

Scratch



AN INTRODUCTION TO PROGRAMMING
COIS 1010H - Digital World

What is Scratch?

- ▶ A “drag-n-drop” programming language created by MIT, designed specifically for teaching programming concepts
- ▶ Used to create basic games or animations
 - ▶ Several scratch terms and concepts are taken right from real-world game development
 - ▶ 2-D graphics
- ▶ Desktop application or online web-based software

<http://scratch.mit.edu>

<https://en.scratch-wiki.info>

<https://scratch.mit.edu/help/>



Accessing the Software

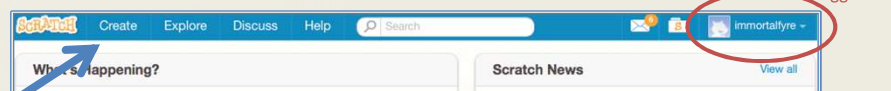
- ▶ The Web-based version of Scratch is probably the easiest way to get started.
- ▶ It doesn't require the install of any software, and will work in any major modern browser.
 - ▶ Note: Internet Explorer does something produce a strange screen stutter, so Chrome, Firefox or Safari might be better
- ▶ Projects are all stored in Scratch's Cloud within your account, so files can't be lost
- ▶ They are also accessible from any computer
- ▶ However, if you need to work without internet (or with dial-up), you download and use the desktop version



Getting Started

- ▶ To use Scratch online, you will need to create an account
 - ▶ Go to scratch.mit.edu, and click on **Join Scratch**
- ▶ Once you've created your account, you can create a new project using the **Create** tab, or **Remix** an existing project

- ▶ Note: Make sure you've created your account/signed-in before creating your project if you want your files to **save!**



- ▶ Previously created projects can be accessed from the **My Stuff** option under your username, when you're logged in

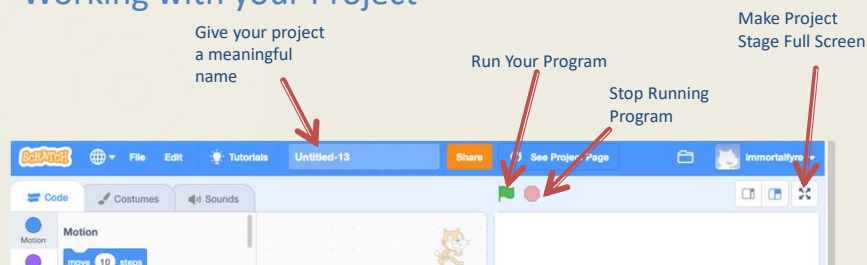
Remixing

- ▶ You can remix an existing program, using the remix button, which copies the project into your account so that you can edit it
- ▶ It's considered bad form to remix and share something without editing it to make it your own!



Naming, Saving, Running Projects

- ▶ Saving your project
 - ▶ While you're logged in, Scratch automatically saves your projects when you make changes
 - ▶ You can force a save of your project by selecting the **Save Now** option from the **File** menu
- ▶ Working with your Project



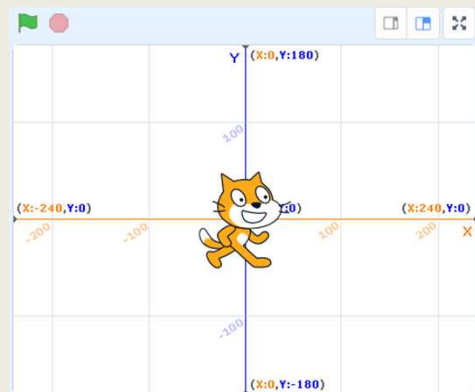
The Stage

- ▶ The **Stage** in Scratch represents the viewable/playable area of a program
- ▶ It is 480px wide by 360px tall, and its size cannot be changed
 - ▶ Note: there are several tricks to make the stage seem bigger
- ▶ Underlying the stage is an **XY grid** which can be used to refer to a position on the stage
- ▶ The programmer can change the appearance of the stage using **Backdrops**



The Grid

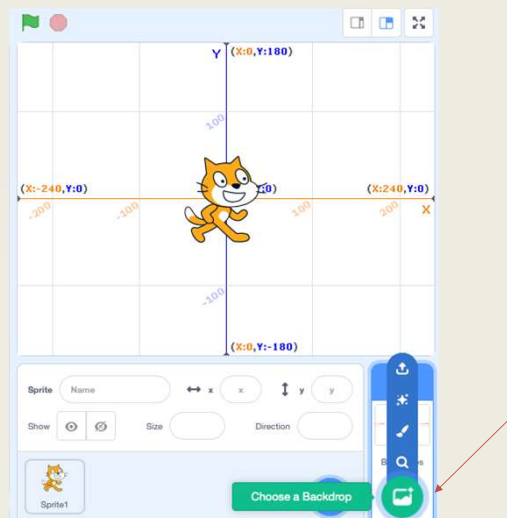
- ▶ Each point on the grid is a pixel on the stage
- ▶ The (0,0) point on the stage is exactly in the middle
 - ▶ This means the left half of the stage has negative x values, and the bottom half of the stage has negative y value
- ▶ The viewable portion of the stage goes from -240 to 240 in the X direction, and 180 to -180 in the Y direction
 - ▶ Using co-ordinates past these values will result in a position off the viewable portion of the stage



Backdrops

- ▶ Backdrops can be used to change the appearance of the stage
- ▶ There are five ways to add a backdrop to your Scratch program
 - ▶ From the existing Scratch Library
 - ▶ Adding an image file from your computer
 - ▶ To display properly it's important the image be 480px by 360px
 - ▶ Created within Scratch using its Paint-like editor
 - ▶ This can also be used to edit backdrops added using the other methods
 - ▶ Captured from the camera, if your computer has one
 - ▶ Surprise (randomly chosen backdrop)
- ▶ You can have multiple backdrops in your Scratch program, and programmatically switch between them

Adding Backdrops



Editing Backdrops

Editing Tools

Undo/Redo

All the backdrops in the program

Backdrops can either be SVG or BMP

Sprites

- ▶ Interactive objects in your scratch program are referred to as Sprites (a term which comes from Game Development)
- ▶ Every Scratch program begins with the default sprite (MIT's Scratch Mascot), which can easily be deleted with the x in the upper left corner
- ▶ Adding new Sprites can be in the same methods as backdrops, except the camera

Sprite1

x 149 y -74

Show

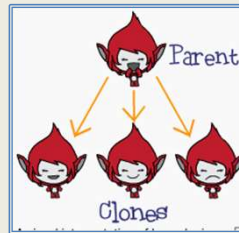
Size 100 Direction

Sprite1 Duck

Choose a Sprite

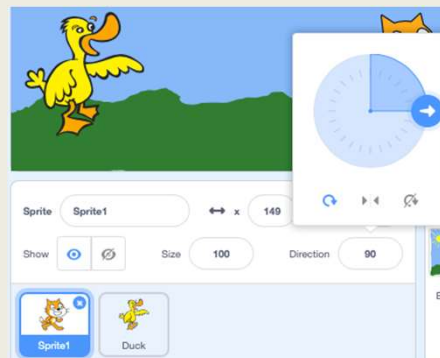
Sprites

- ▶ Each Sprite will contain the **costumes**, **properties**, **scripts** (code), and **sounds** specific to that sprite.
 - ▶ This means that if you delete a sprite, you also delete any code you have written for it!!!
- ▶ Sprites can be cloned either programmatically, or using the Scratch interface.
 - ▶ Each clone initially inherits all the costumes, properties scripts and sounds of the original.
 - ▶ After cloning each sprite can be changed individually
- ▶ Although there is no theoretical limit on the number of Sprites you can have in your program, there is an overall file size limit of 50mb, and a limit of 300 clones.



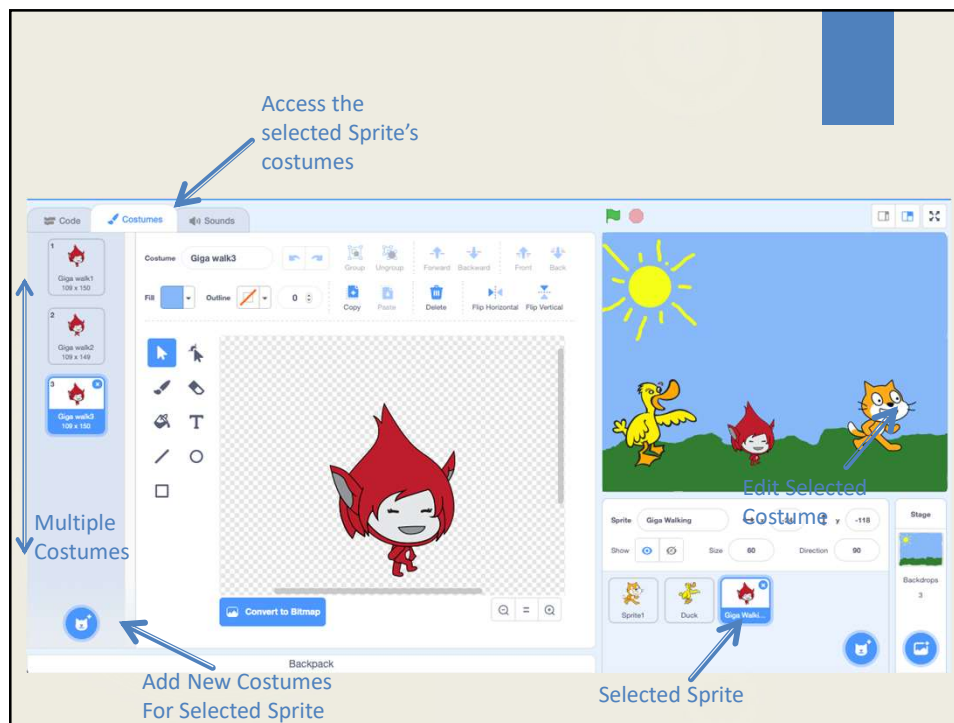
Sprite Properties

- ▶ Each Sprite has the following properties :
 - ▶ Name – Unique for each sprite, **should be meaningful**
 - ▶ Current position & direction
 - ▶ Rotation style
 - ▶ full rotation
 - ▶ left/right rotation
 - ▶ no visible rotation
 - ▶ Visibility (Show, don't show)
- ▶ All these properties except name can also be modified programmatically



Costumes

- ▶ The appearance of a sprite can be customized using costumes, which are actually 2-D images.
- ▶ Since the visible representation of a sprite is an image, on its own it's impossible to do something like program a sprite to move its legs
- ▶ A program often uses multiple costumes for the same sprite to give the appearance of movement (Like a flipbook)
- ▶ A Sprites position on the grid is calculated from the costume's **center**
 - ▶ Scratch 3.0 doesn't currently have a centering tool, so when creating your own sprites it is necessary to pay attention to where their located within the sprite window



Good Sprite Images

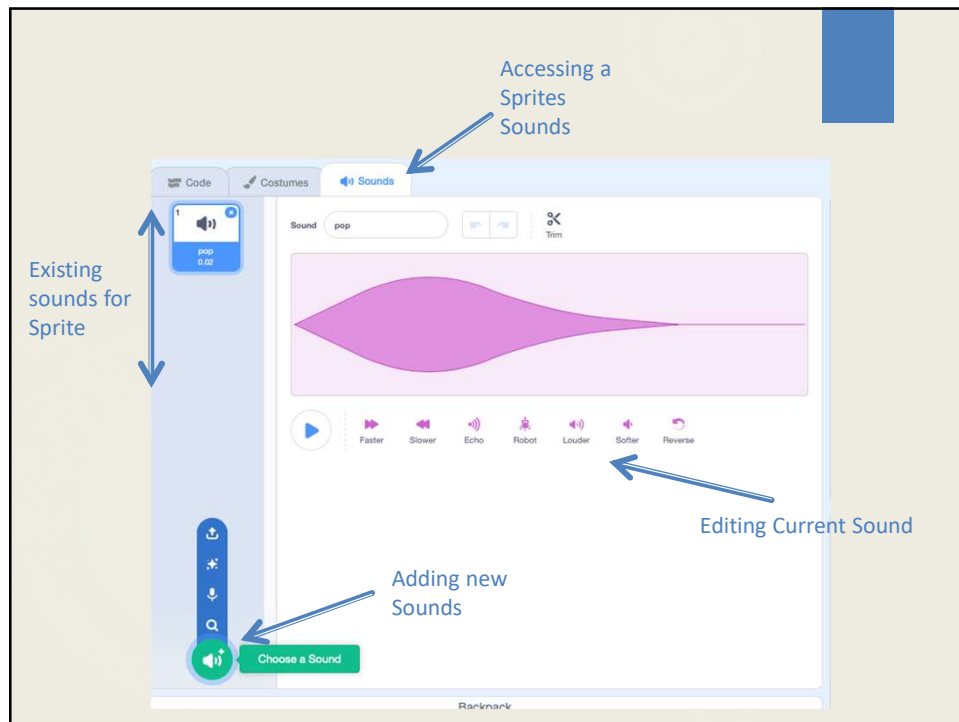
- ▶ The best images to use for Sprites are ones with transparent backgrounds (.png or .gif)
- ▶ **Sprite Sheets** will often provide multiple costumes for the same sprite, in different positions



<http://sindorman.deviantart.com/art/Harry-potter-charset-3-130643533>

Sounds

- ▶ There are several programming blocks (commands) in Scratch for playing sounds.
- ▶ Each of these blocks only works with sounds that have been added to the Sprite executing it.
- ▶ Sounds can be added similarly to backdrops and costumes:
 - ▶ From the Library
 - ▶ Uploaded from the Computer
 - ▶ Recorded new from the Mic.
 - ▶ Randomly

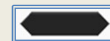
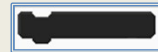


Blocks

- ▶ **Blocks** are puzzle-piece shapes that are used to create code in Scratch.
- ▶ The blocks connect to each other like a jigsaw puzzle
- ▶ Each data type (event, command, reported value, reported boolean, or script end) has its own shape and a specially shaped slot for it to be inserted into
 - ▶ This prevents syntax errors, since pieces will only fit where they are allowed to go.
- ▶ Series of connected blocks are called **scripts**.

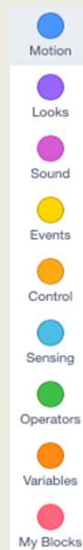
Block Shapes

- ▶ The shape of a block indicates where it should be used
 - ▶ Hat Blocks - Used to begin scripts
 - ▶ Blocks can only be placed below
 - ▶ Stack Blocks – Main commands
 - ▶ Blocks can go above and below
 - ▶ Boolean Blocks – Conditions (either true or false)
 - ▶ Reporter Blocks – Values
 - ▶ Can hold numbers and strings
 - ▶ C Blocks – wrap the stack blocks they contain either for repetition or conditional execution
 - ▶ Cap Blocks – End scripts
 - ▶ Blocks cannot go below them



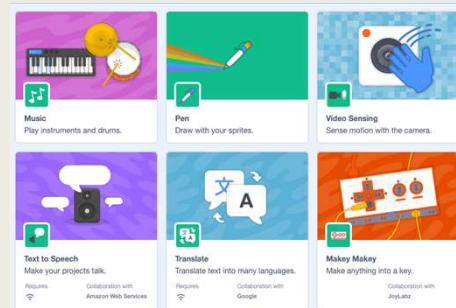
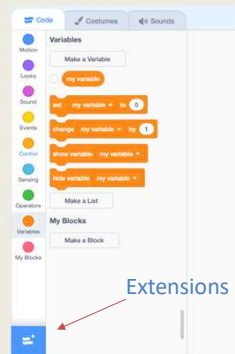
Block Colours

- ▶ Blocks are also colour coded based on their purpose
 - ▶ **Motion** – Moving/Positioning
 - ▶ **Looks** – Talking/Thinking/Costumes/Backdrops/Size/Visibility
 - ▶ **Sound** – Sound
 - ▶ **Variables**– Creating/modifying **Variables** and/or **Lists**
 - ▶ **Events** – control events and triggering of scripts
 - ▶ **Control** – control the execution of other blocks (Wait, Repetition, Condition)
 - ▶ **Sensing** – Detect state of environment or program
 - ▶ **Operators** – Boolean Operators, Math Operators, String Handling
 - ▶ **My Blocks** – You can make your own blocks (code re-use)



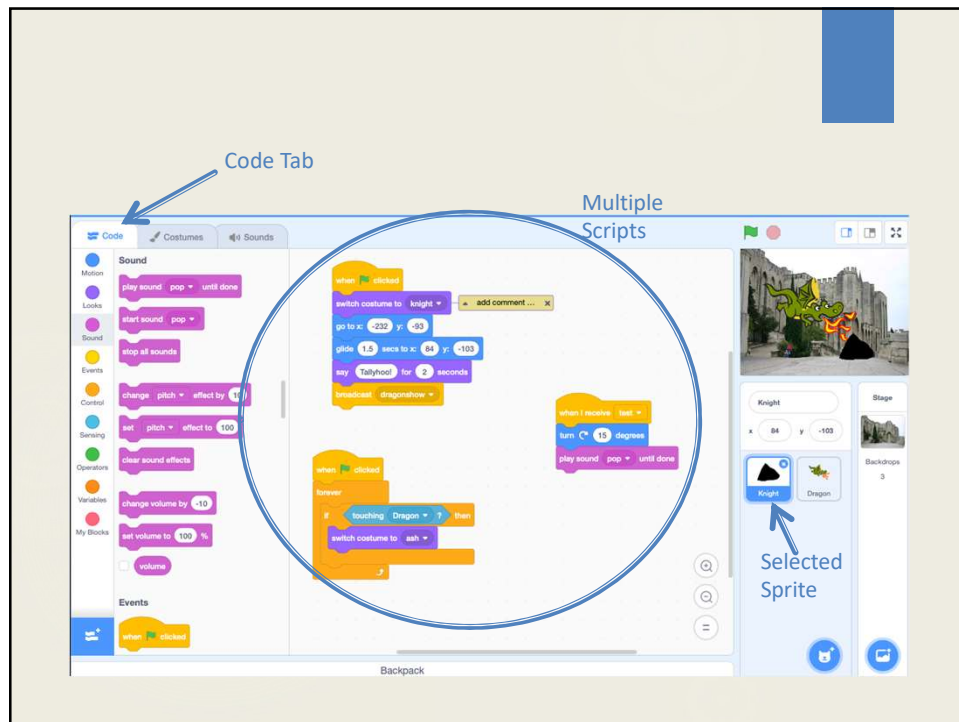
Extensions

- Besides the base blocks, Scratch 3.0 also includes extensions, which allow you to add more blocks for a particular purpose (ie. making music, drawing, translating, working with Bluetooth devices)



Scripts

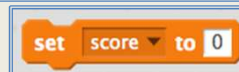
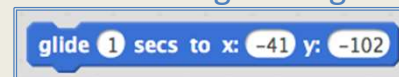
- Scripts (and their contained blocks) are added to the **Code Tab** for a particular sprite (or the stage)
- A script's instructions are applied to/ executed by the sprite that contains it
 - Scripts added to the stage don't have an object to apply the commands to, so there are several blocks that cannot be placed in a script for the stage
- The order and positioning of individual scripts within the scripts tab is meaningless



Arguments

- ▶ Many of the code blocks in scratch have arguments
- ▶ An argument is an area in a block which accepts programmer input or another block
- ▶ Arguments in scratch can be one of several types, which depend on the block containing the argument

- ▶ Numeric insert
- ▶ String insert
- ▶ Built-in drop down list
- ▶ Boolean Block insert
- ▶ Built-in colour selector



Events

- ▶ Scratch is an Event Driven programming language
- ▶ Each script must begin with an event which tells the program when it should execute
 - ▶ When the green Flag is clicked (program start)
 - ▶ When a specific key is pressed
 - ▶ When the sprite is clicked
 - ▶ When the backdrop switches to a specific backdrop
 - ▶ When a specific message is received
 - ▶ When the sprite starts as a Clone
 - ▶ Note: this is a control block rather than an event, but still works to start a script
- ▶ It is possible (and common) to have multiple scripts with the same event
 - ▶ This allows things to happen simultaneously

Stranded Code

- ▶ Any code blocks in your scripts tab that aren't attached to a Hat Block (event), or another block that is attached to a Hat Block will never get executed.
- ▶ Make sure not to leave code stranded



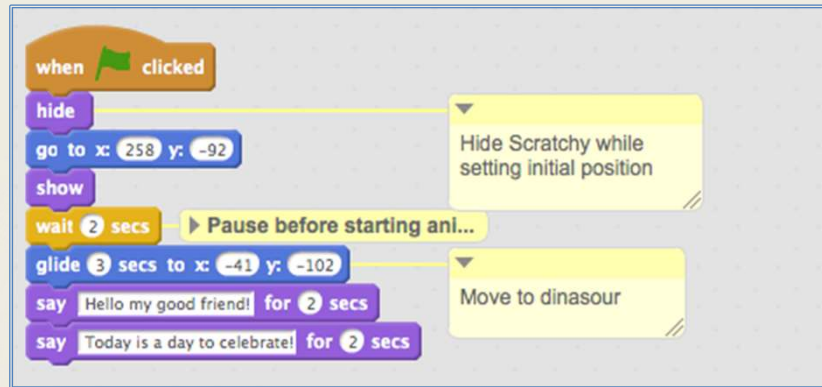
Starting Positions

- ▶ It's important to note that any changes you make to your sprite during the program (size, position, costume, etc..), do not revert back once the program is complete
- ▶ The next time you run the program, everything will begin where the last run ended
- ▶ Because of this, the first thing you should code for all your sprites is their starting **state**
- ▶ This will almost always be a script that runs when the green flag is clicked – since it should happen right at the beginning

Comments

- ▶ Comments allow a programmer to put descriptive text in their program that is ignored during execution
- ▶ They provide other people looking at your program a better understanding of what your code is doing
- ▶ They can also be useful when you return to code after a long time away to remind yourself what you did 😊
- ▶ Comments are added in scratch by right click in the scripts tab
- ▶ They can be adjusted horizontally and vertically
- ▶ They can float free, or be attached to a particular code block

Comments



Repetition

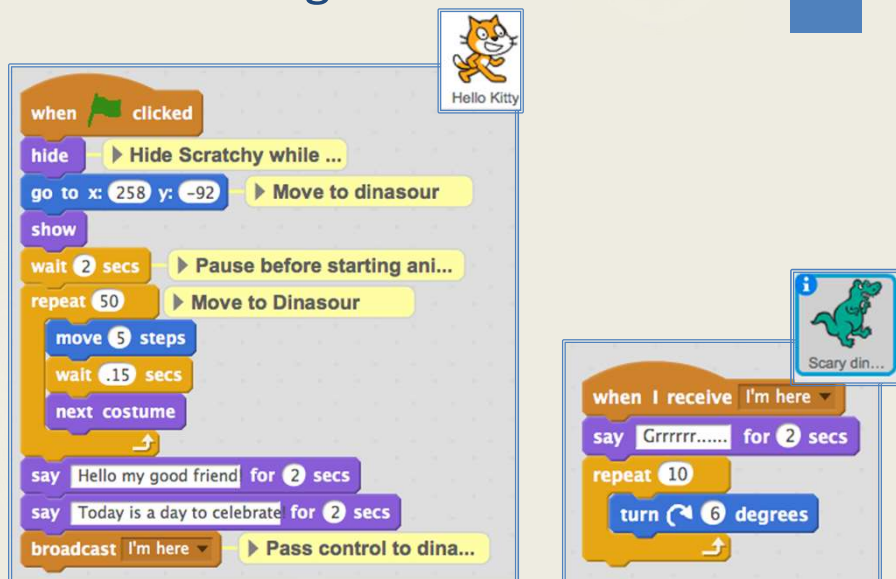
- ▶ In programming, repetition control structures allow the programmer to repeat a section of code without having to put it into the program multiple times
- ▶ Scratch contains 3 types of repetition structures
 - ▶ Numeric – repeat a specific number of times
 - ▶ Conditional – repeat until a specific condition is true
 - ▶ Infinite – repeat forever



Broadcasting

- ▶ Often in an animation/game it is necessary for one sprite to trigger another sprite's behavior
- ▶ In Scratch this is accomplished using **broadcasting**
- ▶ The original sprite **broadcasts** a specific message, and all sprites with a **When I receive** block for that message begin whatever code blocks are attached
 - ▶ The contents of the message don't have any effect on what happens, but for code readability it is always a good idea to choose a meaningful message
 - ▶ If multiple sprites receive the message, they all begin their tasks simultaneously

Broadcasting



The image shows two Scratch scripts illustrating broadcasting. The first script, for a sprite named 'Hello Kitty', starts with a 'when green flag clicked' event. It then performs a series of actions: 'hide' (with a comment 'Hide Scratchy while ...'), 'go to x: 258 y: -92' (commented 'Move to dinosaur'), 'show', 'wait 2 secs' (commented 'Pause before starting ani...'), and a 'repeat' loop of 50 times containing 'move 5 steps', 'wait .15 secs', and 'next costume'. After the loop, it says 'Hello my good friend' for 2 seconds and 'Today is a day to celebrate' for 2 seconds. Finally, it broadcasts the message 'I'm here' with a comment 'Pass control to dina...'. The second script, for a sprite named 'Scary din...', is triggered by 'when I receive I'm here'. It says 'Grrrrrr.....' for 2 seconds and then enters a 'repeat' loop of 10 times, each iteration turning 6 degrees.

```

when green flag clicked
  hide
  go to x: 258 y: -92
  show
  wait 2 secs
  repeat (50)
    move 5 steps
    wait .15 secs
    next costume
  say Hello my good friend for 2 secs
  say Today is a day to celebrate for 2 secs
  broadcast I'm here

when I receive I'm here
  say Grrrrrr..... for 2 secs
  repeat (10)
    turn 6 degrees
  
```

Sensing

- ▶ Sensing blocks in scratch are used to have to program determine different factors within the program.
- ▶ These include things such as:
 - ▶ Position of the mouse
 - ▶ Input from the user
 - ▶ If sprites are touching (each other, or a particular colour)
 - ▶ Keys pressed
 - ▶ System date/time

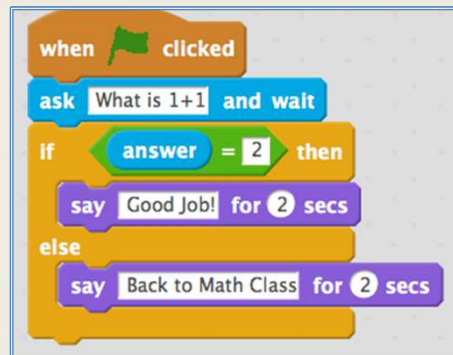
Conditional

- ▶ Conditional statements ask questions about the program state to choose from a set of different sequences of commands.
- ▶ Conditional statements have slots that are shaped with points on either side which evaluate to a true or a false value and execute if the statement is true
- ▶ The if...then, else block allows for one set of statements to execute if the condition is true, and another set to execute if the condition is false



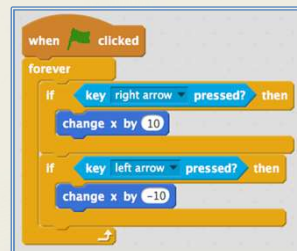
Getting Data From the User

- Sometimes as a programmer, you wish to get and use data from the user
- Scratch provides a block for asking the user a question, and a method to receive their answer



Other User Interactions

- Scratch also contains methods to code based on other user interactions
 - When a sprite is clicked
 - When a key is pressed
- For repetitive key presses (e.g. moving with the arrow keys) there is a slightly more efficient way.



Variables

- ▶ Scratch contains two methods for storing information during the running of your program
 - ▶ Variables
 - ▶ A single piece of information stored in memory
 - ▶ Lists
 - ▶ Multiple variables stored together (think high score board)
- ▶ Variables can be displayed for the user (like a score), or hidden and used for the internal workings of your program (like a calculated velocity when a character is falling)

Variables

