Introduction to Scratch:
Creative Coding

STEELS Standards

- 3.5.6-8.G
- 3.5.6-8.K
- 3.5.6-8.P
- 3.5.6-8.Q
- 3.5.6-8.S
- 3.5.6-8.U
- 3.5.6-8.V
- 3.5.6-8.X
- 3.5.6-8.GG

Objectives

- Students will understand how to navigate the Scratch interface
- Students will understand how to chain motion blocks together to create a simple animation
- Students will understand basic control structures, conditionals, and loops

Materials

- Scratch Website
- A personal computer

Basic Vocab

- **Sprite**
  - The objects on the Scratch stage that perform actions
- **Stage**
  - Where the project is displayed when active
- **Blocks**
  - Programming commands that you snap together to create a program in Scratch
- **Conditional**
  - Expressions that evaluate to either true or false and allow computers to make decisions based on a specific criteria

Introduction

Introduce Scratch as a means to create interactive stories, games, and animations. It was first launched in 2007 by the Lifelong Kindergarten Group at the MIT Media Lab and was developed with the intention of providing an intuitive platform allowing for creative expression. As one of the first kid-friendly open source resources of its time, it drew in users, teachers and students alike, and fostered a community of young minds seeking to create, with some going as far as creating extensions that improve user experience.

Although lacking the finesse of professional cartoons and animations, Scratch covers a variety of actions, environments, and stories, only limited by creativity. To demonstrate the potential of Scratch, you can display some finished projects located here.

To navigate to your own project, click create, located near the top next to the Scratch logo

Basic Blocks

Start with the motion blocks ⇒ As the name suggests, they all make the sprite perform a motion. To utilize a block, drag it from its side panel to the center. Clicking on it will make the sprite move as specified. To delete a block, simply click on it so that it is highlighted, and hit delete.

Go through a few of the motion blocks with the class and observe what happens each time and encourage them to play around with all the possible options so they know what they have at their disposal. Values can be altered as well to perform more specific actions. For example, instead of moving the default ten steps, you can have your sprite take a larger step of 100.

Try some negative values too and draw connections to how the stage is essentially a large coordinate grid

As students grow more comfortable with the interface, they can independently explore the Looks and Sounds blocks as well.

Chaining Blocks

Combining actions allow for multiple actions to happen simultaneously. By snapping two blocks together to create a combined block. Try adding a “Say Hello” block under a “Move 10 Steps” block so that they snap together. Clicking the large block will trigger both actions at once!

Events

Event blocks focus on specifying when an action occurs. Essentially think of it as When x Do y. By adding an Event block to your Action block, you can induce the action at will. Notice that all the Event blocks are header blocks since they are the highest level of control. Try adding a “When Flag clicked” block followed by some basic blocks. Clicking the green flag at the top will run the block. Remove it and the green flag has no effect anymore!

Control

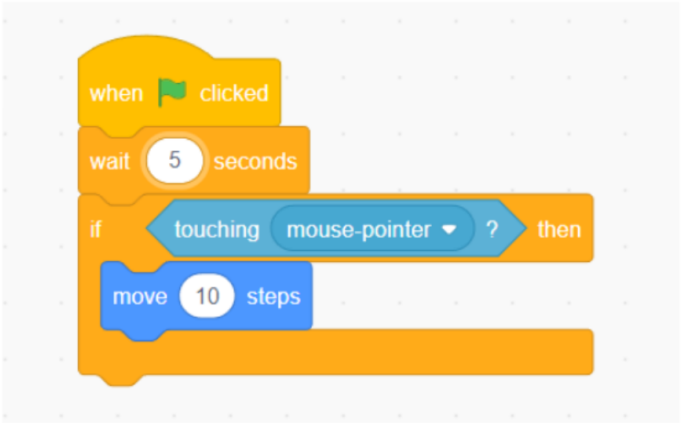
Control blocks help control what occurs on stage. It dictates when and how many times certain actions should be performed. Observe the three main types: loops, ifs, and commands.

Control

Control blocks help control what occurs on stage. It dictates when and how many times certain actions should be performed. Observe the three main types: loops, ifs, and commands.

Loops “loop” through the block UNTIL the end condition is met. Try the “Repeat 10” block. Any action placed inside it will run ten times! In this case, the block is looped through ten times. What other Control blocks are loops? (“Forever” and “Repeat until”)

If blocks only run IF a certain condition is met. Note that the two if blocks take in a diamond shaped condition. Go through all the different types of blocks and take note of which ones would fit. To demo you can build a block stack like the one pictured below. Discuss why you might need the wait in between. (The blocks run sequentially without the wait, once the flag is clicked, the sprite checks if it senses that it is touching the mouse. Obviously he isn’t since the mouse is on the flag and so the if block will never be true)



In this case, the sprite will only move 10 steps IF it is touching the mouse pointer.

The Command blocks are the rest of them. They instruct simple commands such as create a clone or delete a clone.

Fun Bonuses

Access backdrop on the bottom right to change the scene
Choose a sprite on the bottom right to choose from a wide range of sprites and customize its size and direction

Class Activity

Option 1:

Give the class some time to independently explore Scratch and create their own animation. Perhaps help brainstorm ideas related to concepts in their other lessons or ones that teach helpful lessons. For example, an environmental awareness one based in the arctic featuring a polar bear!

Option 2:

Have the whole class collaborate to create one animation! Take ideas and opinions from the students and have everyone involved in the creating and debugging process.

Option 3:

Instruct students to program their sprite to do a variety of tasks. Here it is very important to study how different students may have taken different approaches to come to the same desired output. Analyze the differences and the pros and cons of each approach.

- Have the sprites throw a dance party complete with music and a variety of dance moves
- Create a weather forecast for the week (weather should change day to day or depending on user inputs)
- Create a virtual pet that changes outfits, sounds, and position based on mood, user input, etc.

Summary

Bring the class back together and have them share their animations(as applicable based on classroom activity). Discuss the strategies they employed and the challenges they faced during the activity.

Discuss how concepts (problem solving, sequencing, loops, conditionals, and event handling) introduced in Scratch mirror many engineering and coding principles. Encourage students to think about where else in their lives they may have been unconsciously utilizing these concepts. What other industries do they think these concepts are beneficial?

Emphasize the importance of creativity in Computer Science/Engineering. Coding requires more flexibility than one might think and knowing how to adapt and attack a problem from different angles will make you a successful programmer. In addition, much like with your animation, your computer science projects are only capped by your imagination!

In summary, Scratch acts as an excellent stepping stone into the world of engineering, primarily computer science. It introduces fundamental programming concepts in a visual environment and these concepts act as building blocks to understand more advanced programming languages and techniques! In addition, Scratch promotes creativity and experimentation which allows young programmers to build their problem solving skills.

Discussion

(Try to guide student discussion to touch on these)

- Computer Science concepts are all around us. At its core, computer science breaks down large tasks into small manageable pieces, something we already naturally do.
  - Sequential list of tasks to get ready in the morning
  - Conditionals when making decisions based on a varying factor
  - Looping through the same classes every day until the end of the year
  - For more professional applications
    - Design phases of engineers. Engineers iterate through multiple design phases optimizing and refining each time until the desired performance is met
    - Sensor feedback use conditionals to control robotics
    - Events in Civil engineering such as earthquakes and floods trigger other actions
- Future Applications
  - Software Engineering
    - The concepts taught in Scratch provide a strong foundation for students looking to pursue a career in computer science
  - Game Development
    - Scratch also introduces game design at its most basic level. Refining the crude animation of Scratch to the sophistication of games seen on screens simply requires an

- Software Engineering
  - The concepts taught in Scratch provide a strong foundation for students looking to pursue a career in computer science
- Game Development
  - Scratch also introduces game design at its most basic level. Refining the crude animation of Scratch to the sophistication of games seen on screens simply requires an expansion of these concepts and higher level services and resources
- Education
  - Scratch projects are a fun and informative way to interact with others and perhaps teach something
- Digital Art and Media
  - Those interested in digital art and media may pursue careers in graphic design, animation, or digital marketing, leveraging their programming skills to create engaging visual content for various platforms