Title: “SCRATCHing the Surface”	Targeted Grade: 6 - 12 Lexile: 610L - 800L
Author(s):	Time Expectancy: 60-90 minutes Depth of Knowledge (DOK 1, 2, or 3): 3
Computer Science Learning Objectives: Student will: ● create their own Scratch accounts with unique usernames and passwords, demonstrating digital responsibility and ownership. ● identify and categorize different block types in Scratch's user interface, such as motion, looks, sound, and events, by correctly labeling them based on their colors. ● proficiently utilize the "Events," "Motion," "Looks," "Sound," and other relevant blocks to craft a functional program that showcases their creativity and understanding of coding concepts.	
Concepts/Keywords: Scratch, blocks, program, syntax, sequence, command, coding	
K-12 CSTA Identifier(s)	**Standard(s)** *and Descriptive Statement(s)*
2-AP-10	**-Use flowcharts and/or pseudocode to address complex problems as algorithms.** (Subconcept: Algorithms; Practice: 4.4, 4.1)
2-AP-11	**-Create clearly named variables that represent different data types and perform operations on their values.** (Subconcept: Variables; Practice: 5.1, 5.2)
1A-IC-18	**-Keep login information private, and log off of devices appropriately.** (Subconcepts: Safety Law & Ethics; Practice: 7.3)
K-2 Computer Framework(s)	Practice # and Statement(s)
P4. Developing and Using Abstractions	1. **Extract common features** from a set of interrelated processes or complex phenomena.
P5. Creating Computational Artifacts	2. **Create a computational artifact** for practical intent, personal expression, or to address a societal issue.
ISTE Standards	Standard(s)/Statement(s)
2. Digital Citizen: Students recognize the rights, responsibilities and opportunities of living, learning and working in an interconnected digital world, and they act and model in ways that are safe, legal and ethical.	2d. Students manage their personal data to maintain digital privacy and security and are aware of data-collection technology used to track their navigation online.
4. Innovative Designer: Students use a variety of technologies within a design	4c. Students develop, test and refine prototypes as part of a cyclical design process.

process to identify and solve problems (continued from previous page) by creating new, useful or imaginative solutions. 5. Computational Thinker: Students develop and employ strategies for understanding and solving problems in ways that leverage the power of technological methods to develop and test solutions.	4d. Students exhibit a tolerance for ambiguity, perseverance and the capacity to work with open-ended problems. 5d. Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.
Additional Content Standard #(s)	Standard(s)/Statement(s)
NGSS: MS-ETS1-2 CCSS-ELA: RI.6.7 Integration of Knowledge and Idea	Evaluate competing design solutions using a systematic process to determine how well they meet the criteria and constraints of the problem. **SEP**:Developing Models **DCI**: ETS1.B: Developing Possible Solutions & ETS1.C: Optimizing the Design Solution Integrate information presented in different media or formats (e.g., visually, quantitatively) as well as in words to develop a coherent understanding of a topic or issue.
State (or International) Standard(s): (TBD and identified by location of instructor utilizing lesson).	
References	K-12 CSTA Standards: Computer Science Teachers Association (2017). *CSTA K–12 Computer Science Standards, Revised 2017.* Retrieved from https://csteachers.org/k12standards/. K-12 Computer Science Framework: https://k12cs.org/wp-content/uploads/2016/09/K%E2%80%9312-Computer-Science-Framework.pdf Next Generation Science Standards: https://www.nextgenscience.org/standards/standards Common Core State Standards for ELA: http://www.thecorestandards.org/ELA-Literacy/ ISTE Standards: https://www.iste.org/standards/for-students Bloom's Digital Taxonomy Verbs: https://libguides.bc.edu/c.php?g=628962&p=4506921 Scratch: https://scratch.mit.edu/
Lesson Resources/Folder Access (Link)	

Overview: The overall purpose of "SCRATCHing the Surface" is to introduce students to computer programming through Scratch, a block-based programming language. This lesson aims to provide students with hands-on experience in creating a Scratch account, identifying different types of blocks, and exploring tutorials to develop their foundational coding skills. By the end of the lesson, students will gain familiarity with Scratch's user interface, understand the significance of block types, and apply their knowledge to complete basic coding projects.

Part I will guide students through the process of creating their own Scratch accounts, where they will choose unique usernames and passwords. This will introduce students to internet safety and responsibility, which is paramount in professional and personal computer science.

In Part II, students will be introduced to the various block types that serve as the building blocks of coding. The instructor can facilitate the exploration of the Scratch interface, guiding students to identify and categorize blocks based on their distinct colors. Students will gain an understanding of how each block type contributes to the overall structure of code.

In Part III, students will progress to a series of Scratch tutorials where they will utilize their knowledge of block types to assemble sequences of blocks into various programs. Each student should complete one tutorial from each category of “Animations”, “Art”, “Music”, “Games” and “Stories” for a total of five. Students will showcase their achievements by providing screenshots of their work.

In the Extend Your Thinking reading about block-code, students will learn more about block-based coding, which is syntax-free and a visual programming language. This reading, while focused on block-based coding’s capabilities, will also introduce students to other forms of programming that may be ahead.

Ultimately, the "SCRATCHing the Surface" lesson aims to cultivate an environment where students can develop a foundational understanding of programming concepts while nurturing their creativity and computational thinking skills. This should empower students to become confident digital creators and problem solvers.

Rationale/Background: The purpose of this lesson is to provide students with hands-on experience in a block-based programming environment, Scratch. Scratch is an engaging and accessible platform for students. This lesson leverages students' background knowledge and interests, such as art and games, to introduce them to foundational programming concepts using a visual and interactive approach.
Prior to this lesson, students should be able to fluently navigate in a browser and understand how to explore a common website. Students with learning needs should be ideally identified prior to the lesson to best assist them with the completion of the lesson. This lesson is intended to be an introduction to subsequent lessons, but they will be able to express personal interests by choosing which tutorials they would like to complete.

Teacher (Required) Materials/Resources: Hard copy or digital copy of this lesson for the instructor and each student (if teaching this lesson in person), Internet access, a means of meeting virtually (either by Google Meet, Zoom [although Zoom is not preferred due to threat of hacking and inappropriate disruptions], or FaceTime (depending on the size of the intended class).

“Scratching the Surface Appendix 1” Internet-Free Supplement: This document is provided in the event that any student does not have access to the Internet in their home environment and the “Extend Your Thinking” portion is assigned to be completed as homework.

Student Materials:

1. An iPad, laptop or desktop computer.
2. Internet access.
3. Either hard copy or digital copy (that will allow writing on [such as Notability]) of “Student Sheet: SCRATCHing the Surface”
4. Scratch accounts created during the lesson.
    a. Requires an email account to create. Students may use their own or a parent/guardian’s.
5. Ability to screenshot or photograph their assembled block sequences for each tutorial category.
    a. Encourage students to look up how to screenshot on their own devices, and assist when necessary.
    b. Example: Use the Google search engine to search “how to screenshot on Windows 10”
6. For “Extend Your Thinking: Blocks Build Brilliance”: Either hard copy or digital copy of reading material. Students may follow directions to the link provided on their student sheet, or print a hardcopy from “Scratching the Surface Appendix 1.”

Guided Practice/Instructor Procedures:

A) Introduction and Motivation:

1. Assess prior knowledge by asking students about their familiarity with Scratch or other coding experiences.
2. Motivate students by explaining the creative possibilities and real-world applications of programming skills.
3. Activate prior knowledge by discussing any coding concepts students have encountered previously.
4. State the lesson's purpose: To introduce students to Scratch, explore its block types, and complete tutorials to develop basic programming skills.

B) Lesson Body:

Part I: Creating Your Scratch Account

1. Guide students through the steps to create a Scratch account or log in if they already have one.
2. Emphasize the importance of keeping usernames and passwords secure and not using real names.
3. Ensure students write down their username and password in a safe place.

Part II: Identifying the Blocks

1. Instruct students to access Scratch and choose the "Create" option.
2. Guide students to identify block types based on their colors and complete the table on their student sheets.
3. Clarify any confusion about block colors by offering assistance.
4. Explain the meaning and function of the three key symbols used in assembling projects.

5. Explain each block is a “command”, a collection of commands form “sequences”, and all related sequences form a “program.”

Part III: Learning the Moves

1. Direct students to the Scratch "Ideas" section and select the "Choose a tutorial" option.
2. Instruct students to watch the "Getting Started" tutorial video and answer the related questions.
3. Instruct students to explore the different tutorial categories, complete one tutorial from each category, and take screenshots or photographs of their assembled blocks.
4. Encourage higher-level learners to experiment with combining multiple tutorials to create more complex projects.

C) Lesson Closure:

1. Ask students to share their experiences and any challenges they faced during the lesson.
    a. “Did anyone have a favorite tutorial?”
    b. “Did anyone learn something that surprised you?”
2. Ask the students to summarize the key concepts covered in the lesson such as blocks, commands, sequences, and program.
    a. “How would you define a "block" in the context of Scratch?”: Blocks are the building blocks of programming. They represent individual commands or instructions.
    b. “What's the significance of a "sequence" of blocks?”: A sequence is like a series of steps that the computer follows in order. It's the order in which we place the blocks that determines what the computer does.
    c. “And what is a "program"?”: A program is a collection of instructions, or code, that tells the computer what to do. In the previous lesson, writing detailed instructions for how to make a peanut butter and jelly sandwich was simulation writing a program.
3. Extend Your Thinking reading can be completed during the end of class if time allows, or be assigned as homework. Discuss in class if possible.

Student Misconceptions:

- Programming is overly complex and difficult to learn.
- All code is represented by text or syntax is the same in all programming languages.
- Failure in programming is negative or means one is bad at it.
    - Programming is full of trial-and-error to problem solve. Each failure is a step toward success!

Reading Selection: Ask students to type in the following link, https://www.codingal.com/coding-for-kids/coding-guides/block-coding-guide/, also found at the bottom of the lesson’s student sheet, or provide a hard copy of “Scratching the Surface Appendix 1” if Internet is not available.

Assessment:

A) Student assessment (by instructor):

Informal Assessment: The instructor will engage in ongoing discussions with students, observing their interactions with Scratch, their engagement levels, and their comfort in using block-based coding. The instructor will actively address student questions and provide guidance, focusing on clarifying concepts related to block categories, sequencing, and the understanding of key symbols. Additionally, the instructor will monitor students' participation in tutorial exploration, taking note of their willingness to experiment and apply concepts independently.

Formal Assessment: The instructor will formally assess students' understanding of block categories and their ability to create functional programs in Scratch. This assessment will involve a review of the labeled block categories and their associated types identified by students in Part II of the lesson. The instructor will evaluate students' proficiency in assembling a functional program by examining their completed tutorial activities across the various categories. Students' performance in designing block sequences to accomplish specific tasks and their utilization of key symbols will be used to gauge mastery of the concepts taught in the lesson.

B) Instructor Self and Student Evaluation: The instructor is encouraged to complete the following as the lesson is being carried out or reflected after the lesson is completed.

Three Strengths of This Lesson:
1) ____
2) ____
3) ____

Three Elements/Areas for Improvement:
1) ____
2) ____
3) ____

Identification of students (**using initials, not names**) who were not successful in meeting the stated objectives: ____
____

How shortcomings will be addressed prior to starting next session:
____
____
____

Scope and Sequence: Prior Assignment: “Easy as 1, 2, 3”
Next Lesson: “iGrasshopper”

Look-Ahead: In the next lesson “iGrasshopper”, students will again need access to the Internet and their Scratch accounts. Students will design their own Sprite to use while exploring more about how code blocks work, writing sequences, and testing loops.