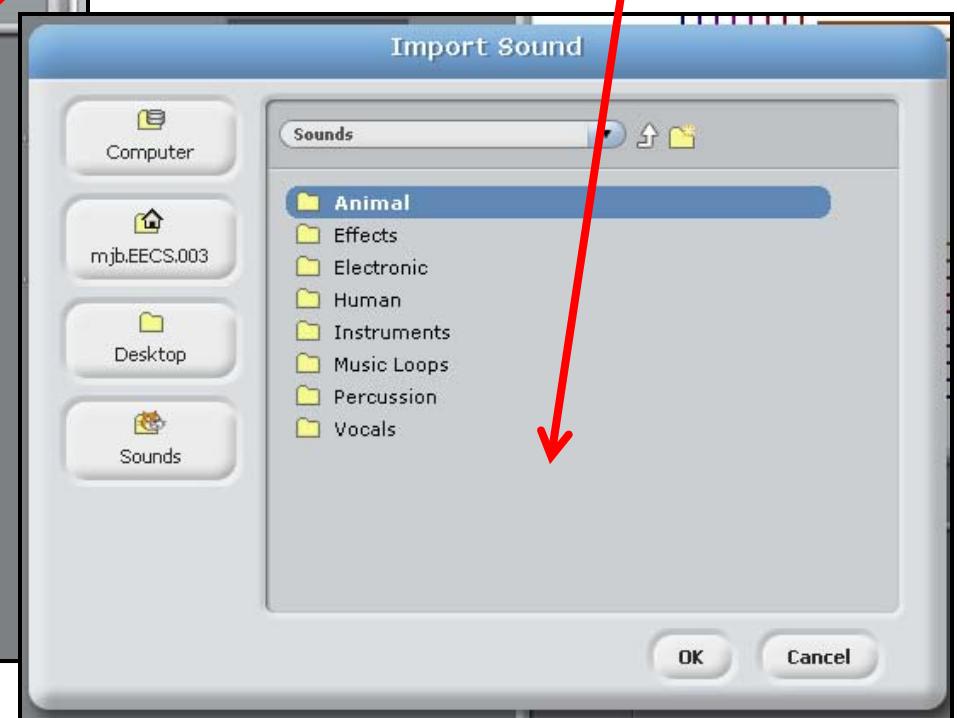


Making More Sounds Available for Your Animation!

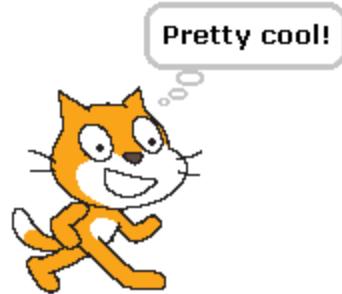
6. Click on the **Scripts** tab to use the new sound
2. Click on the **Import** button
4. See what it sounds like



1. Click on the **Sounds** tab
3. Select a sound
5. Give it a name you like



Other Things To Try



1. Have other sprites moving at the same time (hint: each sprite should have its own script)
2. Do something differently depending on the user pressing specific keyboard keys

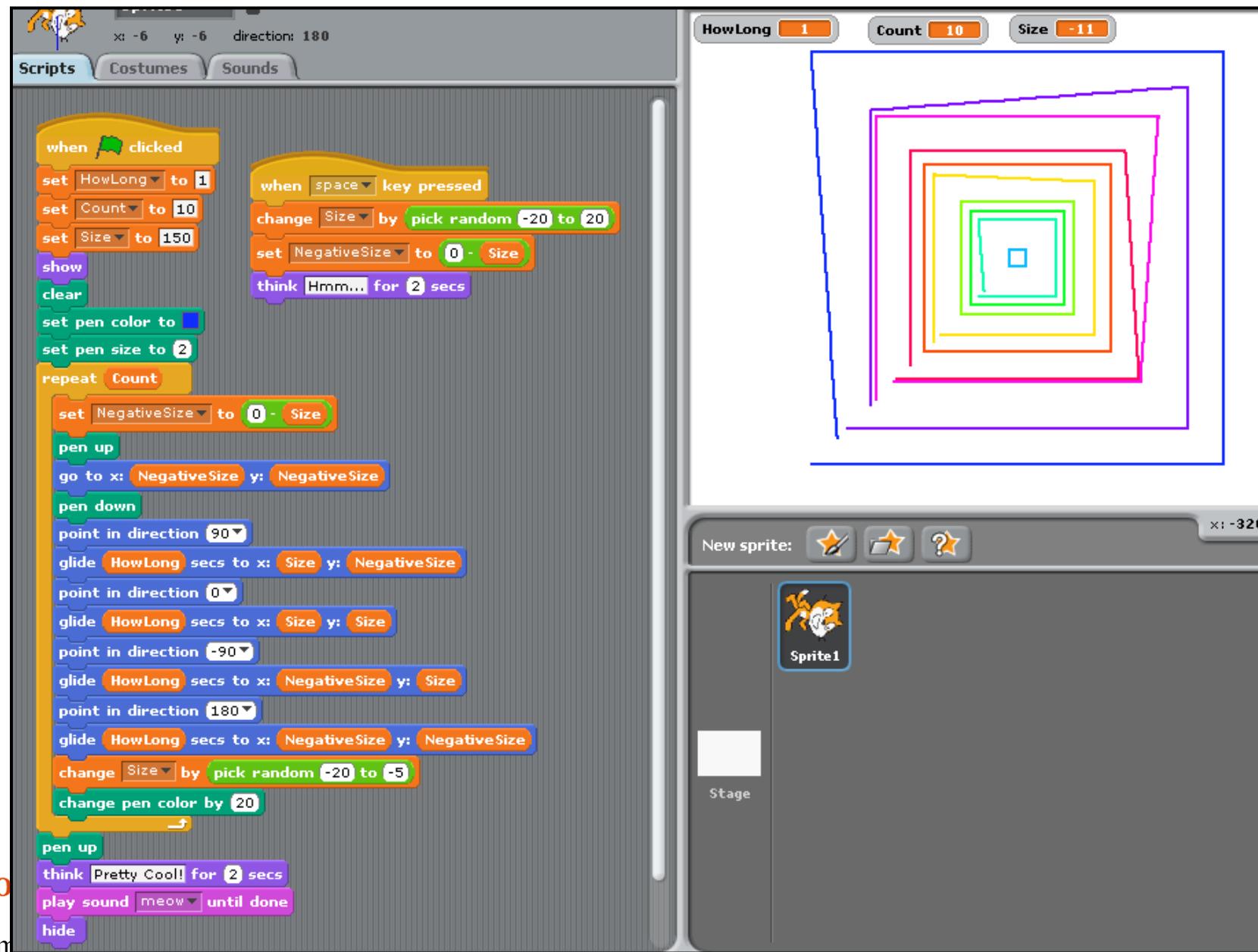


Oregon State
University

Computer Graphics

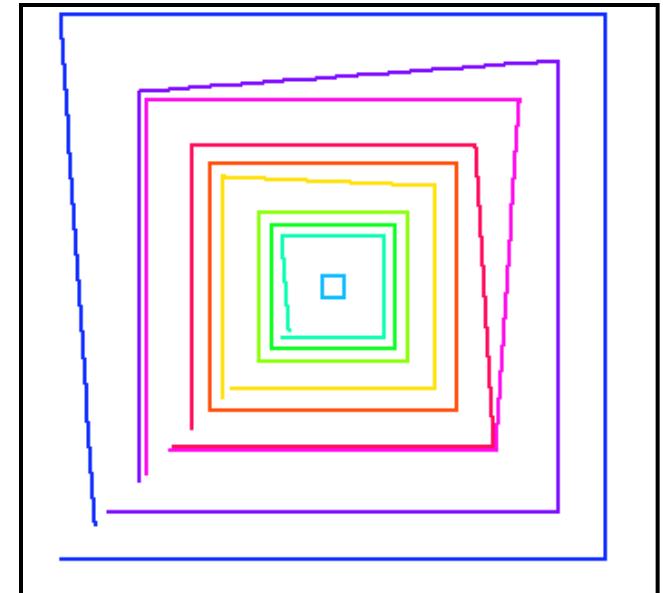
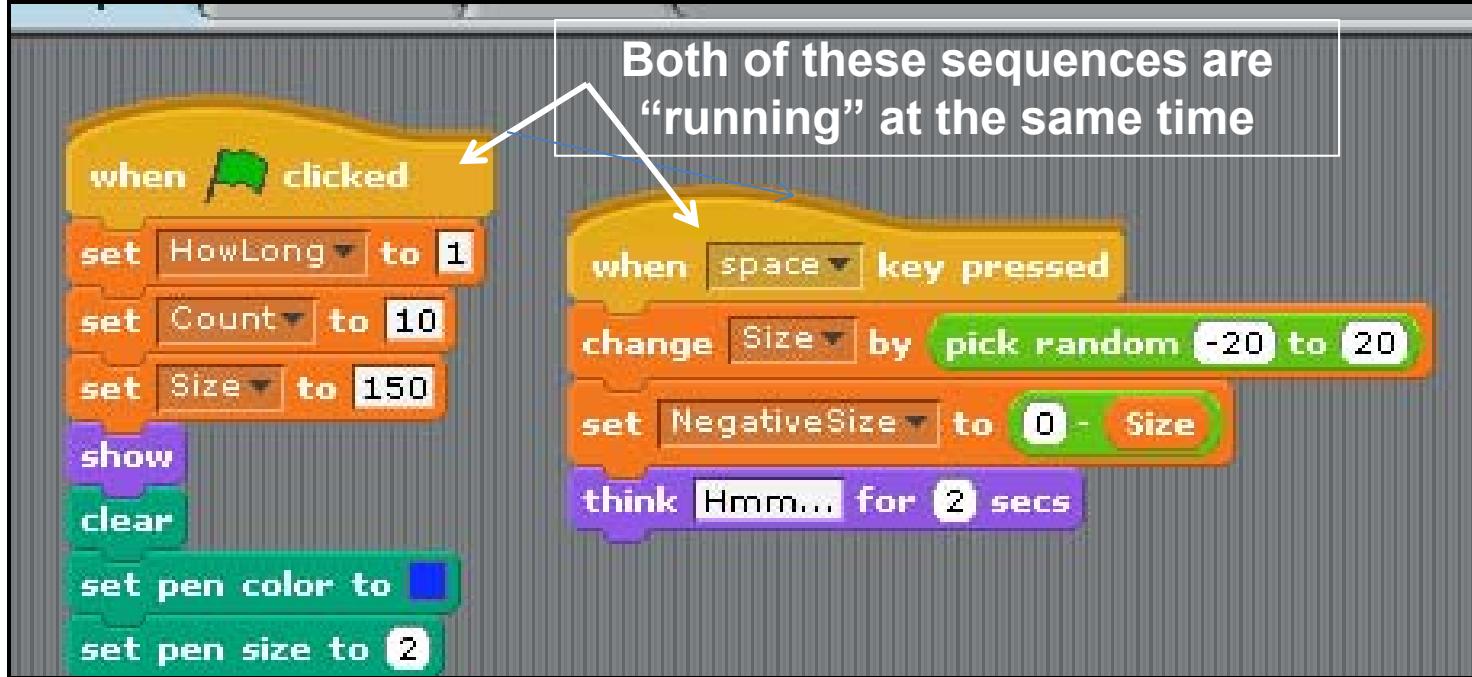
Changing the Animation When a Key is Pressed

58



Changing the Animation When a Key is Pressed

59



Using Your Own Files

60

Backgrounds: bmp, gif, jpg, png

Aspect ratio needs to be 4:3

480 x 360?

640 x 480?

Costumes: bmp, gif, jpg, png

The cat is 95 x 111

Looks like most any resolution will work

Sounds: wav



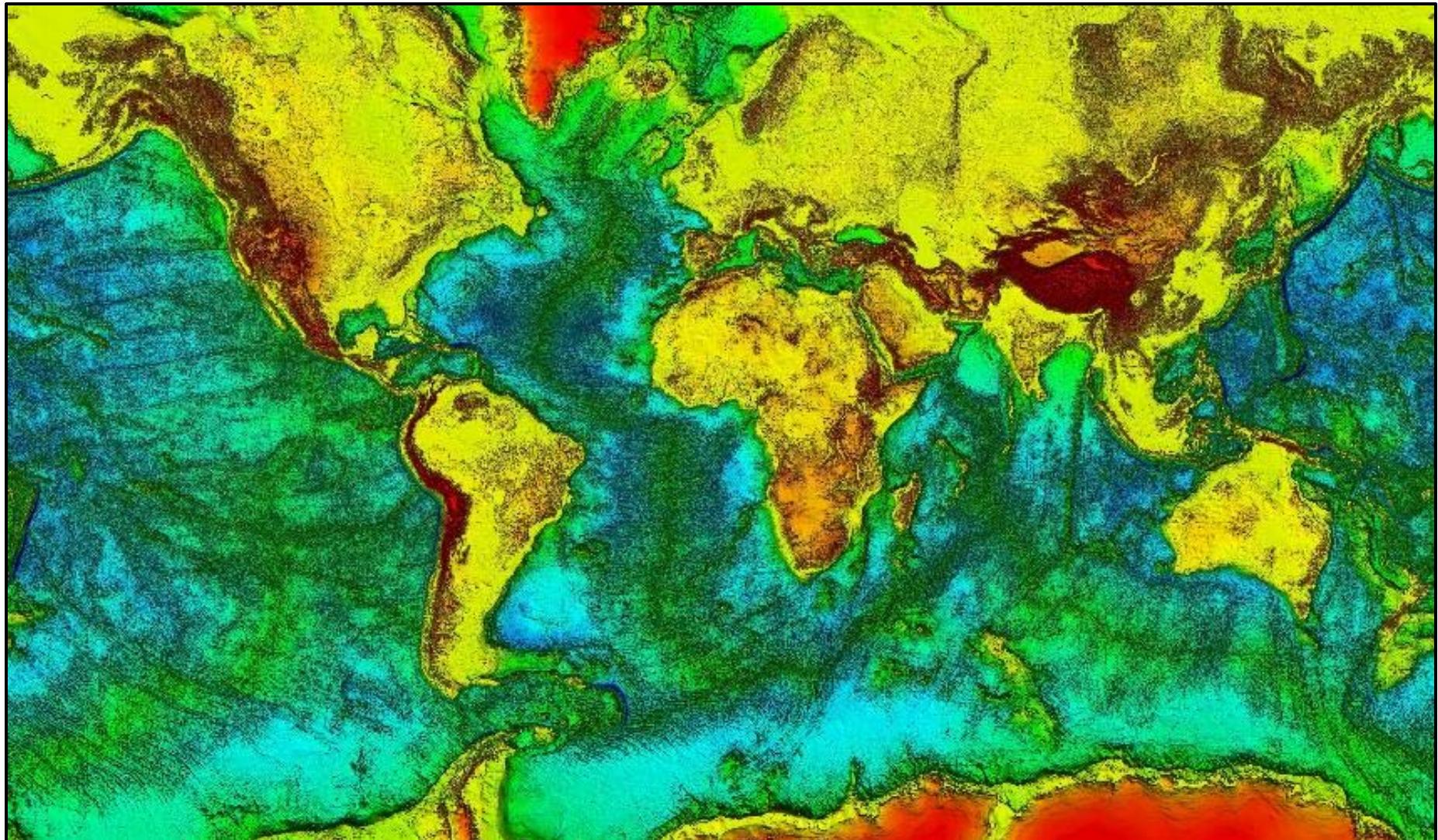
Oregon State
University

Computer Graphics

mjb – August 1, 2018

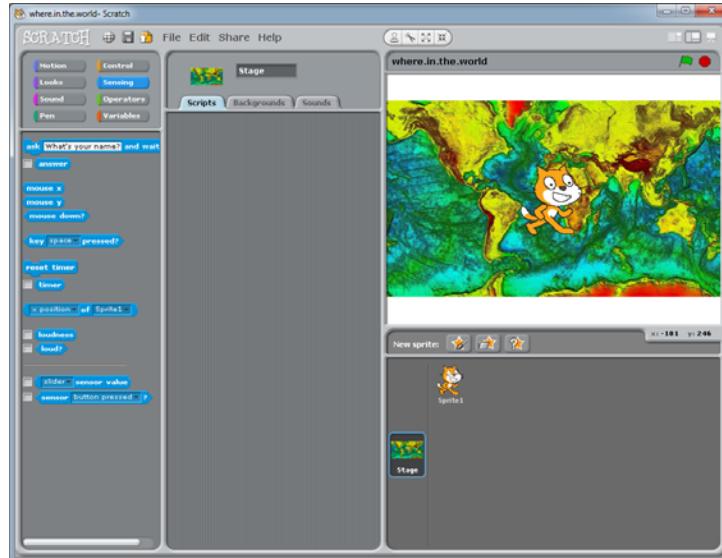
ChromaDepth: Encoding Depth into Image Colors

61

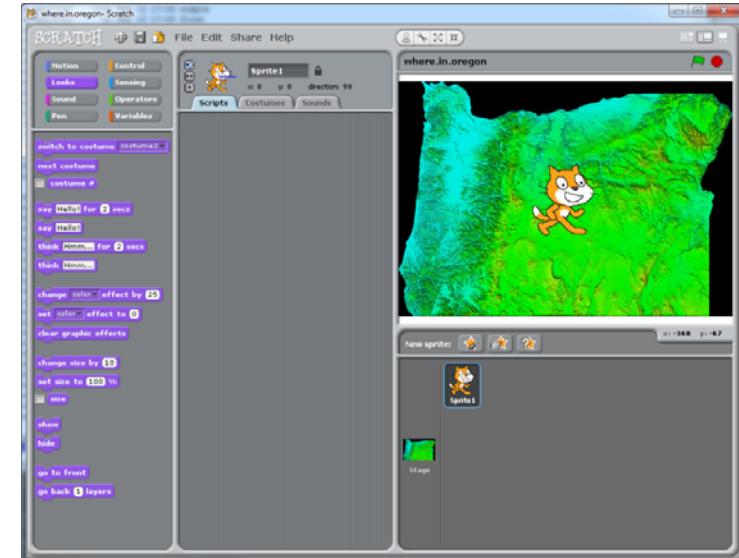


Custom ChromaDepth Map Backgrounds for Scratch

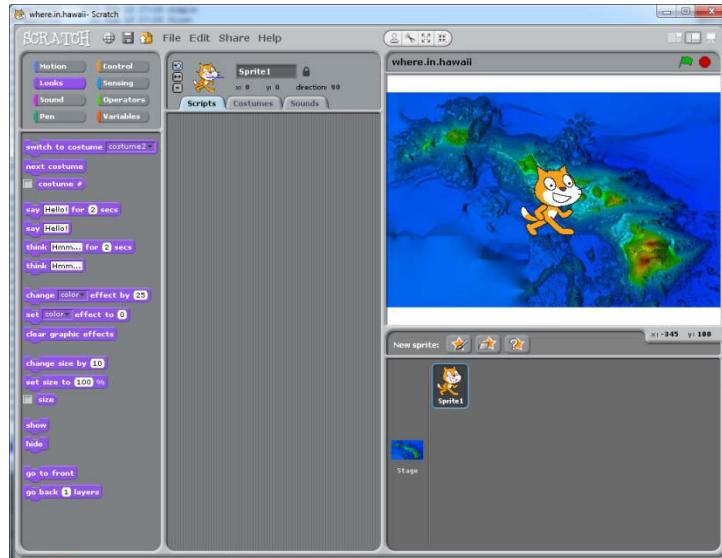
62



Background file = worldmap.jpg



Background file = oregonmap.jpg

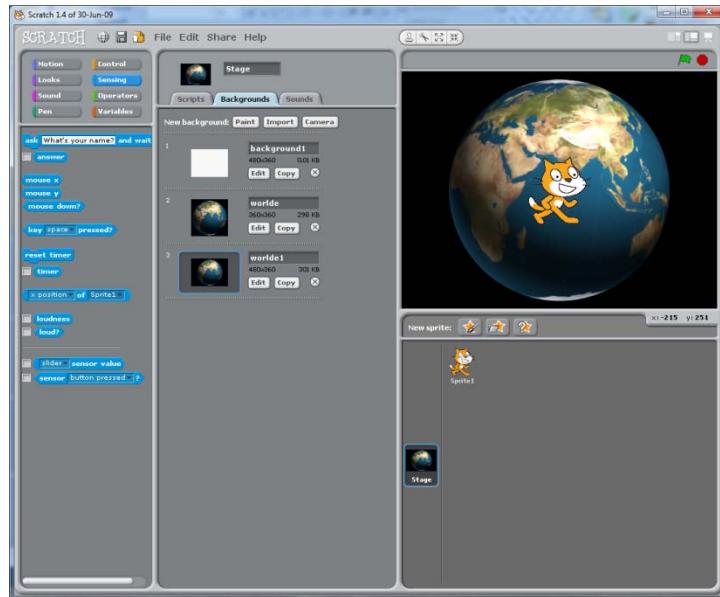


Background file = hawaiimap.jpg
Computer Graphics

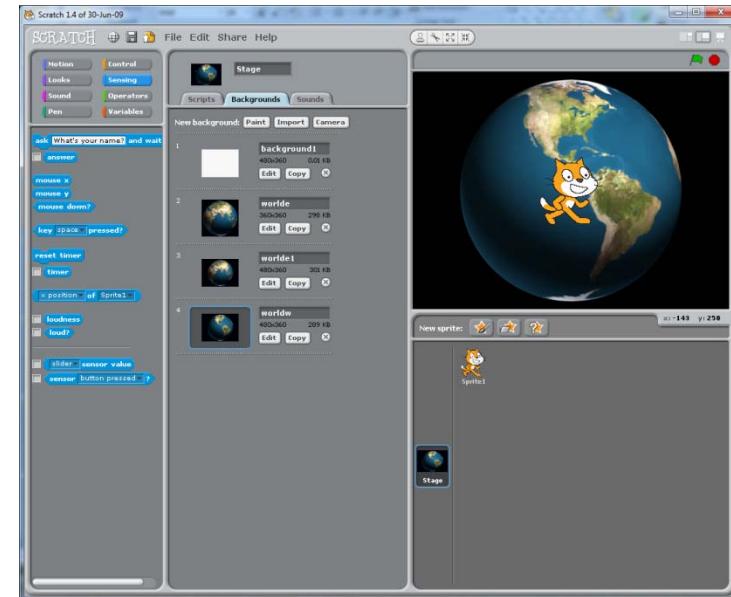
You can find these background map files at:
<http://cs.oregonstate.edu/~mjb/scratch>

You can find more ChromaDepth stuff at:
<http://cs.oregonstate.edu/~mjb/chromadepth>

Other Custom Map Backgrounds for Scratch



Background file = earthe.jpg



Background file = earthw.jpg



Background file = mars.jpg

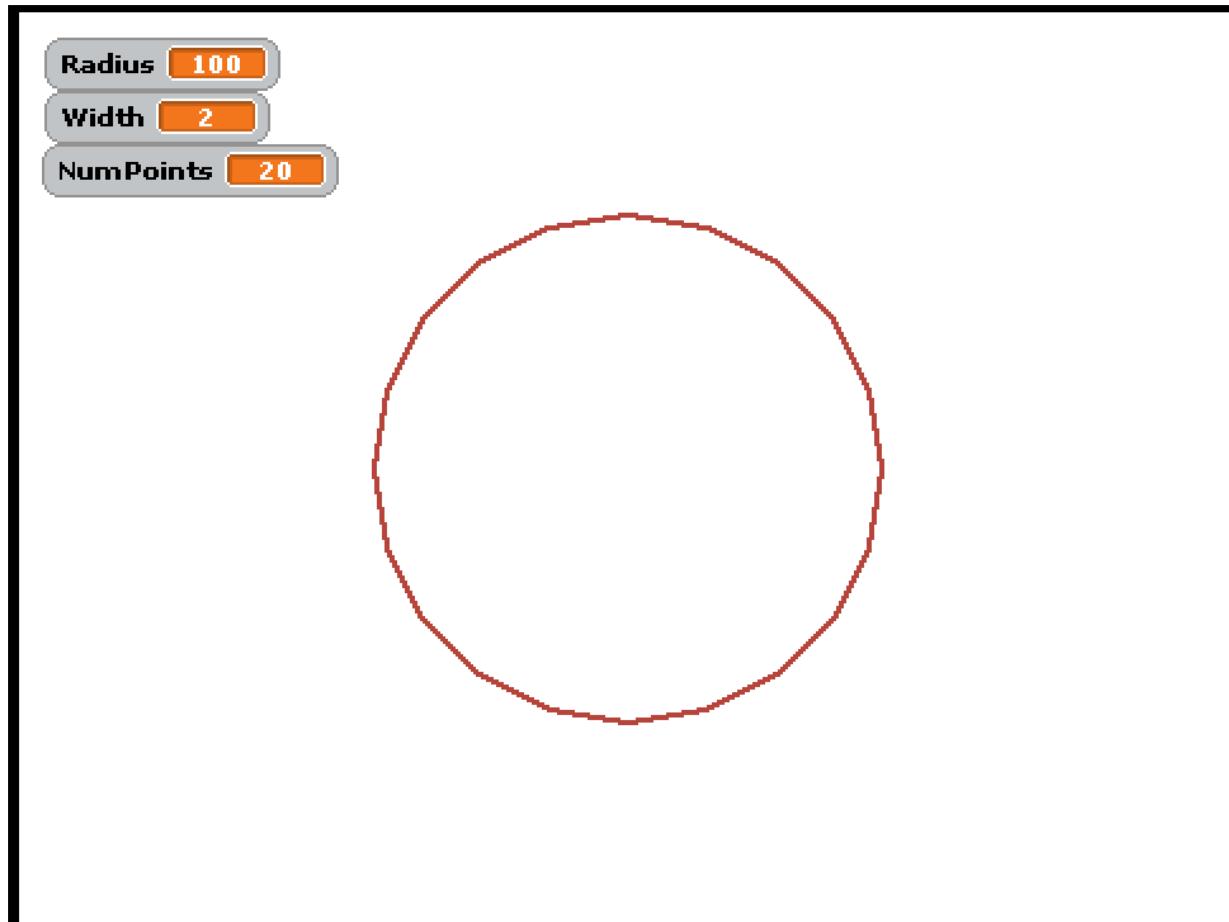
You can find these background map files at:
<http://cs.oregonstate.edu/~mjb/scratch>

Exercises



Drawing a Circle

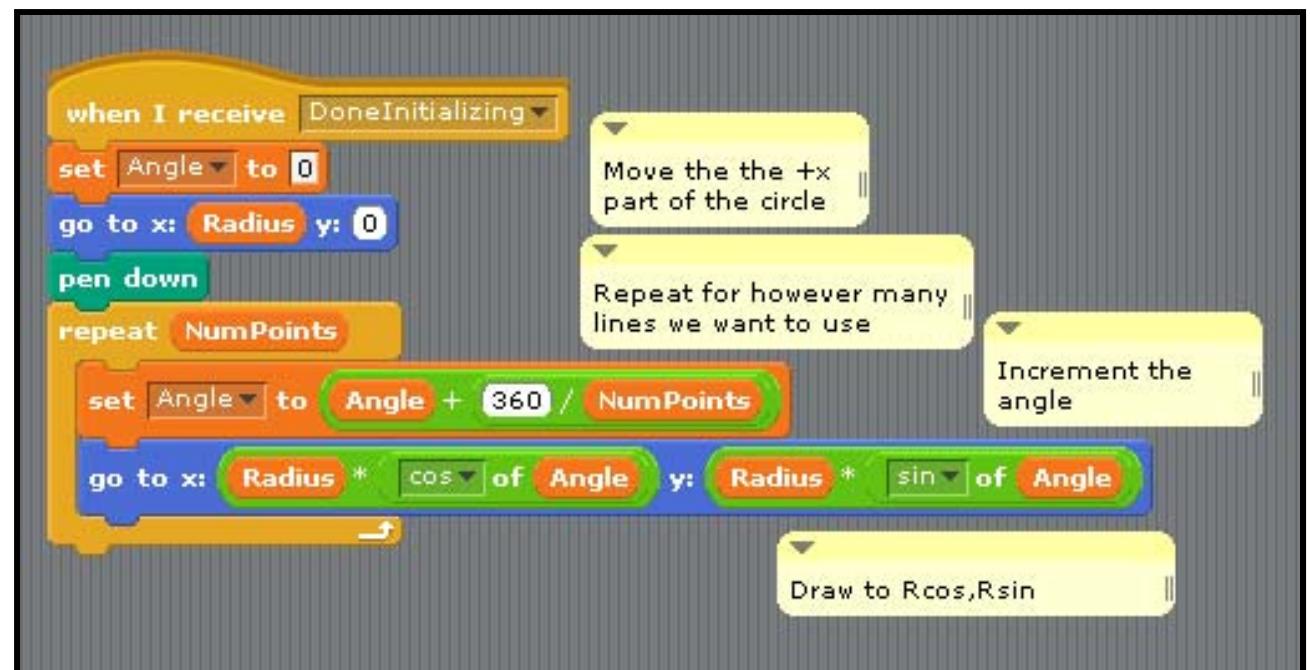
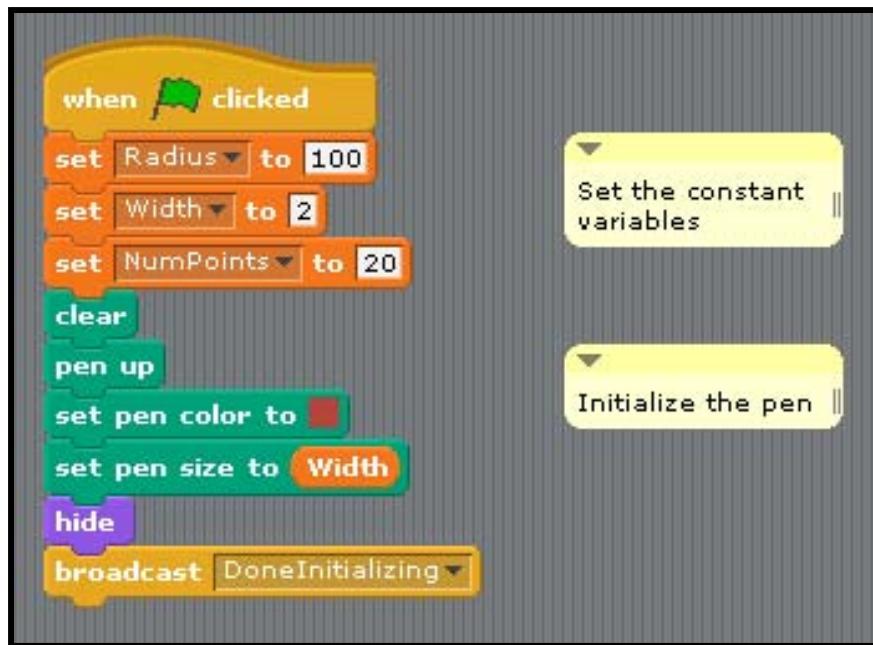
There are a lot of fun things you can do with Scratch that are of the form “increment-some-quantity-to-create-a-cool-drawing”. In this case, the “some-quantity” is the angle.



Oregon State
University

Computer Graphics

Drawing a Circle



Oregon State
University

Computer Graphics

Can We Imitate a SpiroGraph®? Sure!

67

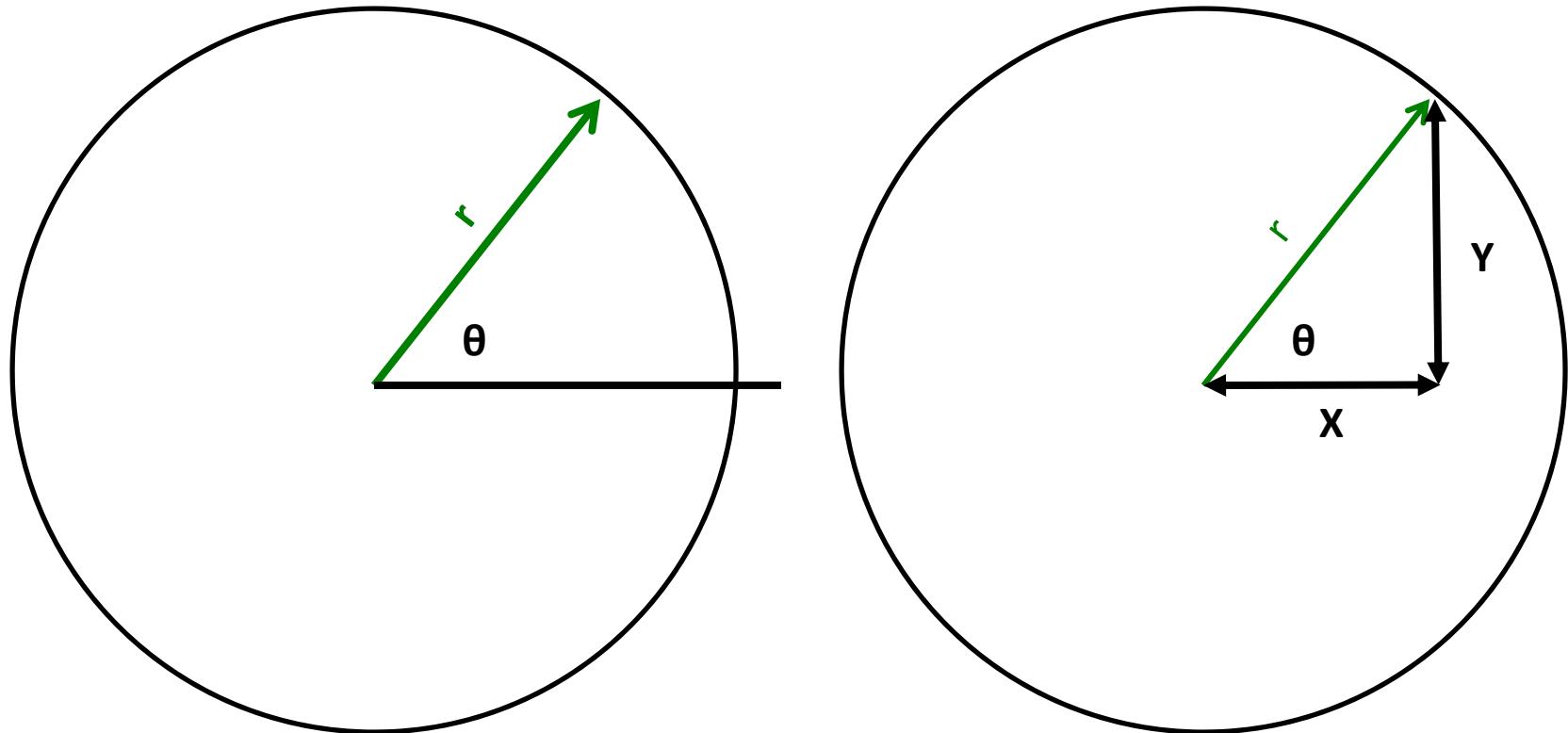


Oregon State
University
Computer Graphics

mjb – August 1, 2018

First, We Need to Know About Polar Coordinates

68



$$r = \sqrt{x^2 + y^2}$$

$$x = r \cos \theta$$

$$\theta = \tan^{-1} \left(\frac{y}{x} \right)$$

$$y = r \sin \theta$$



Oregon State
University

Computer Graphics

mjb – August 1, 2018

$$r = \sin(N\theta)$$

$$x = r \cos \theta$$

$$y = r \sin \theta$$

Start with $\theta=0$. and keep incrementing it by a small amount

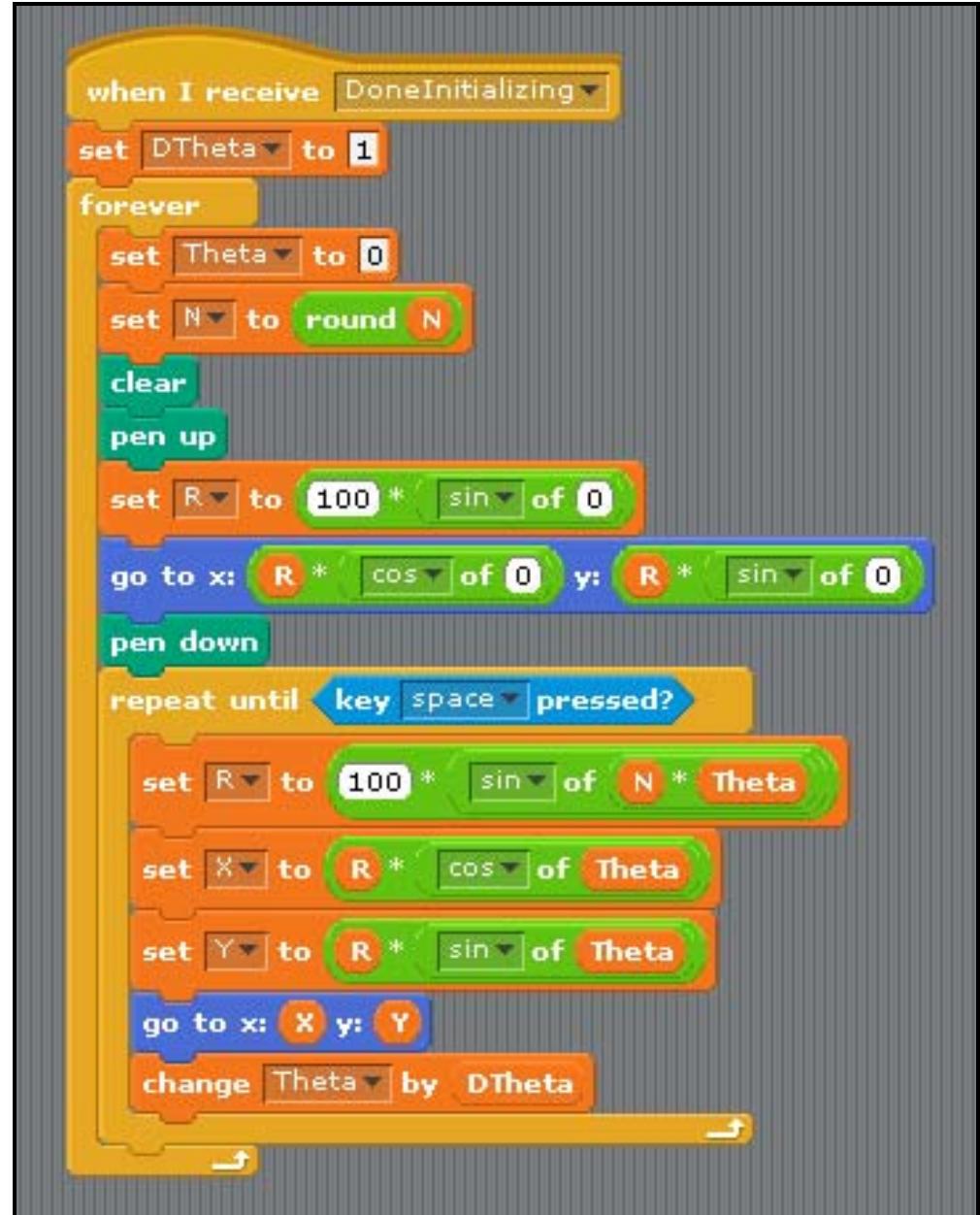


Oregon State
University

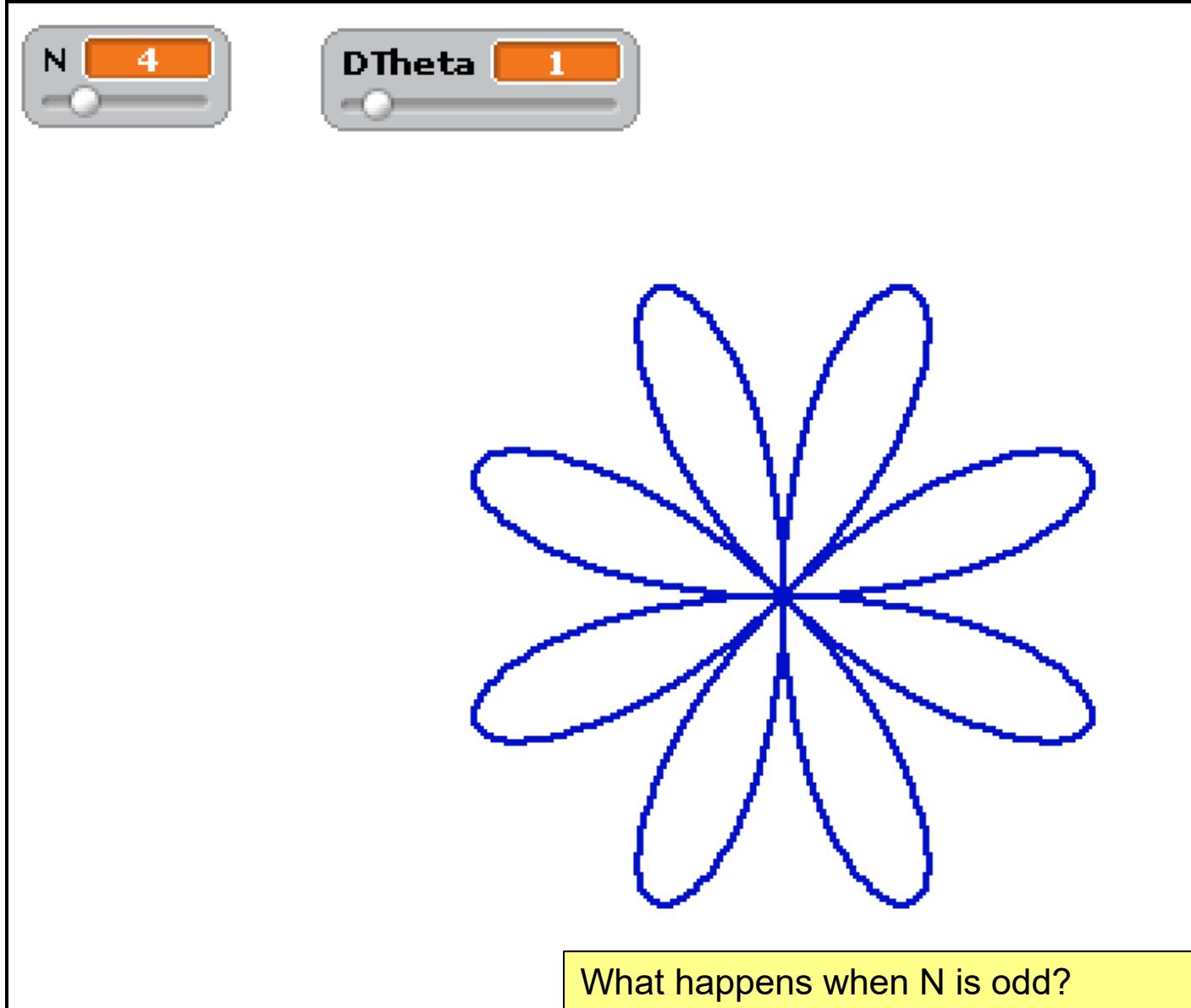
Computer Graphics

“Polar Rose”

70



“Polar Rose”



What happens when N is odd?
What happens when N is even?
What happens if N doesn't have to be a whole number?

“Polar Tulip”

72

$$r = \sin \theta + \sin^3\left(\frac{5\theta}{2}\right)$$

$$x = r \cos \theta$$

$$y = r \sin \theta$$

Start with $\theta=0$. and keep incrementing it by a small amount

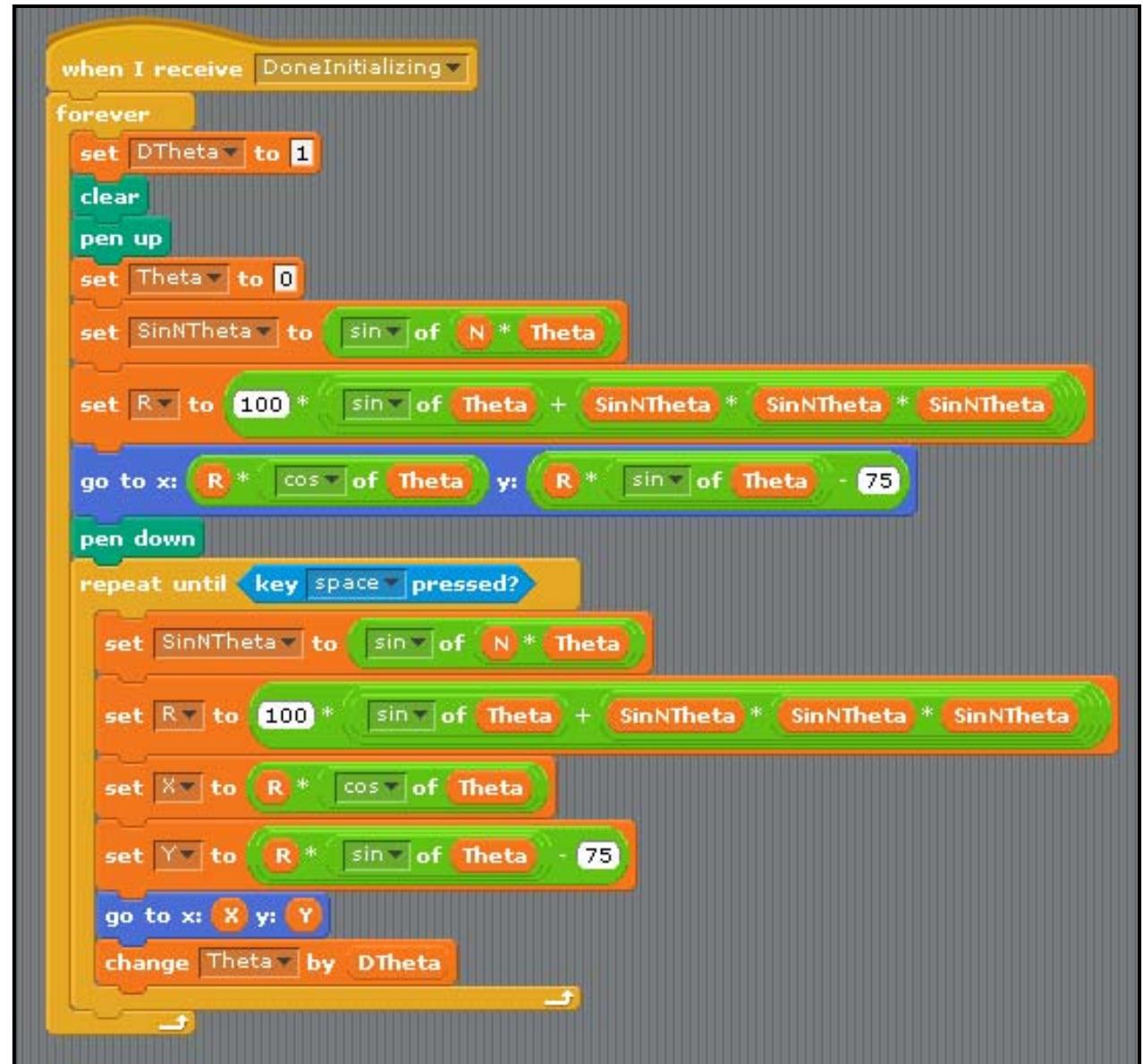


Oregon State
University

Computer Graphics

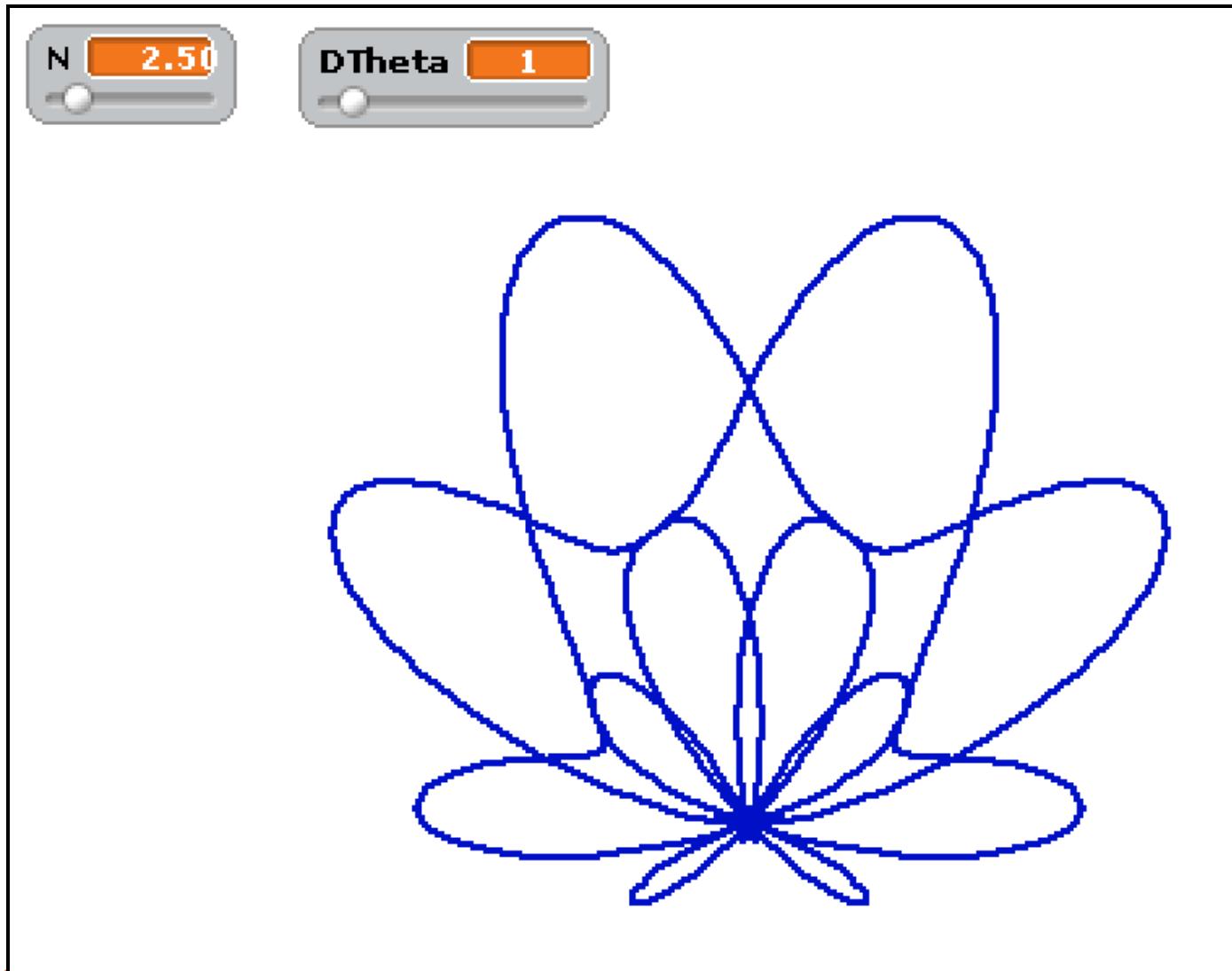
mjb – August 1, 2018

“Polar Tulip”



“Polar Tulip”

74



Limaçons (French for “snail”)

75

$$r = 1 + c \cdot \sin \theta$$

$$x = r \cos \theta$$

$$y = r \sin \theta$$

Start with $\theta=0$. and keep incrementing it by a small amount

From James Stewart, *Single Variable Calculus*, Brooks/Cole, 2001.

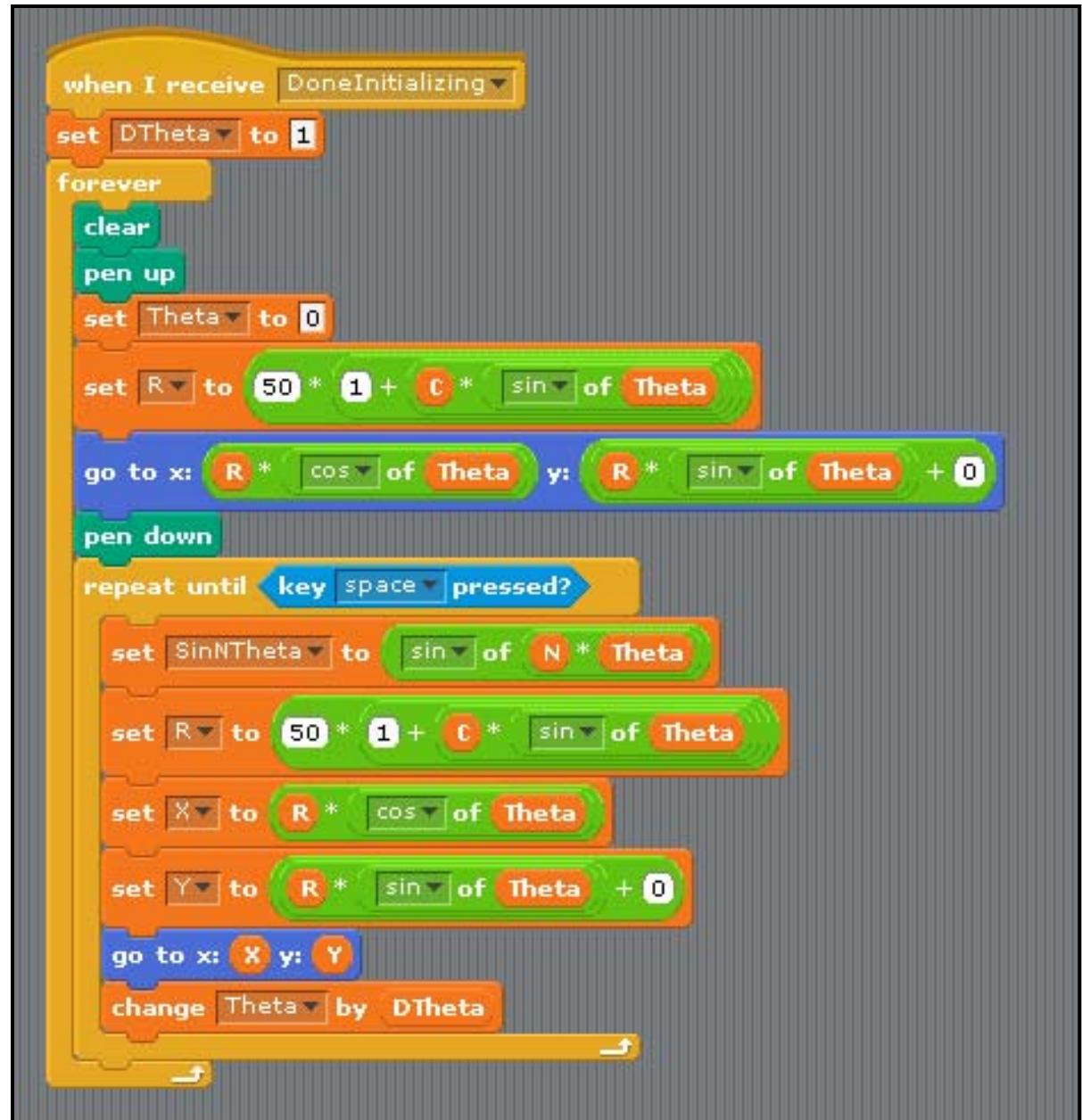
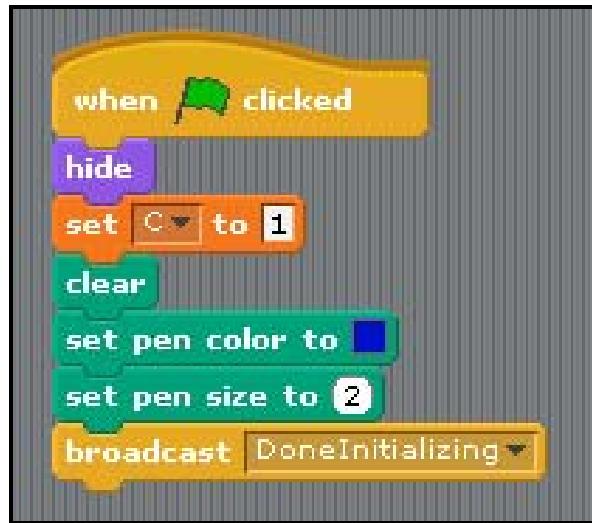


Oregon State
University

Computer Graphics

mjb – August 1, 2018

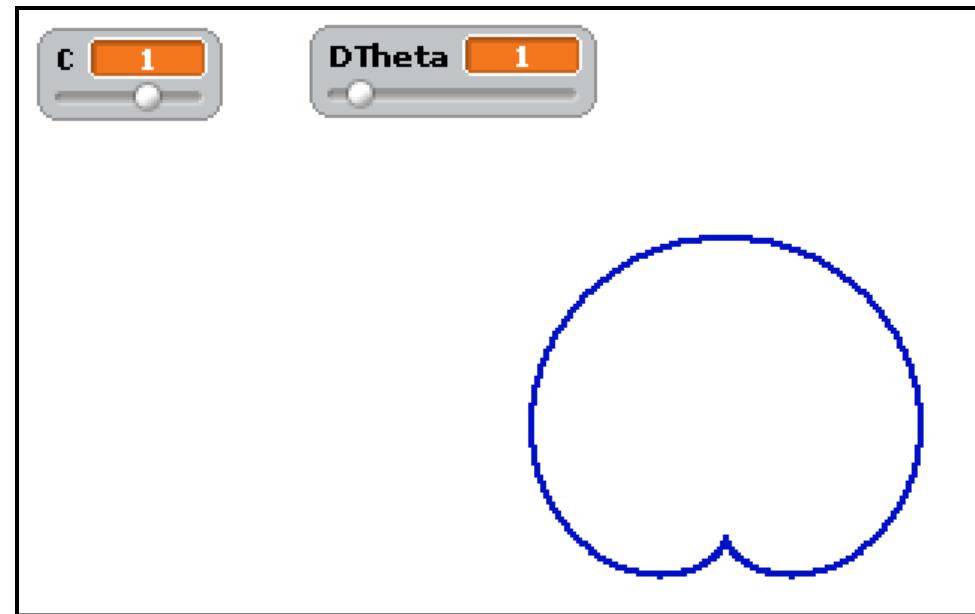
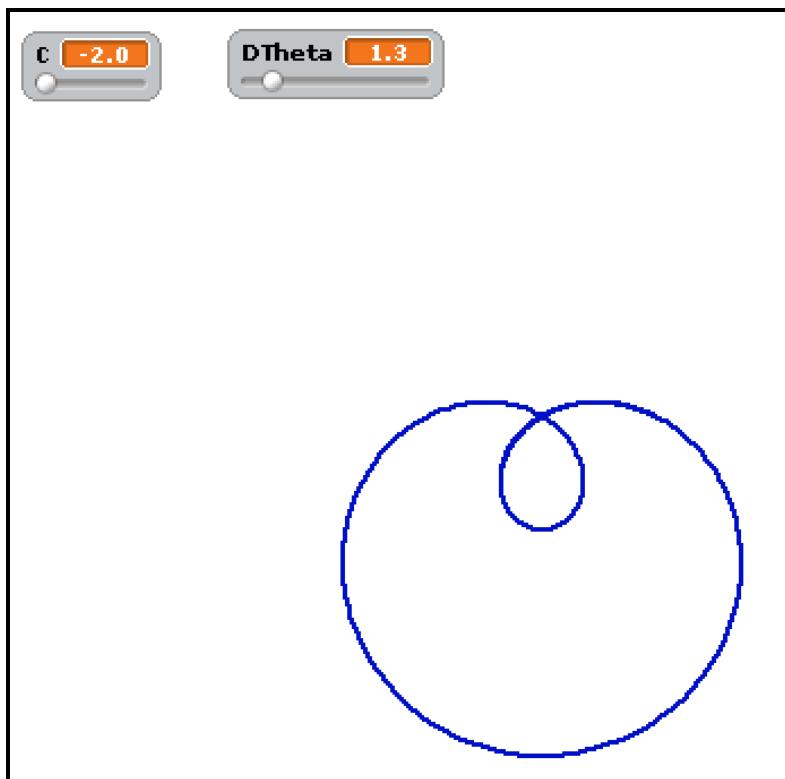
Limaçons



Oregon State
University

Computer Graphics

Limaçons



c = 1 is a “cardiod”



Oregon State
University
Computer Graphics

$$x = A \cos(ct + d)$$

$$y = B \sin(et + f)$$

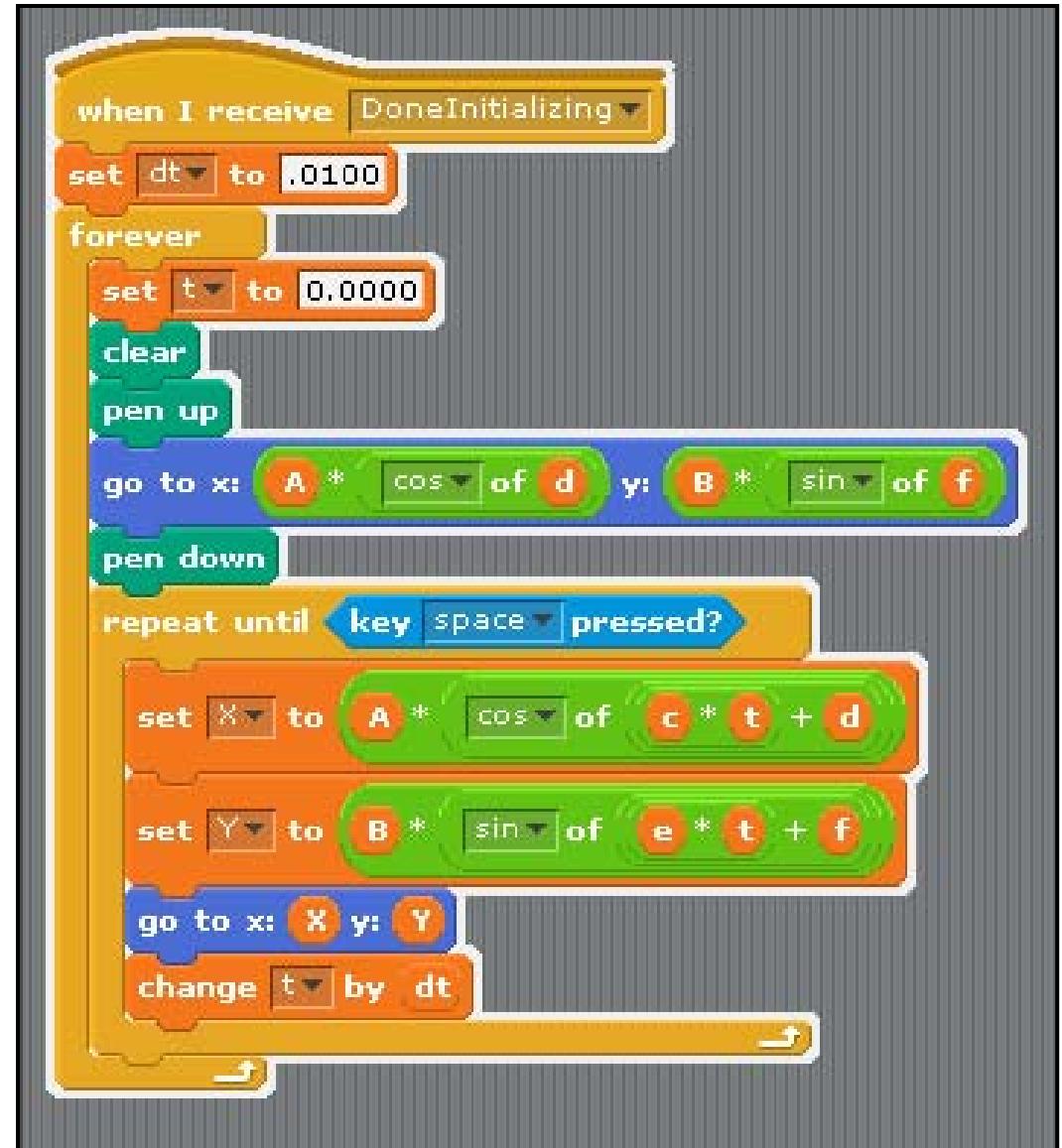
Start with t=0. and keep incrementing it by a small amount



Oregon State
University

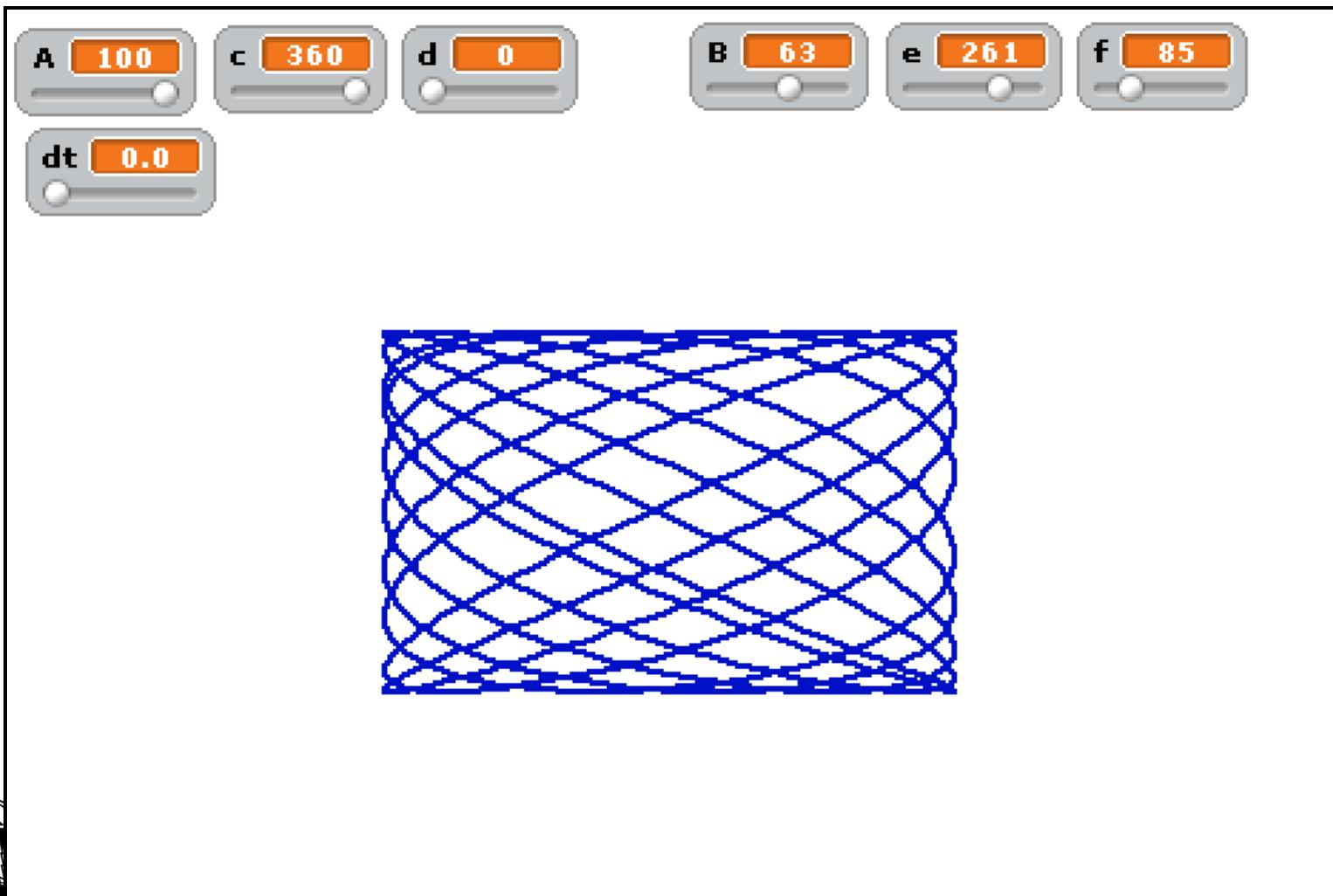
Computer Graphics

Lissajous Pattern



Lissajous Pattern

80



Here are some others that should work too,
but I haven't had the chance to try them yet

Epicycloid: $x = (R + r) \cos \theta - r \cos\left(\frac{R + r}{r} \theta\right)$

$$y = (R + r) \sin \theta - r \sin\left(\frac{R + r}{r} \theta\right)$$

<http://en.wikipedia.org/wiki/Epicycloid>

Epitrochoid: $x = (R + r) \cos \theta - d \cos\left(\frac{R + r}{r} \theta\right)$

$$y = (R + r) \sin \theta - d \sin\left(\frac{R + r}{r} \theta\right)$$

<http://en.wikipedia.org/wiki/Epitrochoid>



Oregon State
University

Computer Graphics

Here are some others that should work too,
but I haven't had the chance to try them yet

Hypotrochoid:

$$x = (R - r) \cos \theta + d \cos\left(\frac{R - r}{r} \theta\right)$$

$$y = (R - r) \sin \theta - d \sin\left(\frac{R - r}{r} \theta\right)$$

<http://en.wikipedia.org/wiki/Hypotrochoid>

Hypocycloid:

$$x = (R - r) \cos \theta + r \cos\left(\frac{R - r}{r} \theta\right)$$

$$y = (R - r) \sin \theta - r \sin\left(\frac{R - r}{r} \theta\right)$$

<http://en.wikipedia.org/wiki/Hypocycloid>

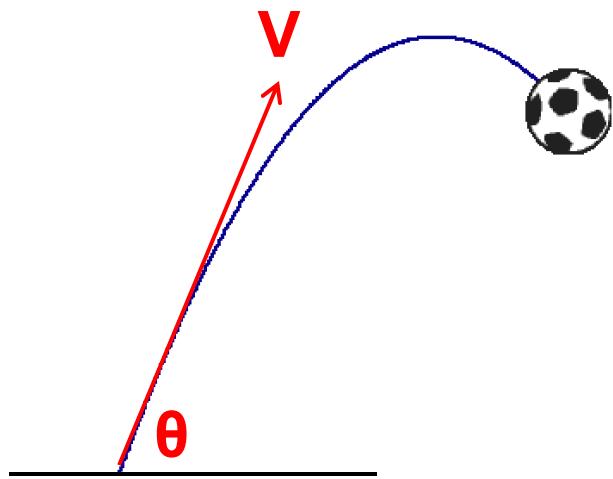


Oregon State
University

Computer Graphics

Projectile Motion

83



Use:

m for length units
seconds for time units

$$g = 9.8 \text{ m/sec}^2$$

$$v_{x0} = V \cos \theta$$

$$x = x_0 + v_{x0}t$$

$$v_{y0} = V \sin \theta$$

$$y = y_0 + v_{y0}t + \frac{1}{2}gt^2$$



Oregon State
University
Computer Graphics

mjb – August 1, 2018

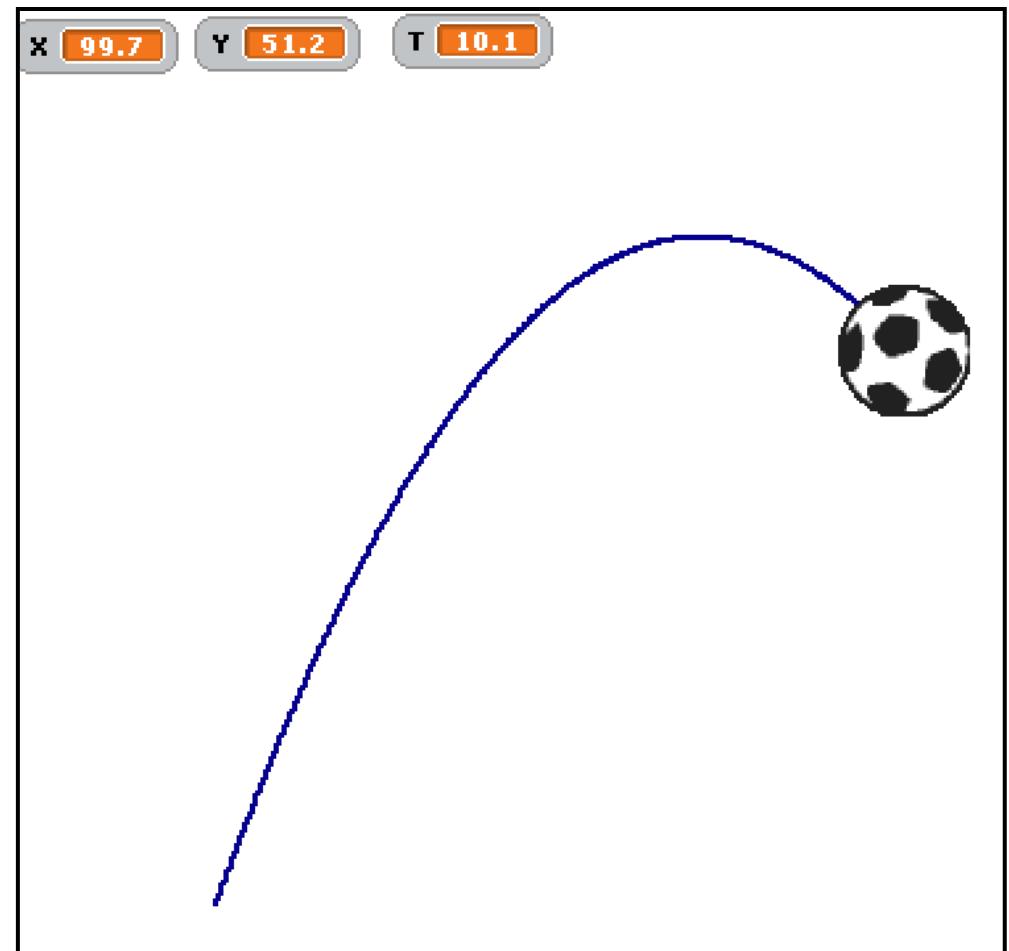
Projectile Motion

```

when green flag clicked
  set [G v] to [1 * -9.8]
  set [X0 v] to [-160]
  set [Y0 v] to [-160]
  set [Vel0 v] to [75]
  set [Angle0 v] to [70]
  set [Vx0 v] to [Vel0 * cos of Angle0]
  set [Vy0 v] to [Vel0 * sin of Angle0]
  clear
  set pen color to [blue]
  set pen size to [2]
  pen up
  go to x: X0 y: Y0
  broadcast [DoneInitializing v]

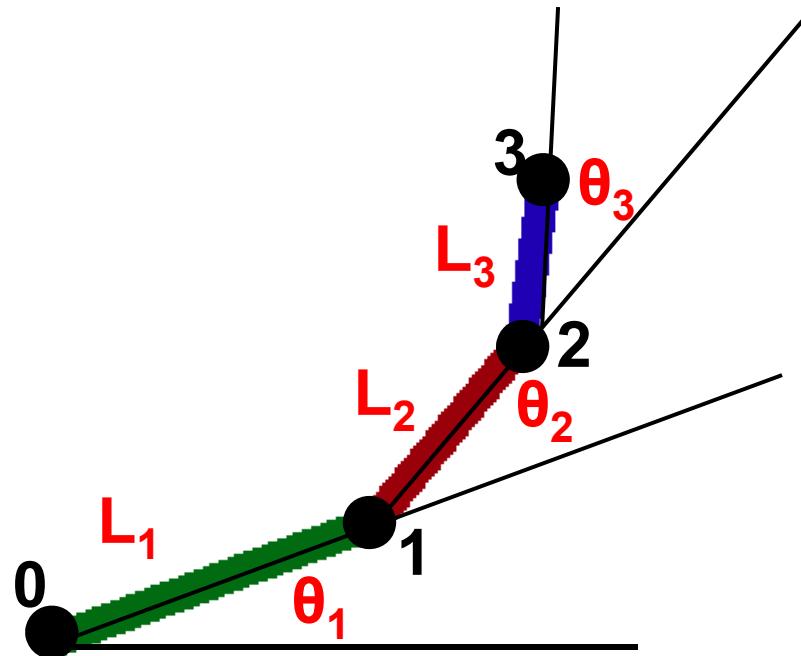
when I receive [DoneInitializing v]
  pen down
  reset timer
  repeat (600)
    set [T v] to [1 * timer]
    set [X v] to [X0 + Vx0 * T]
    set [Y v] to [Y0 + Vy0 * T + 0.5 * G * T * T]
    go to x: X y: Y

```



Forward Kinematics

85



$$x_1 = x_0 + L_1 \cos \theta_1$$

$$y_1 = y_0 + L_1 \sin \theta_1$$

$$x_2 = x_1 + L_2 \cos(\theta_1 + \theta_2)$$

$$y_2 = y_1 + L_2 \sin(\theta_1 + \theta_2)$$

$$x_3 = x_2 + L_3 \cos(\theta_1 + \theta_2 + \theta_3)$$

$$y_3 = y_2 + L_3 \sin(\theta_1 + \theta_2 + \theta_3)$$



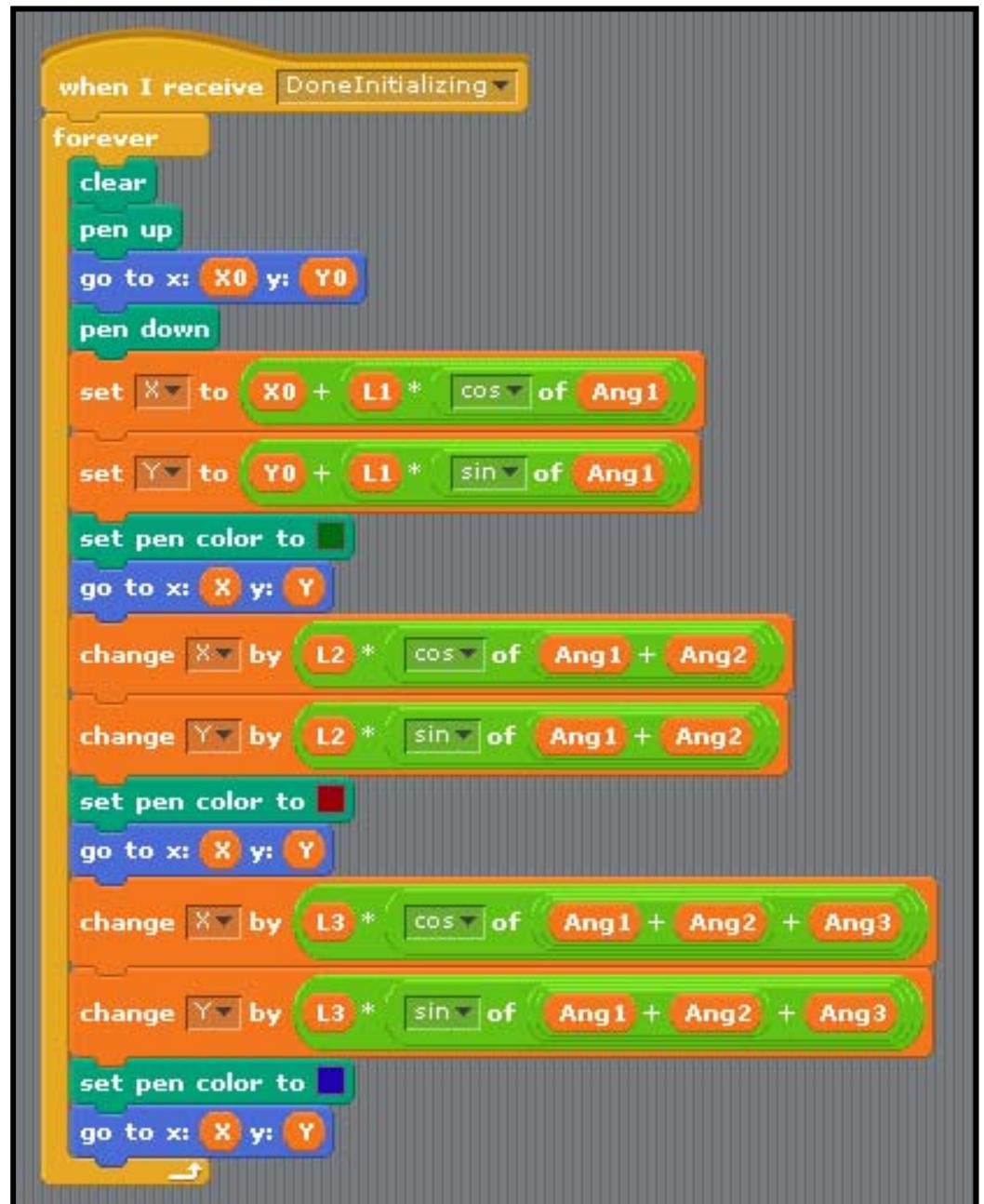
Oregon State

University

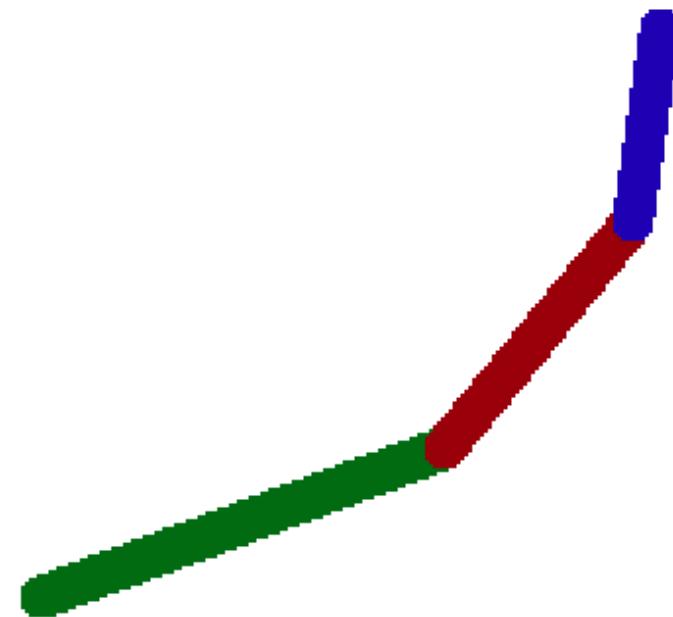
Computer Graphics

mjb – August 1, 2018

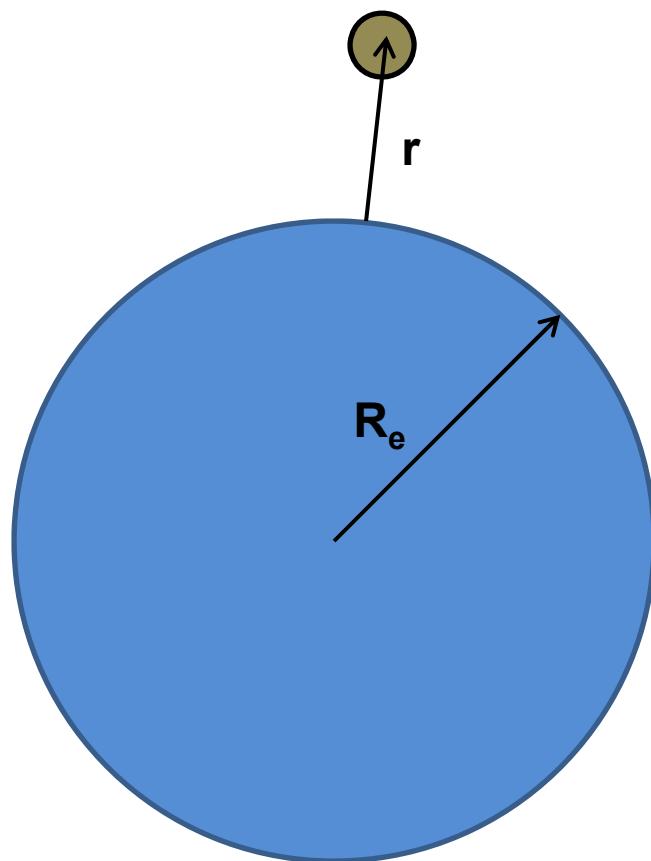
Forward Kinematics



Forward Kinematics



Orbital Mechanics



$$P = \sqrt{\frac{4\pi^2(R_e + r)^3}{gR_e^2}} \quad \theta = \frac{360}{P} t$$

Use:

km for length units
minutes for time units

$$g = 35.28 \text{ km/min}^2 \\ R_e = 6400 \text{ km}$$

Scale animation time:

The Scratch **timer** returns time in seconds, but we will treat it as minutes.

We will also use a **TimeScale** factor.

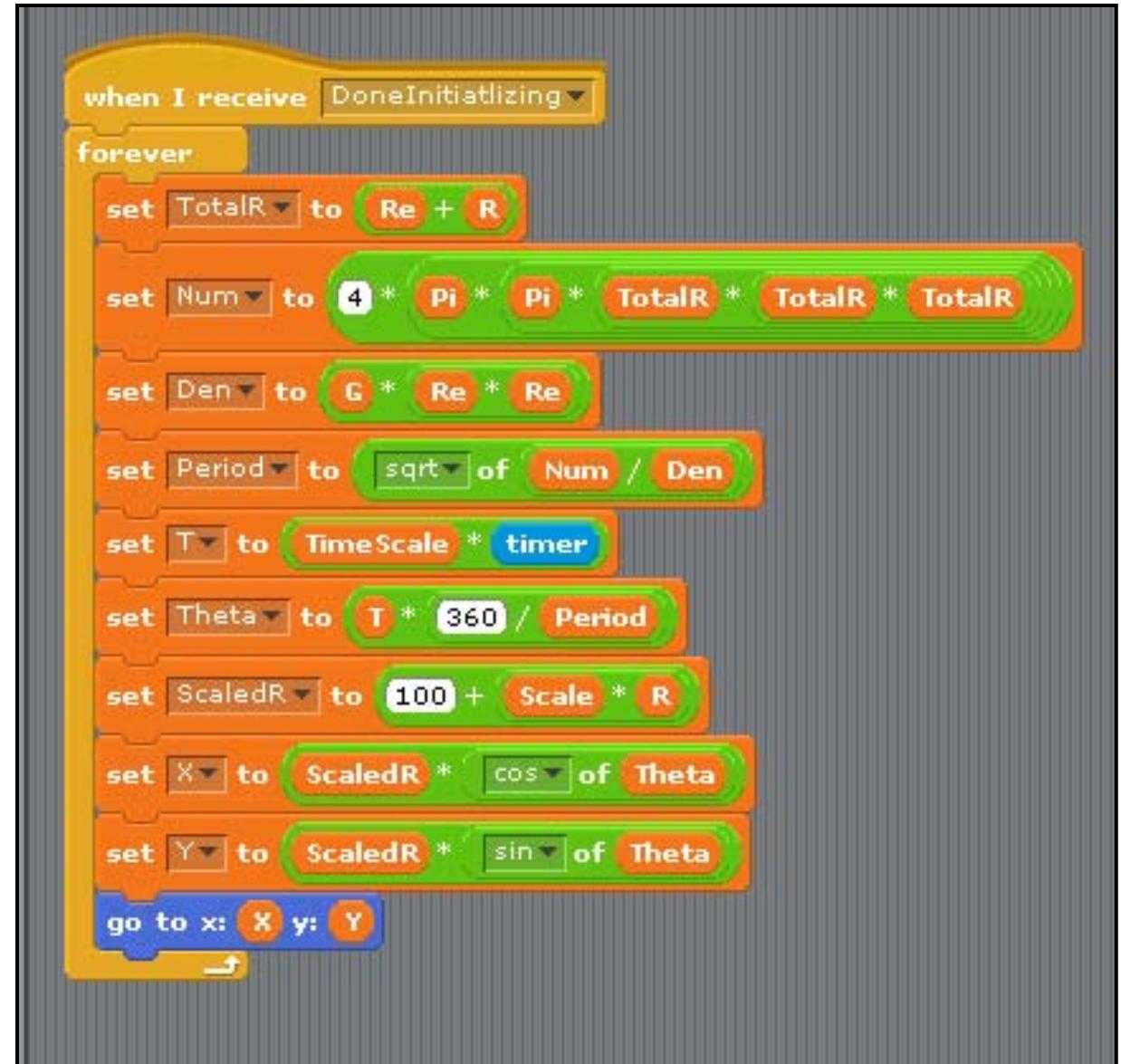
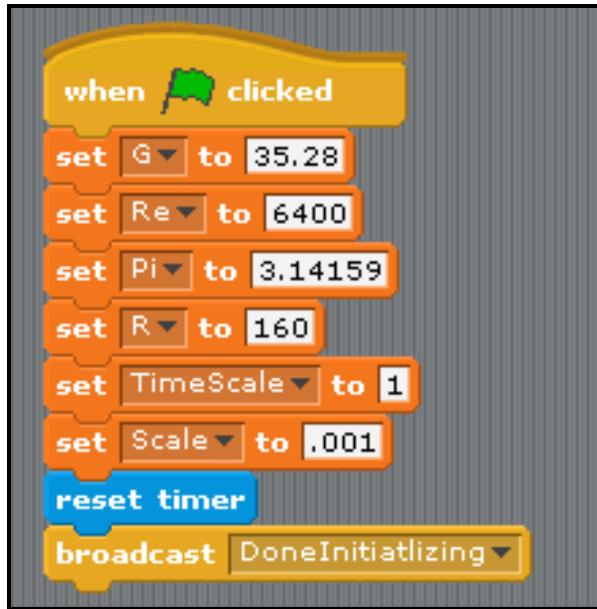


Oregon State

University

Computer Graphics

Orbital Mechanics

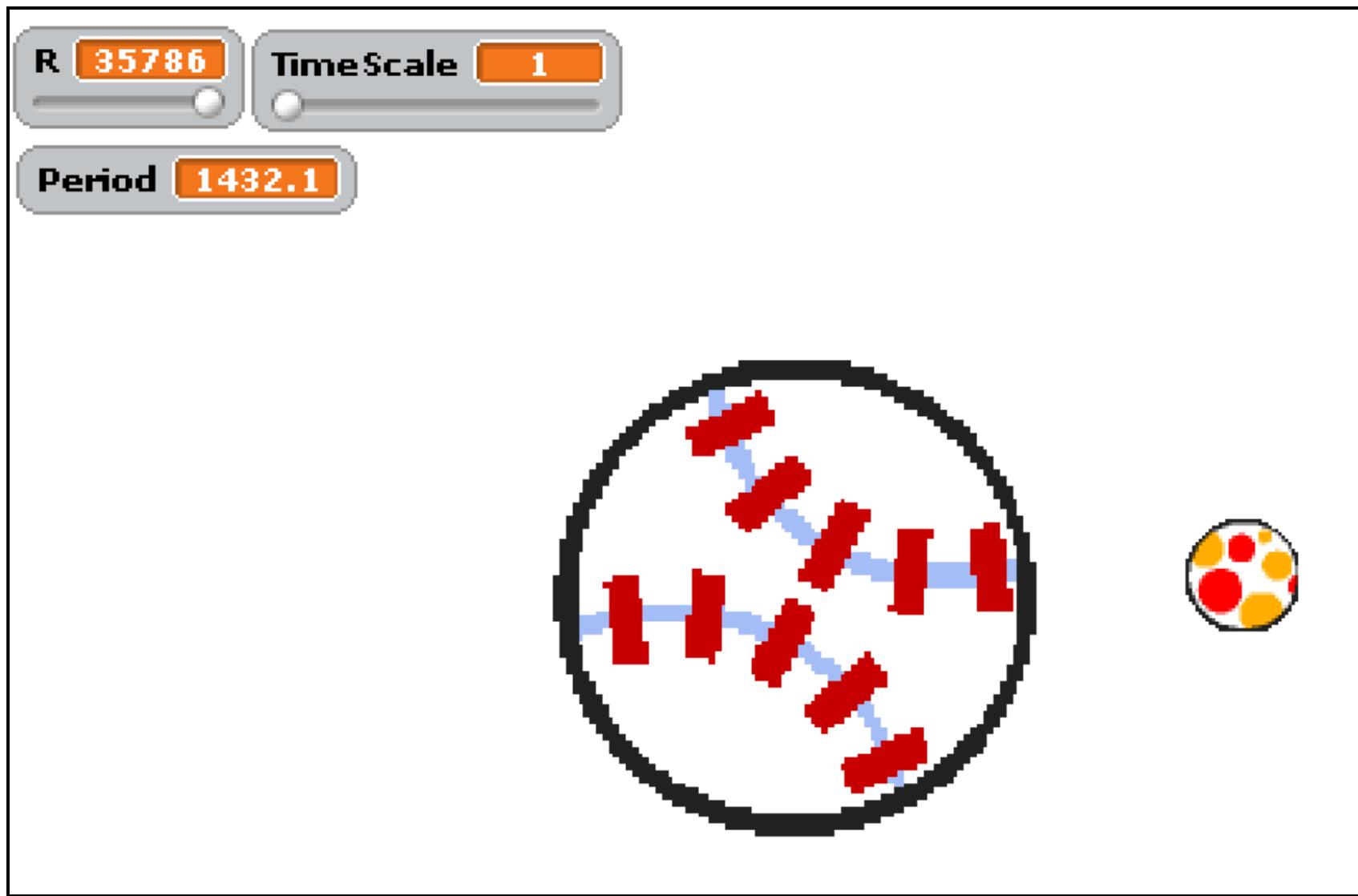


Oregon State
University

Computer Graphics

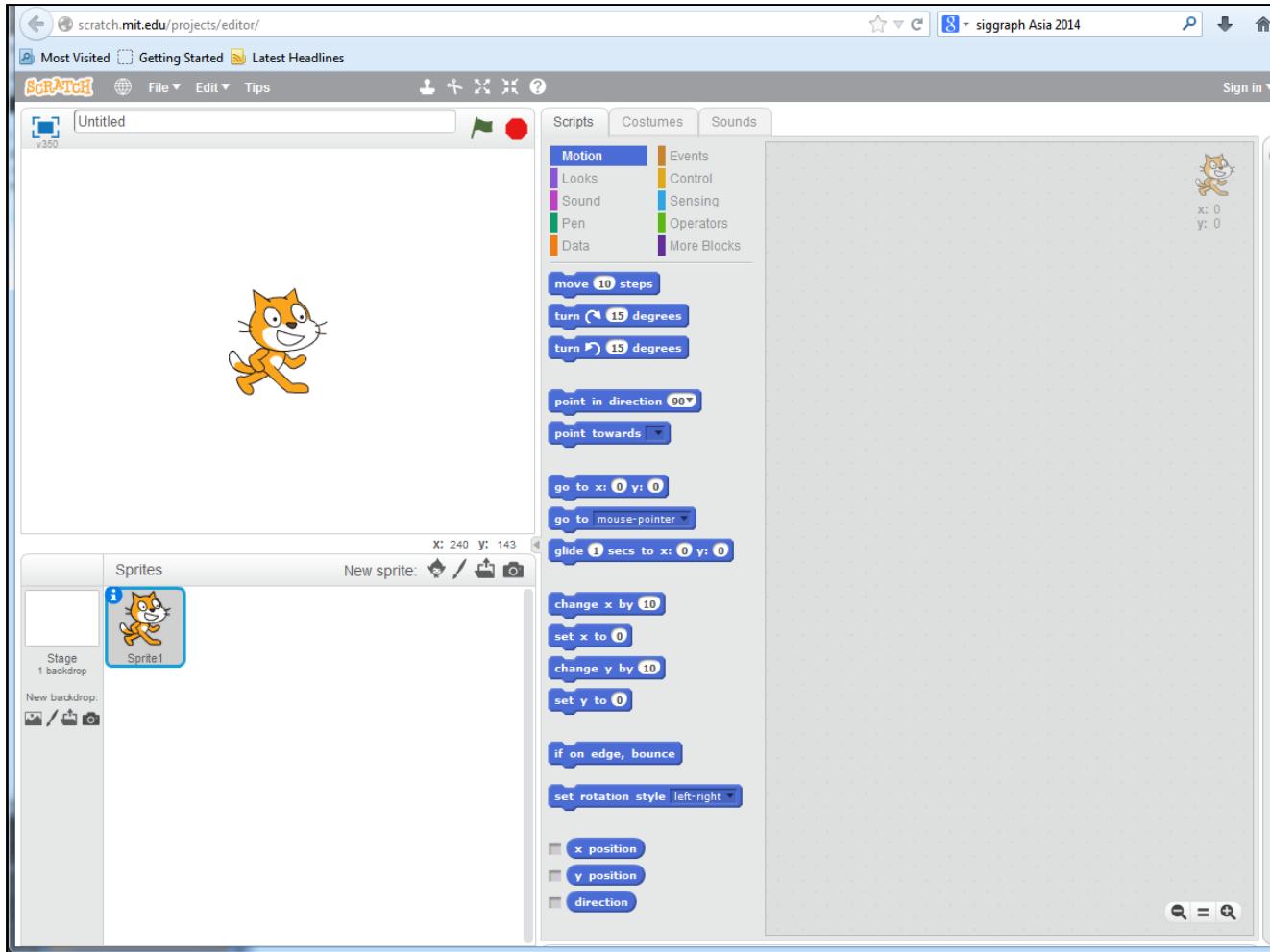
Orbital Mechanics

90



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

<http://scratch.mit.edu/scratch2download/>



Oregon State
University

Computer Graphics

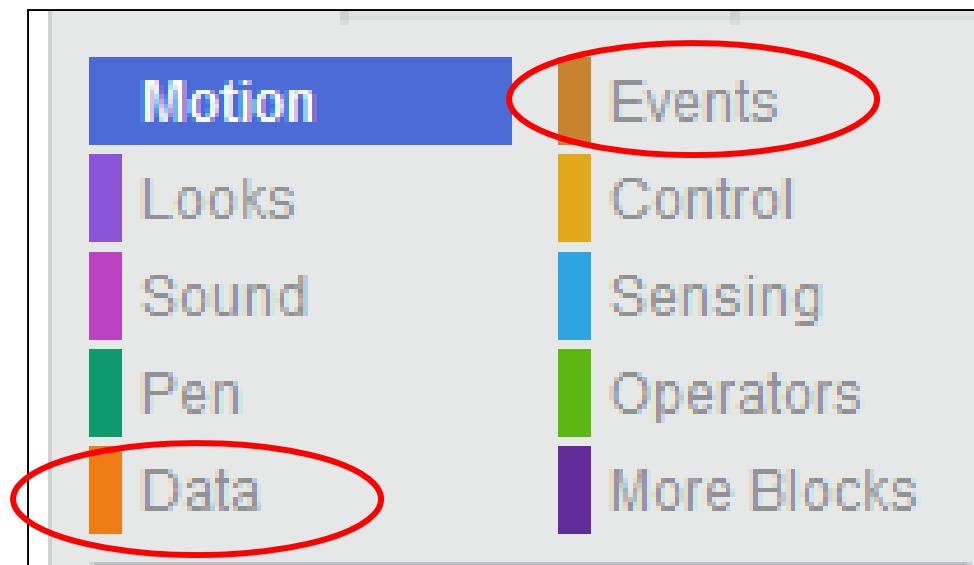
Differences in Scratch 2.0

92

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

<http://scratch.mit.edu/scratch2download/>

- The online version can run from a web browser
- Clones
- Code Blocks
- Some of the categories have been renamed
 - Control → Events + Control
 - Variables → Data
- Motion detection



Oregon State
University

Computer Graphics

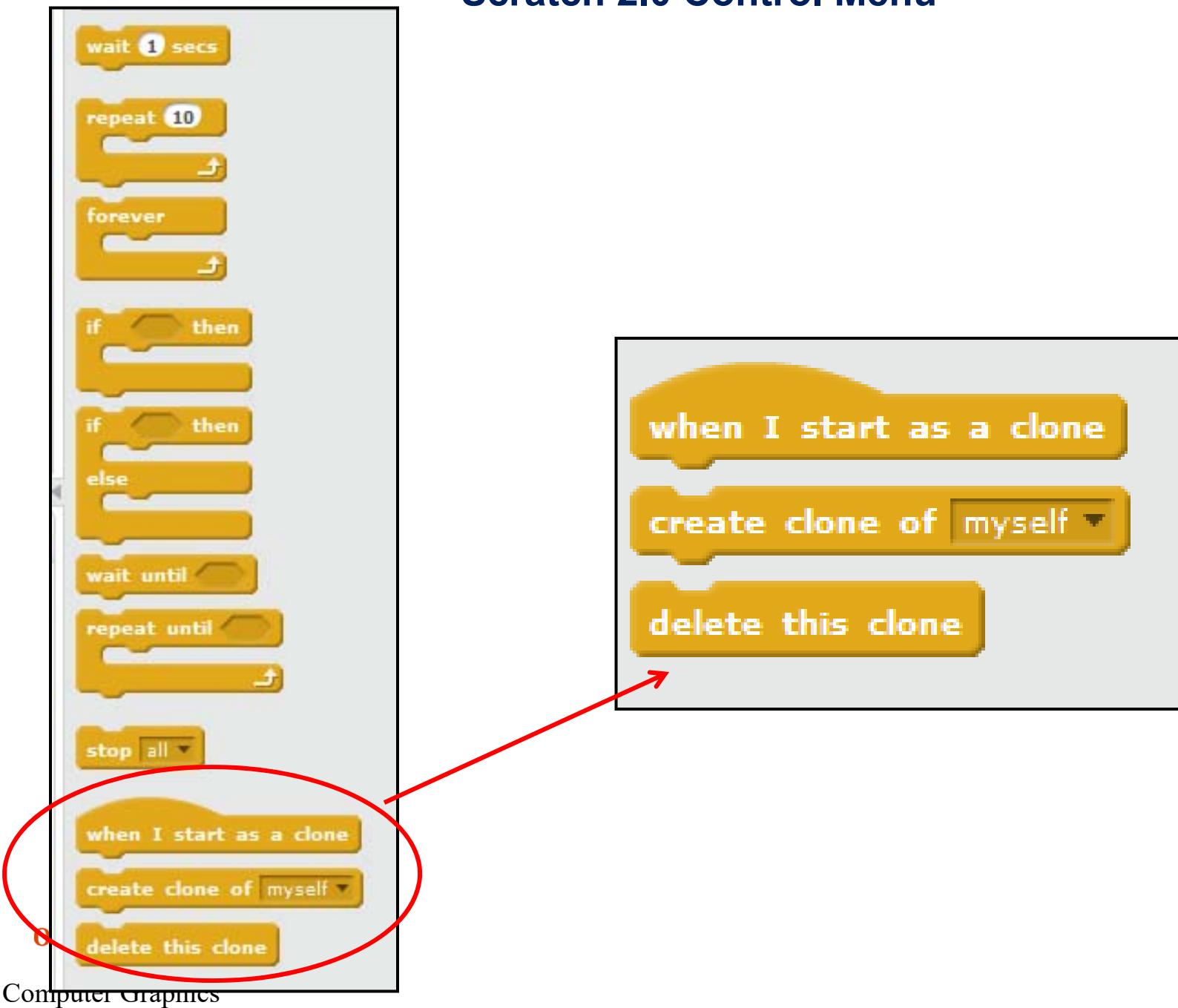
mjb – August 1, 2018

Scratch 2.0 Events Menu

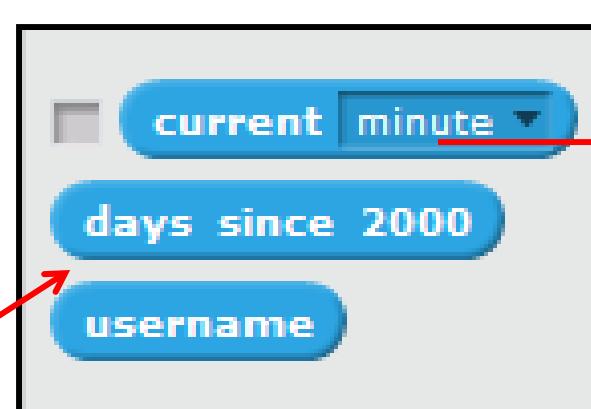
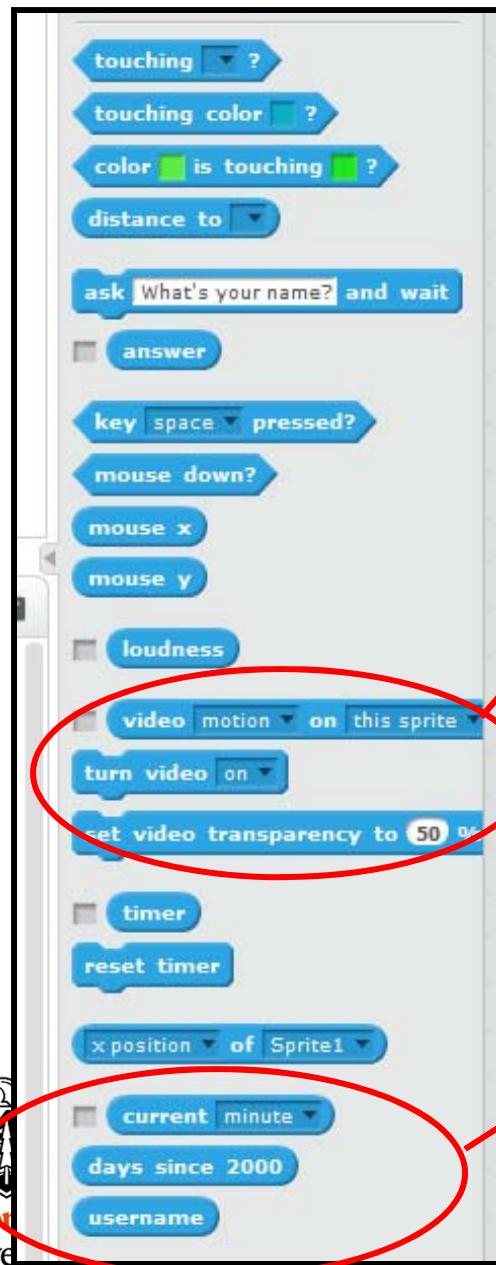
93



Scratch 2.0 Control Menu

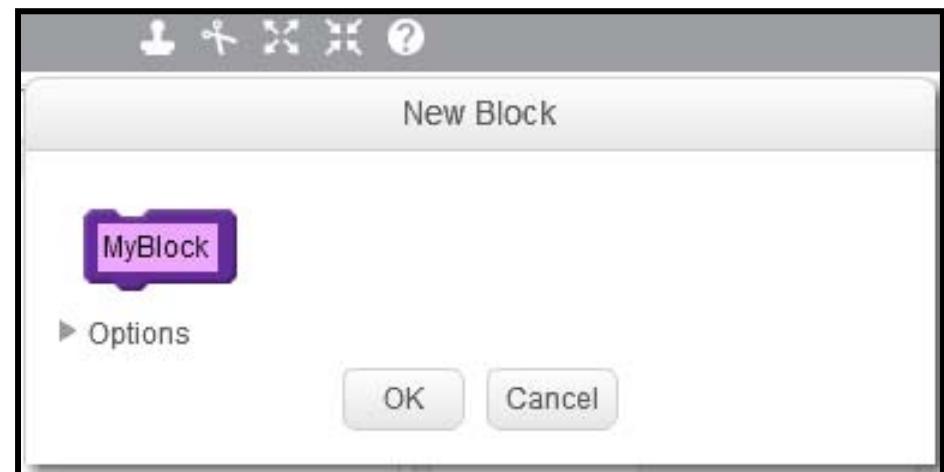
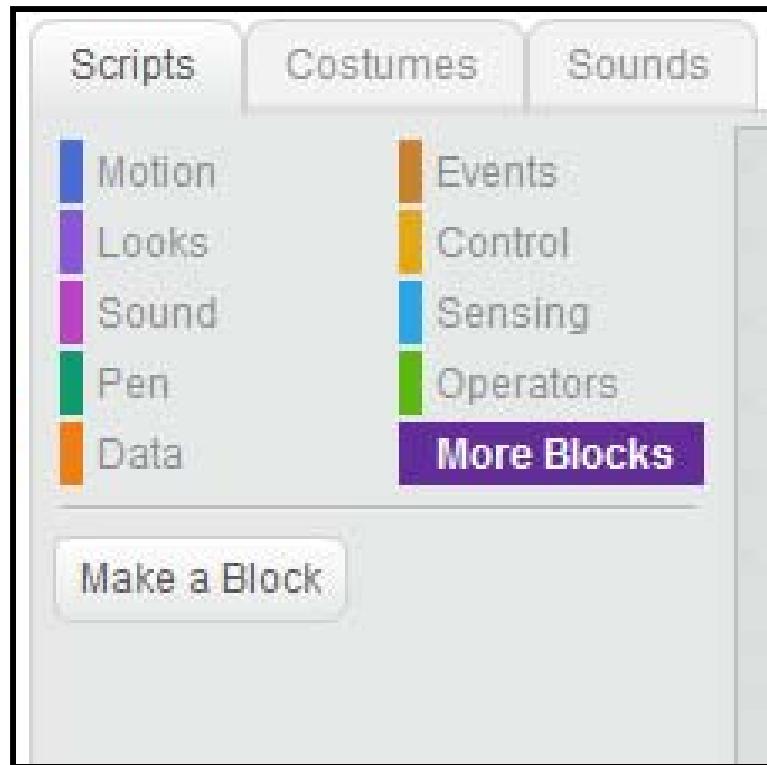


Scratch 2.0 Sensing Menu



Blocks (aka Procedures, Functions, Subroutines)

96



Oregon State

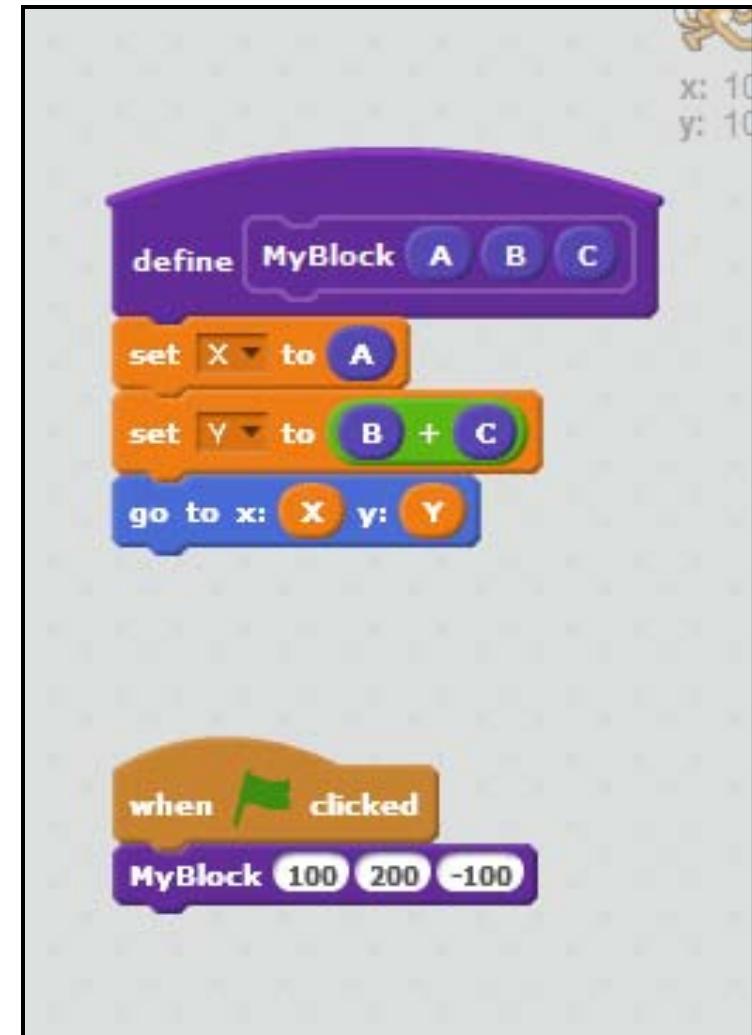
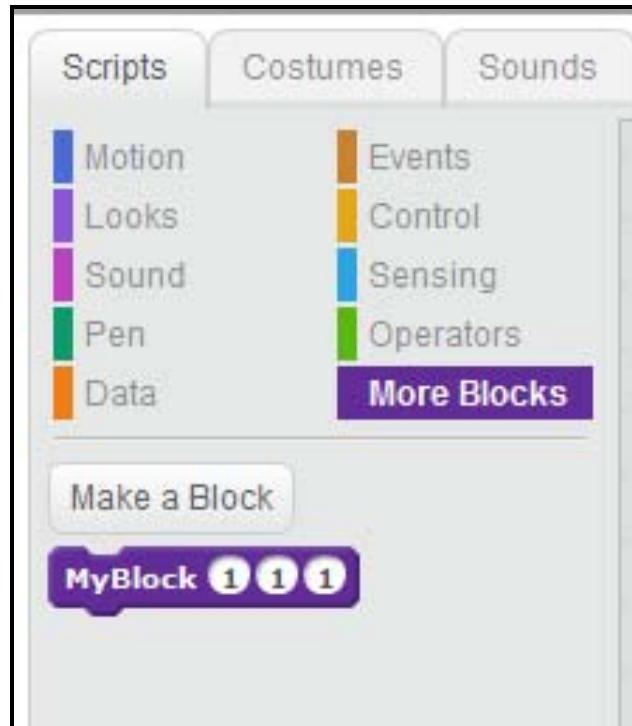
University

Computer Graphics

mjb – August 1, 2018

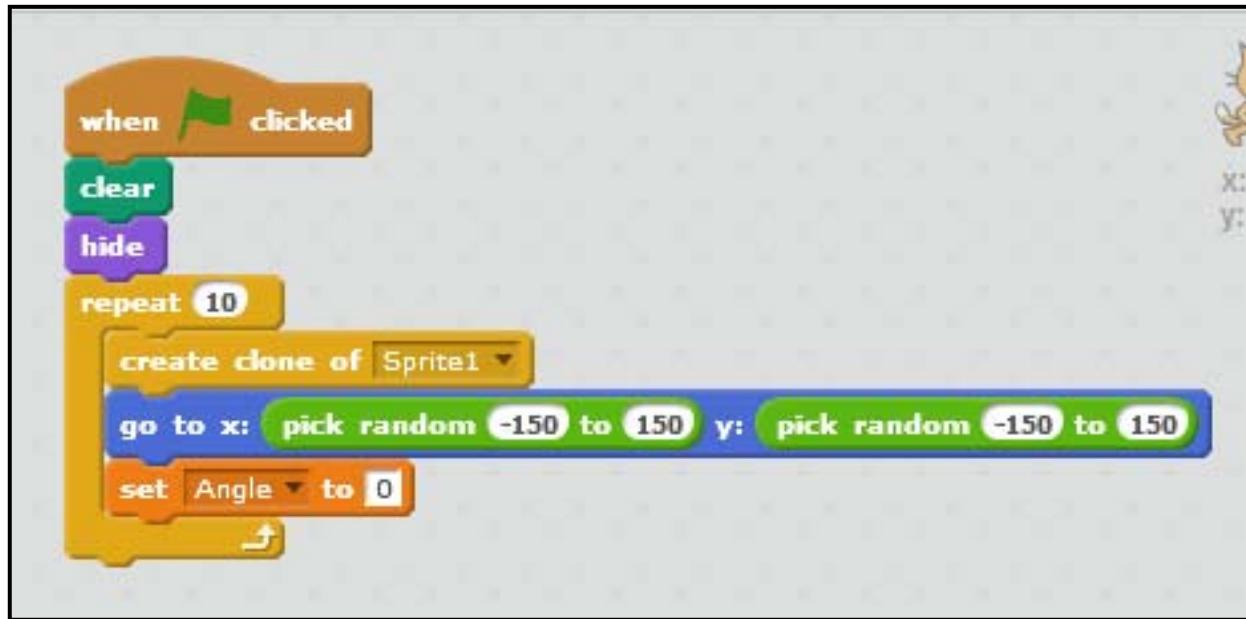
Blocks (aka Procedures, Functions, Subroutines)

97

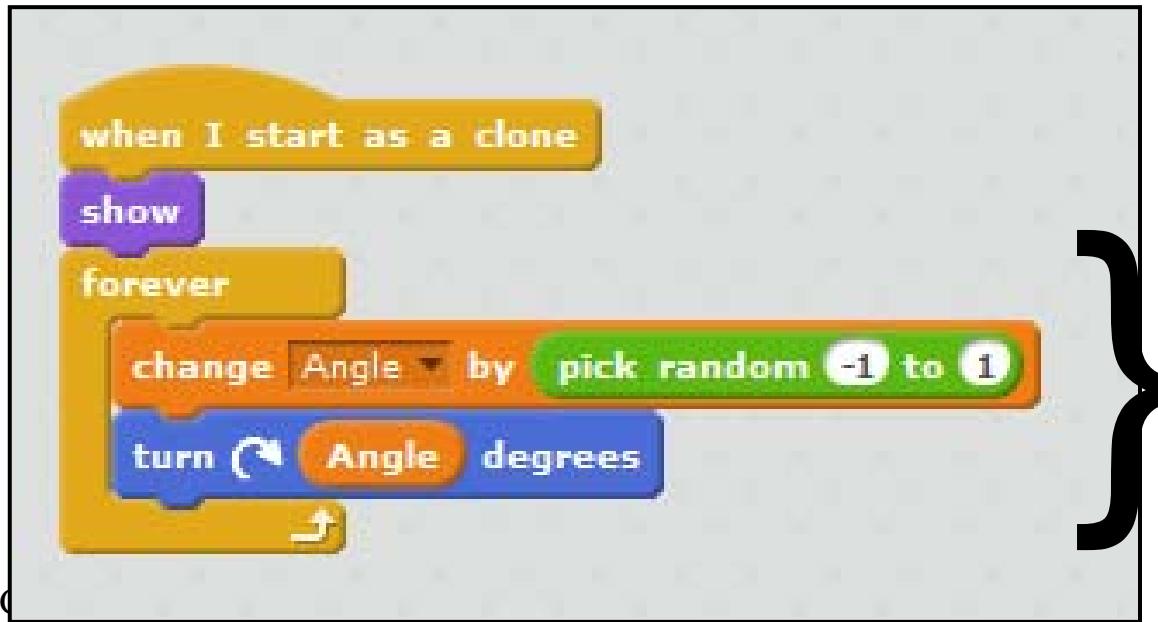


Send in the Clones

98



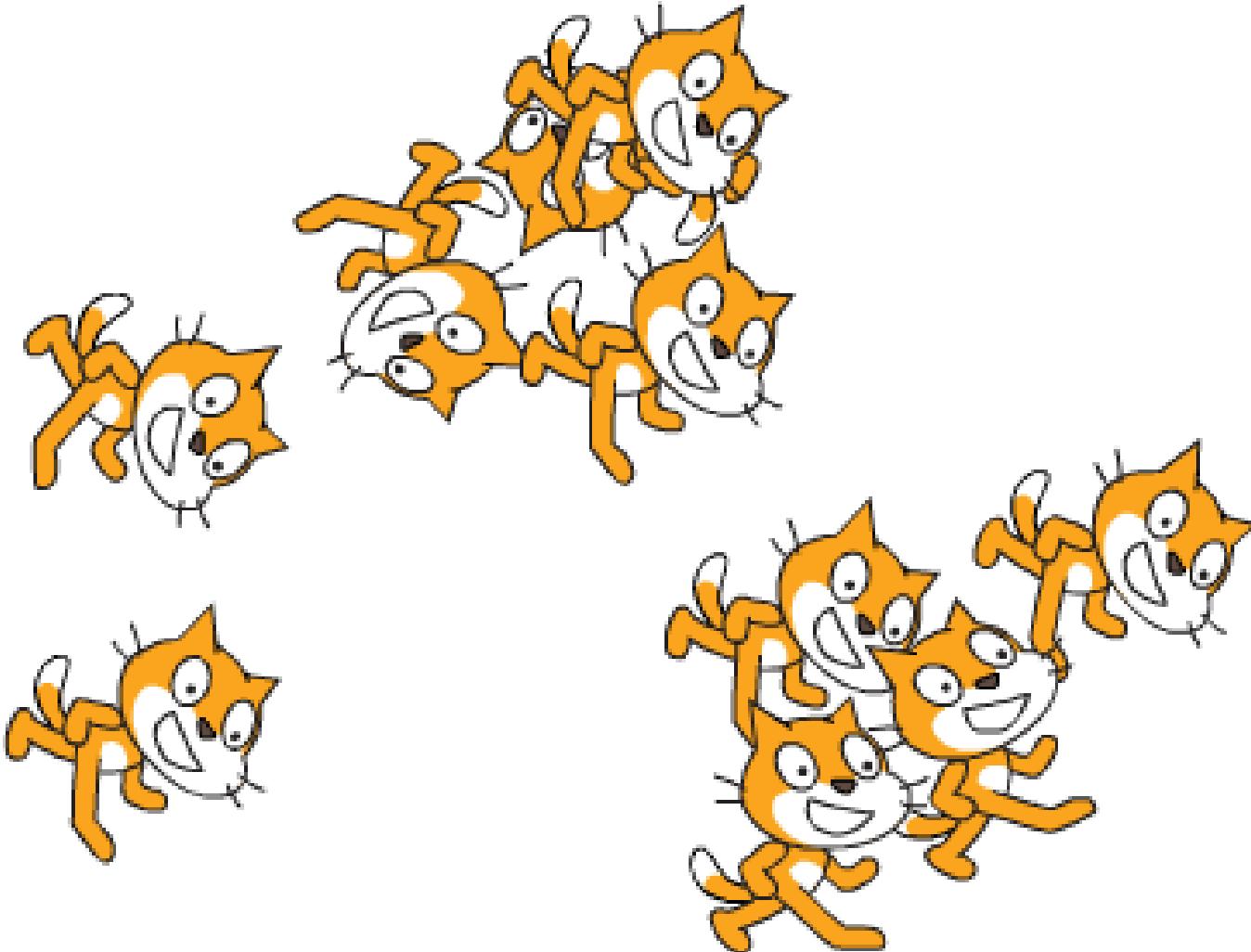
Creates a clone of the cat sprite and a separate copy of all its variables, properties, and scripts.



This script gets run for each of the clones. Sure beats writing this 10 times!

Send in the Clones

99



Oregon
University

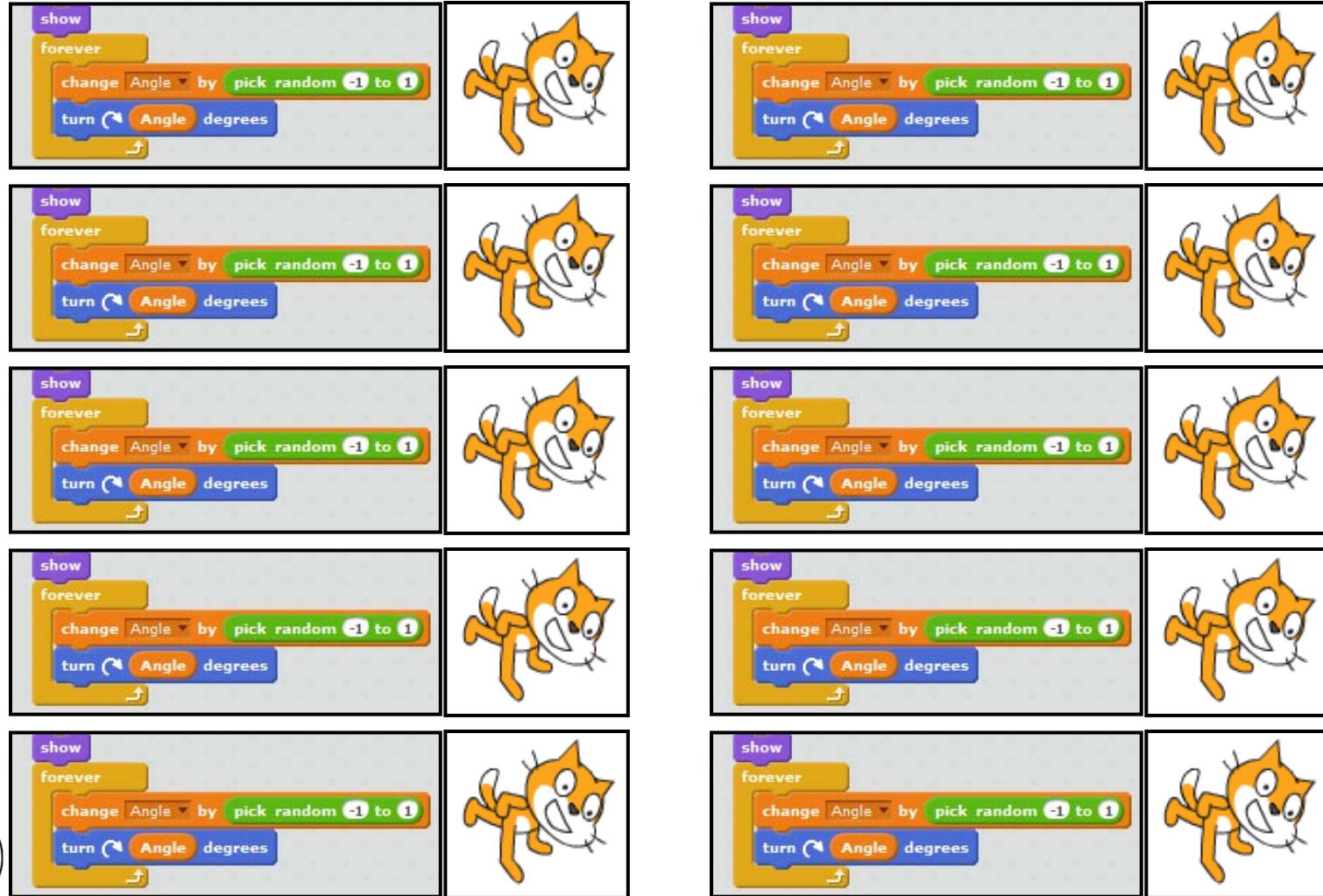
Computer Graphics

mjb – August 1, 2018

Send in the Clones

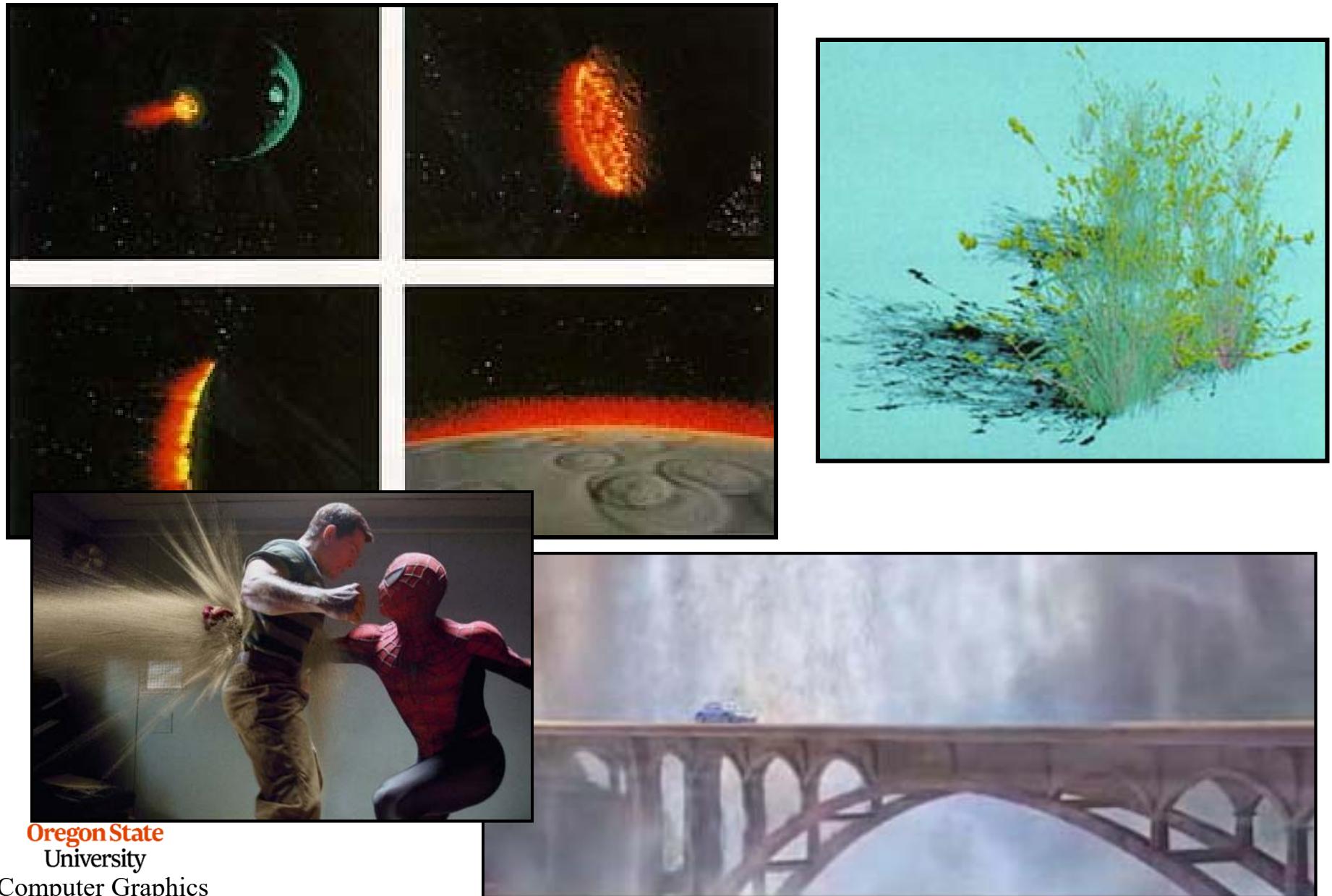
100

Think of it as 10 simultaneous scripts are running, each controlling its own sprite:



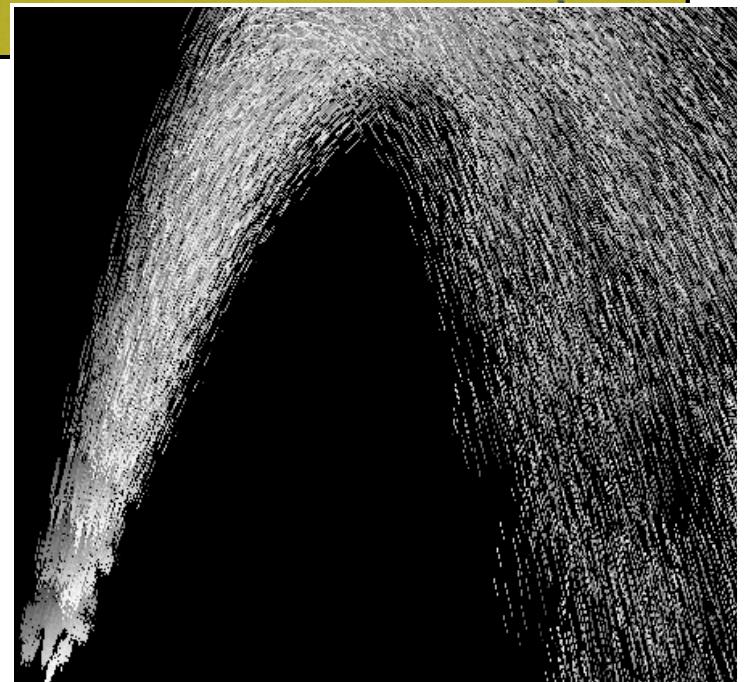
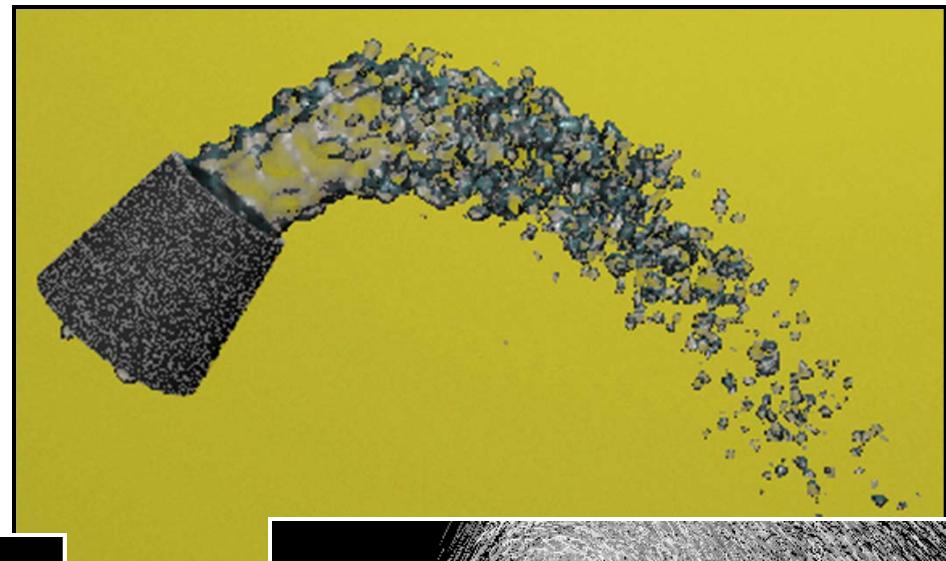
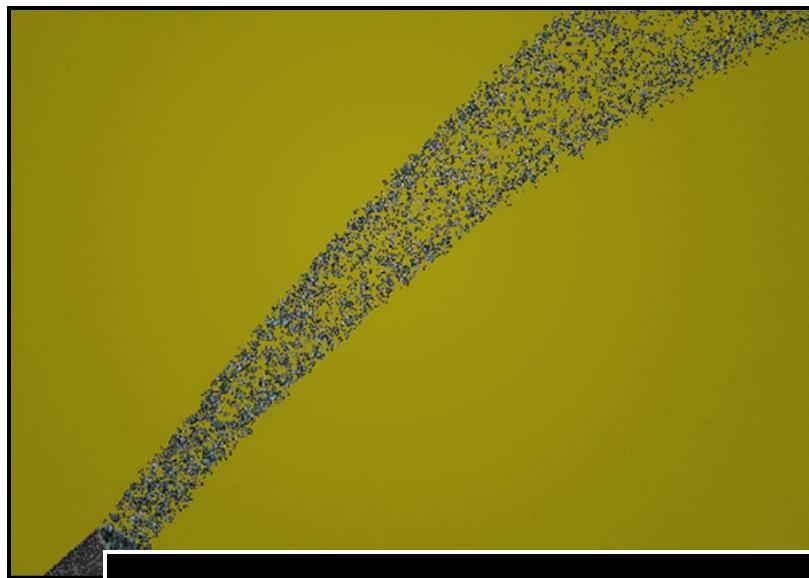
Using Clones to Create a Particle System

101



Particle System

102



Particle System – Doesn't Have to Be Actual “Particles”

103



Oregon State
University

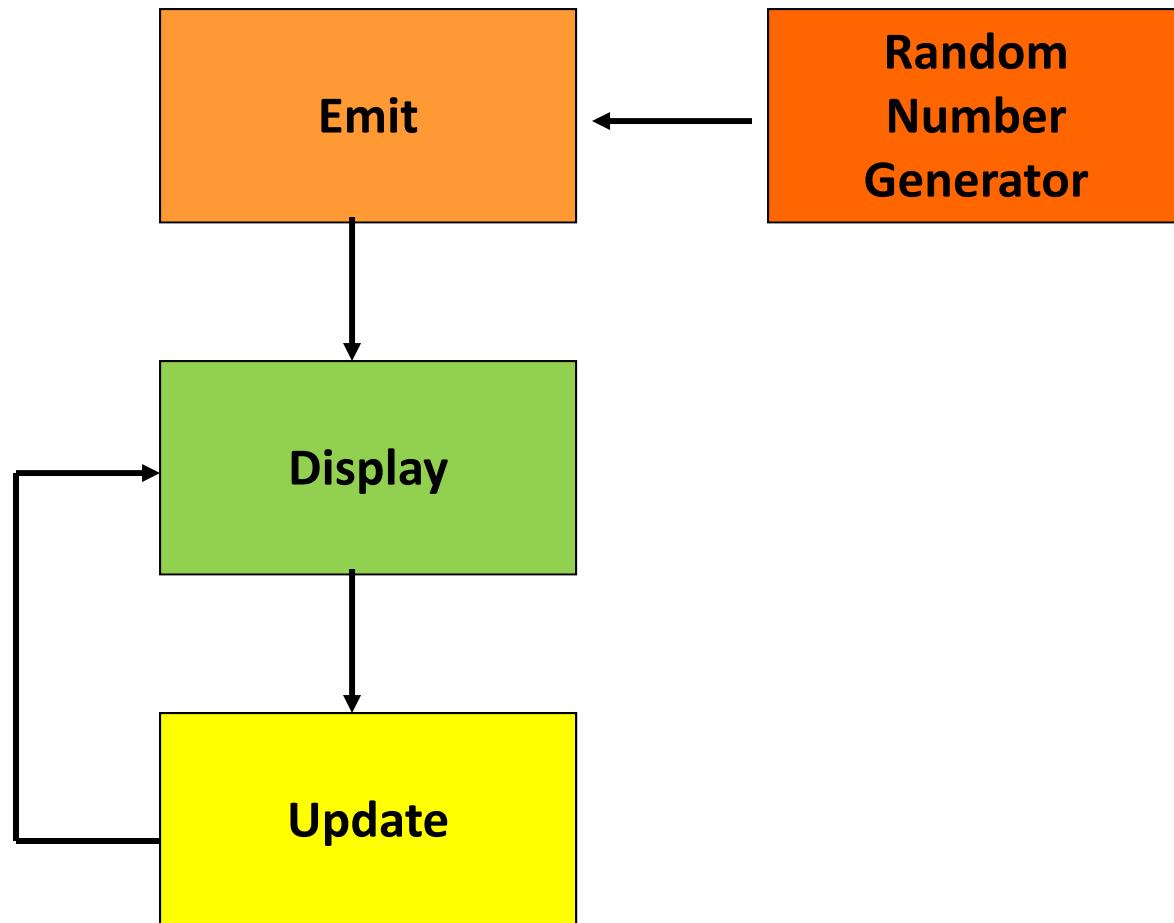
Computer Graphics

mjb – August 1, 2018

Particle System

104

The basic process is this:

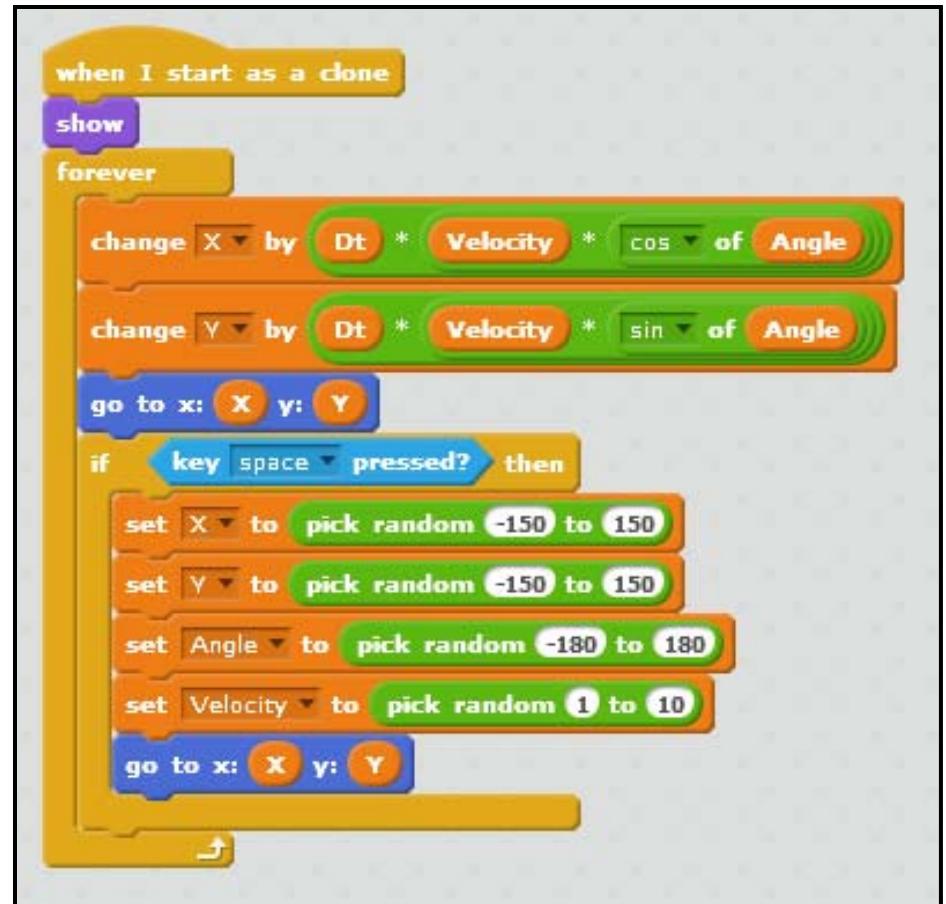
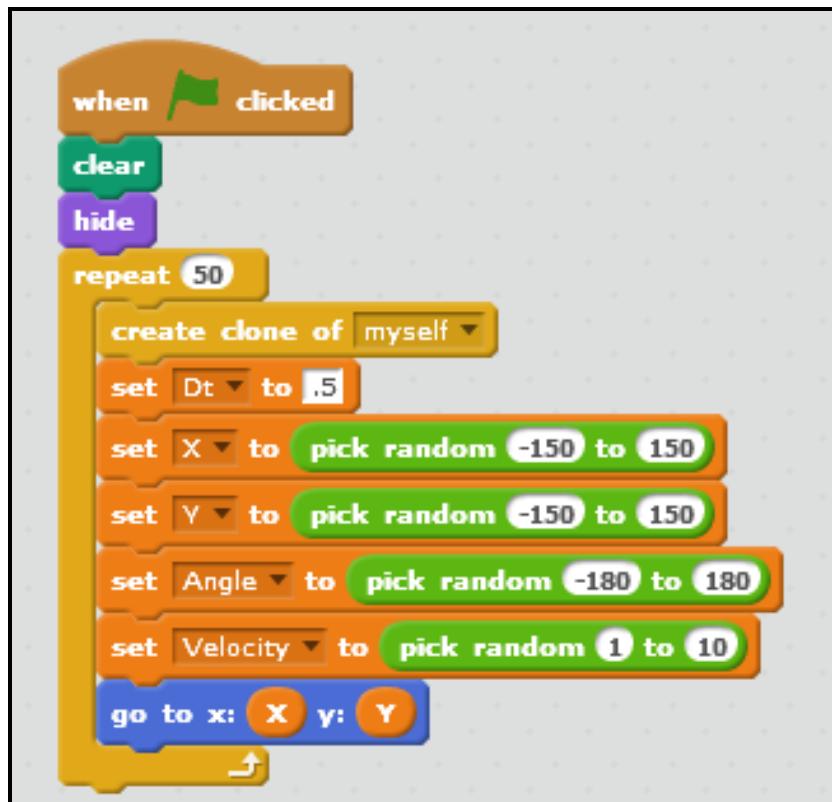


Oregon State
University

Computer Graphics

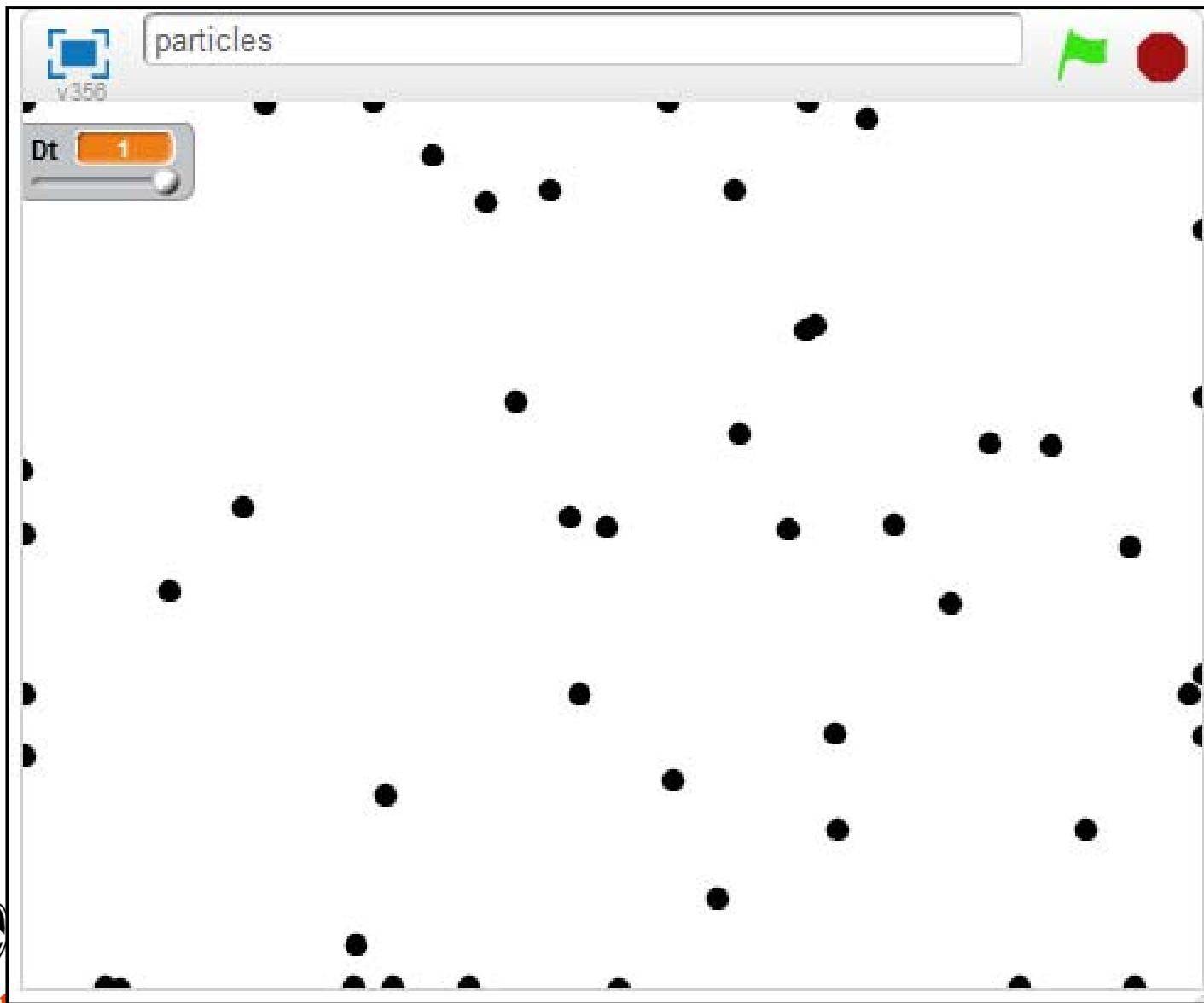
mjb – August 1, 2018

Particle System



Particle System

106



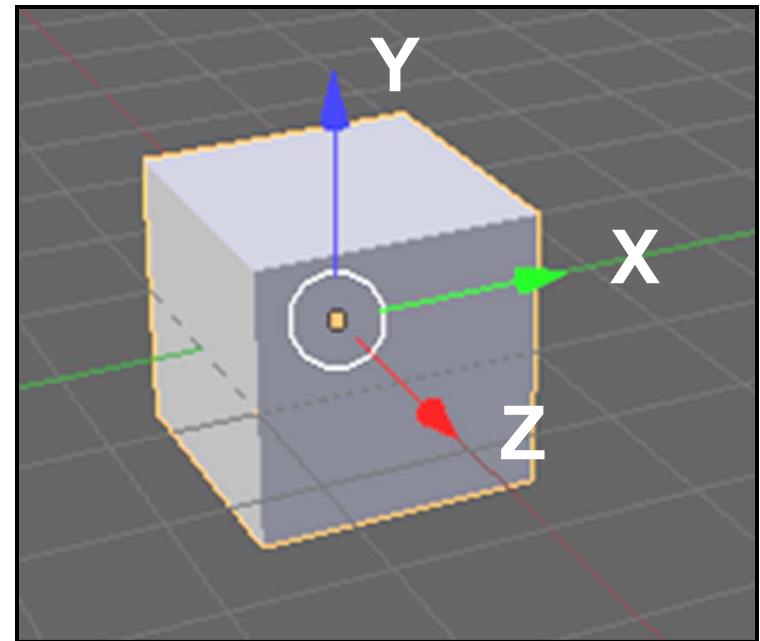
A 3D Right-handed Coordinate System:

First rotating about the Y axis

$$x' = x \cos \theta_y + z \sin \theta_y$$

$$y' = y$$

$$z' = -x \sin \theta_y + z \cos \theta_y$$



Then rotating about the X axis

$$x'' = x'$$

$$y'' = y' \cos \theta_x - z' \sin \theta_x$$

$$z'' = y' \sin \theta_x + z' \cos \theta_x$$



Oregon State
University

Computer Graphics

Three Dimensions

Note: a great use for Blocks (aka Procedures, Functions, Subroutines))

```

define ThreeD [X] [Y] [Z]
set [C] to [cos of Angle]
set [S] to [sin of Angle]
set [cx] to [cos of AngleX]
set [sx] to [sin of AngleX]
set [xp] to [X * C + Z * S]
set [yp] to [Y]
set [zp] to [Z * C - X * S]
set [xpp] to [xp]
set [ypp] to [yp * cx - zp * sx]
set [zpp] to [zp * cx + yp * sx]
set [zpp] to [150 - zpp]
go to x: [xpp] y: [ypp]

```

The Scratch script defines a procedure named "ThreeD" which takes three parameters: X, Y, and Z. It calculates the cosine and sine of the current angle and the cosine and sine of the angle around the X-axis. It then performs vector transformations to calculate new coordinates xp, yp, and zp based on the input X, Y, and Z. Finally, it moves the sprite to the calculated position.

```

define Square [Z]
pen up
ThreeD [MinusSize] [MinusSize] [Z]
pen down
ThreeD [Size] [MinusSize] [Z]
ThreeD [Size] [Size] [Z]
ThreeD [MinusSize] [Size] [Z]
ThreeD [MinusSize] [MinusSize] [Z]

```

The Scratch script defines a procedure named "Square" which takes one parameter: Z. It first lifts the pen, then calls the "ThreeD" procedure with parameters MinusSize, MinusSize, and Z. It then lowers the pen, calls "ThreeD" with parameters Size, MinusSize, and Z, then with parameters Size, Size, and Z, then with parameters MinusSize, Size, and Z, and finally with parameters MinusSize, MinusSize, and Z.

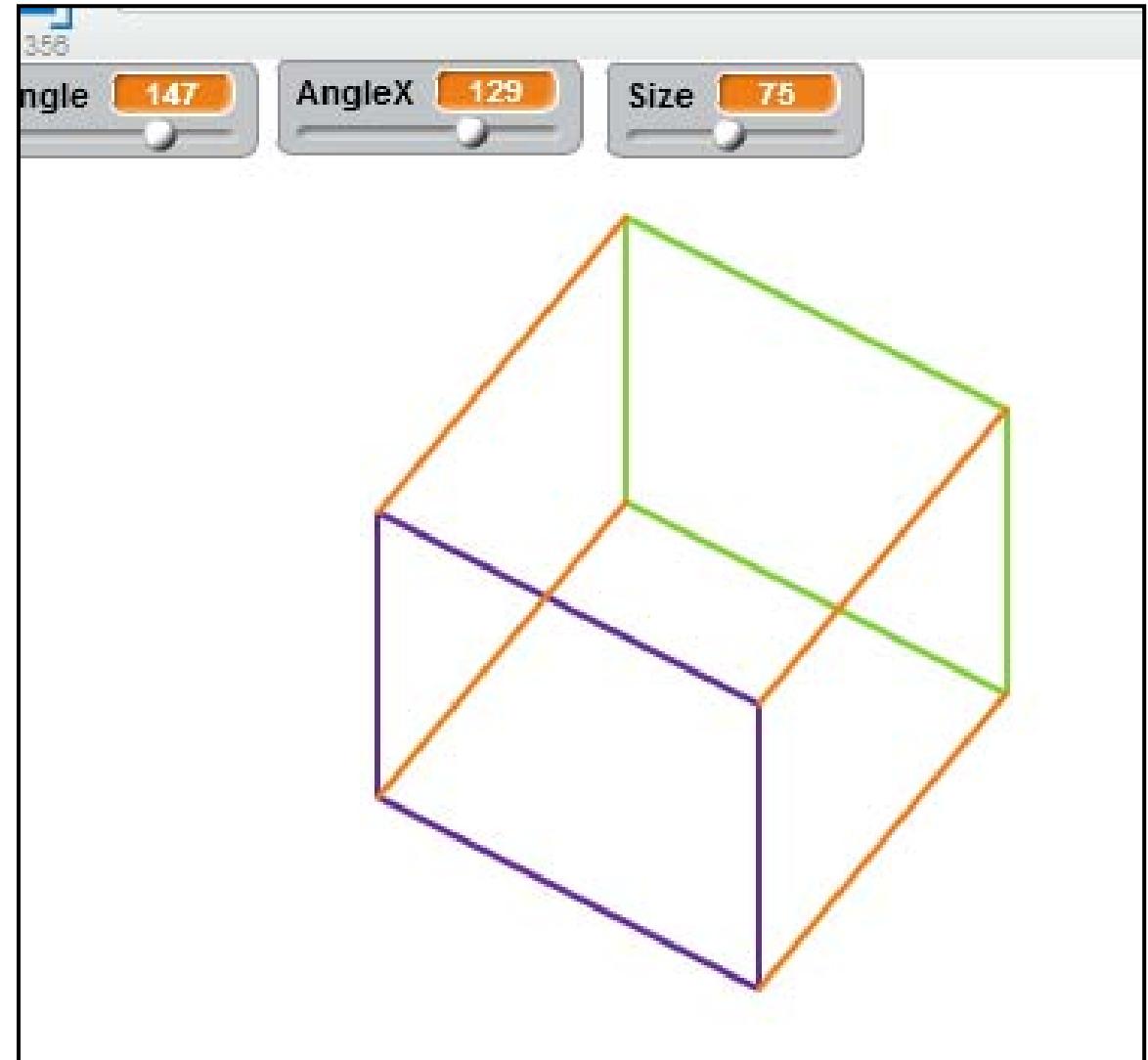
```

define LineZ [X] [Y]
pen up
ThreeD [X] [Y] [MinusSize]
pen down
ThreeD [X] [Y] [Size]

```

The Scratch script defines a procedure named "LineZ" which takes two parameters: X and Y. It first lifts the pen, then calls the "ThreeD" procedure with parameters X, Y, and MinusSize. It then lowers the pen and calls "ThreeD" with parameters X, Y, and Size.

Three Dimensions



How Scratch Relates to C/Java Programming

110

The image shows a Scratch script editor window. On the left, there's a sidebar with buttons for 'Make a variable', 'Delete a variable', and several checkboxes for variables: 'Count' (checked), 'Delay' (checked), 'NegativeSize' (unchecked), and 'Size' (checked). Below these are four orange control blocks: 'set Count to 0', 'change Count by 1', 'show variable Count', and 'hide variable Count'. On the right, there are three code snippets corresponding to these blocks:

- int Count ;
- Count = 0 ;
- Count = Count + 1 ;



Oregon State

University

Computer Graphics

How Scratch Relates to C/Java Programming

111



```
sleep( 1 );
```

```
for( ; ; )  
{  
    statements ...  
}
```

```
for( i = 1 ; i <= 10; i = i + 1 )  
{  
    statements...  
}
```

How Scratch Relates to C/Java Programming

112



```
while( i < j )  
{  
    statements...  
}
```

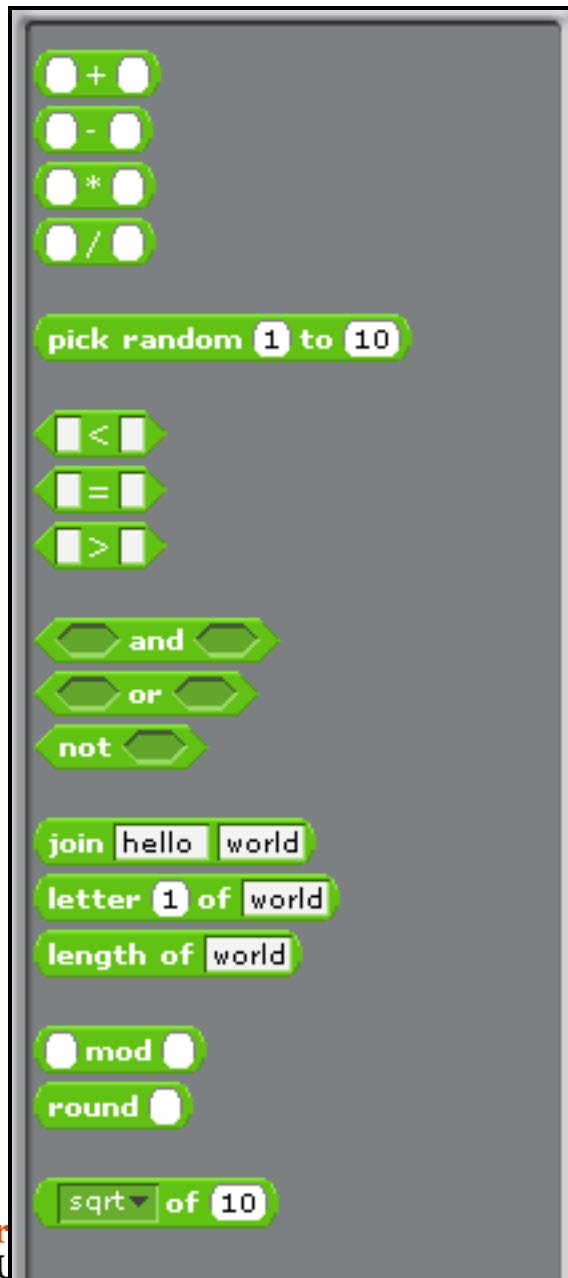
```
if( i < j )  
{  
    statements ...  
}
```

```
if( i < j )  
{  
    statements ...  
}  
else  
{  
    other statements ...  
}
```

```
do  
{ ;  
}  
while( ! ( i < j ) );
```

```
do  
{  
    statements ...  
} while( ! ( i < j ) );
```

How Scratch Relates to C/Java Programming



$i + j$
 $i - j$
 $i * j$
 i / j

`if(i < j)`
`if(i == j)`
`if(i > j)`

`if(i < j && j < k)`

`if(i < j || j < k)`

`If(!(i < j))`

$i \% j ;$

How Scratch Relates to C/Java Programming

114

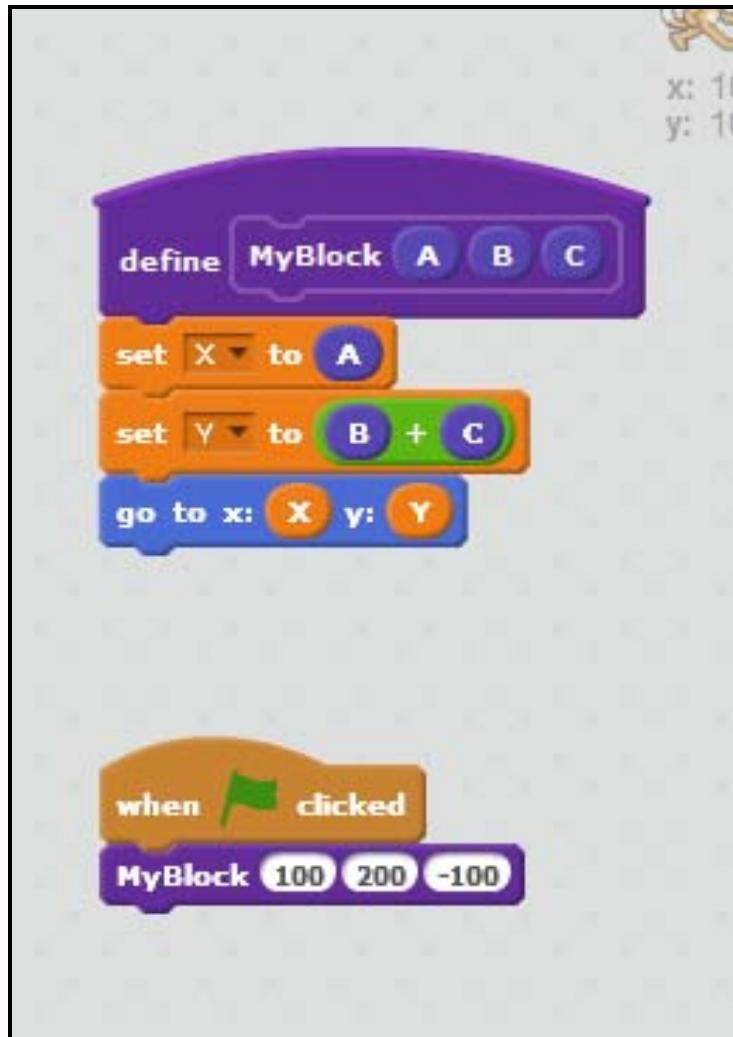


```
public class Flashing implements ActionListener
{
    //declare what event will be listening:
    button.addActionListener(this);
    ...

    // declare what to do when the event happens:
    public void actionPerformed(ActionEvent e)
    {
        // do something to make the display flash:
        ...
    }
}
```

How Scratch Relates to C/Java Programming

115



```
void  
MyBlock( int A, int B, int C )  
{  
    int X = A;  
    int Y = B + C;  
    glVertex2i( X, Y );  
}
```

```
MyBlock( 100, 200 -100 );
```



Oregon State
University
Computer Graphics

mjb – August 1, 2018

Best Practices, I

Start with the goals in mind

- List of comments

- State Diagrams

- Data Flow Diagrams

Don't hard-code constants into the code – use variables so it is clear what you are doing

Use variables to simplify expressions

Pick good variable names

Initialize all variables, even to zero

Modularize

- Separate sections with Broadcasts

- Separate initializing from executing

- Use Function Blocks

Generalize

For example, don't use **timer**, use **t = scale*timer**



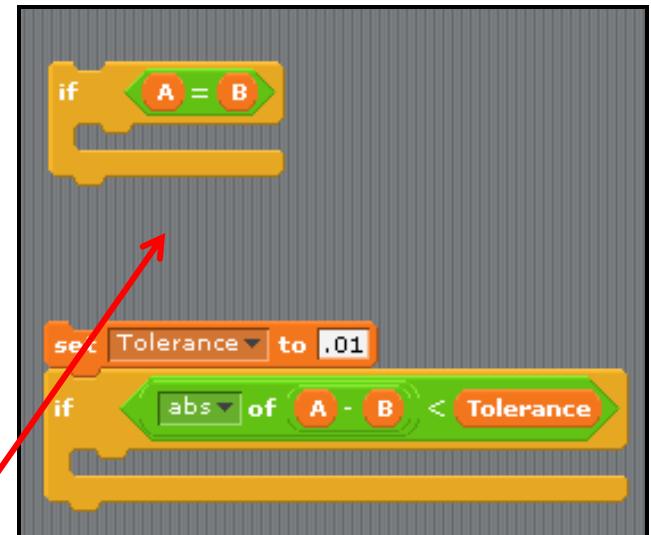
Oregon State
University

Computer Graphics

Best Practices, II

117

Remember that, unless specified, Scratch commands happen “instantaneously” and then the script moves on to the next instruction

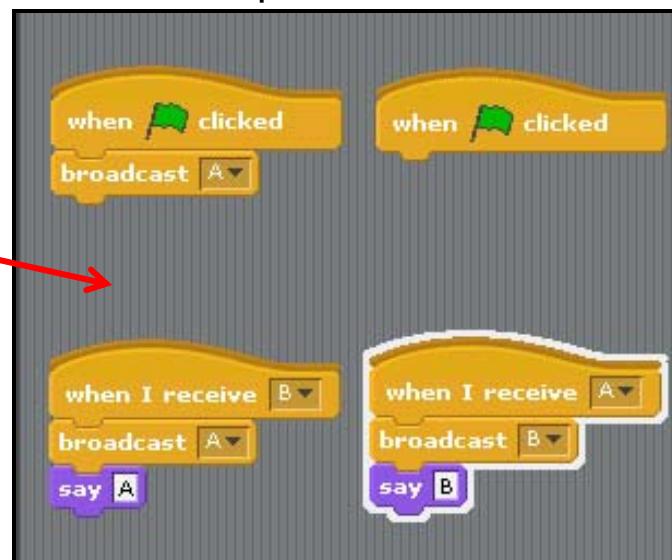


Be careful you get loop conditions correct

Floating point numbers rarely exactly equal each other

Use < or > to compare them

Use Absolute Value function to compare them



Debugging

“Print” variables into the scene

Enlarge your view of the script

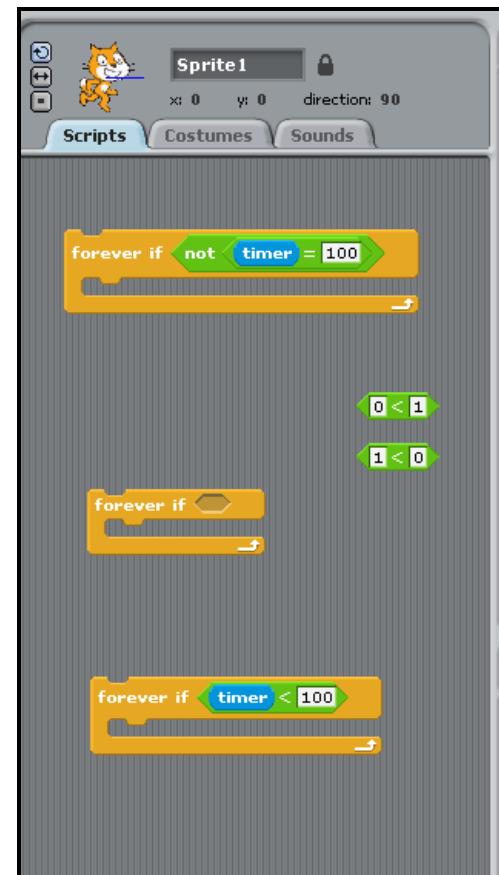
Single-stepping

If-statements that make something visible happen

Sound

Say or Think

Color change



Oregon State
University

Computer Graphics