CREATIVE COMPUTING

Karen Brennan | Christan Balch | Michelle Chung
Harvard Graduate School of Education

TABLE OF CONTENTS

BACKGROUND

Welcome to the Creative Computing Curriculum Guide!

To help you dive into the world of creative computing as quickly as possible, we have assembled answers to eight common questions:

1. What is Creative Computing?
2. What is Scratch?
3. What is this guide?
4. Who is this guide for?
5. What do I need in order to use this guide?
6. What is included in this guide?
7. How should I use this guide?
8. Where did this guide come from?

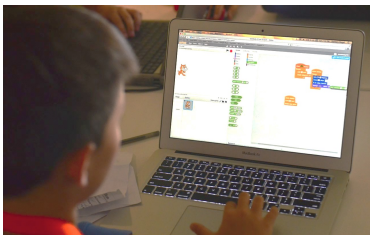
WHAT IS CREATIVE COMPUTING?

Creative computing is about creativity. Computer science and computing-related fields have long been introduced to young people in a way that is disconnected from their interests and values – emphasizing technical detail over creative potential. Creative computing supports the development of personal connections to computing, by drawing upon creativity, imagination, and interests.

Creative computing is about empowerment. Many young people with access to computers participate as consumers, rather than designers or creators. Creative computing emphasizes the knowledge, practices, and fundamental literacies that young people need to create the types of dynamic and interactive computational media that they enjoy in their daily lives.

Creative computing is about computing. Engaging in the creation of computational artifacts prepares young people for more than careers as computer scientists or programmers. It supports young people's development as computational thinkers – individuals who can draw on computational concepts, practices, and perspectives in all aspects of their lives, across disciplines and contexts.

WHAT IS SCRATCH?



There are many different tools that can be used for creative computing. In this guide, we use Scratch, which is a free computer programming language available at http://scratch.mit.edu. With Scratch, people can create a wide variety of interactive media projects – animations, stories, games, and more – and share those projects with others in an online community. Since Scratch's launch in May 2007, hundreds of thousands of people all around the world have created and shared more than 6 million projects.

WHAT IS THIS GUIDE?

This guide is a collection of ideas, strategies, and activities for an introductory creative computing experience using the Scratch programming language. The activities are designed to support familiarity and increasing fluency with computational creativity and computational thinking. In particular, the activities encourage exploration of key computational thinking concepts (sequence, loops, parallelism, events, conditionals, operators, data) and key computational thinking practices (experimenting and iterating, testing and debugging, reusing and remixing, abstracting and modularizing). Learn more about computational thinking – what it is and how to assess its development in learners – from resources in the appendix or by visiting http://scratched.gse.harvard.edu/ct

Inspired by constructionist approaches to learning, the activities in this guide emphasize the following principles:

PRINCIPLE #1: CREATING

Offer opportunities for learners to engage in designing and making, not just listening, observing, and using.

PRINCIPLE #2: PERSONALIZING

Offer opportunities for learners to engage in activities that are personally meaningful and relevant.

PRINCIPLE #3: SHARING

Offer opportunities for learners to engage in interactions with others as audience, coaches, and co-creators.

PRINCIPLE #4: REFLECTING

Offer opportunities for learners to review and rethink their creative practices.

WHO IS THIS GUIDE FOR?

No matter your current context or prior experience, this guide was designed with a wide range of learners and educators in mind. Here are a few examples of who might use the guide and how they might use it:

K-12 TEACHER

Scratch is being used in thousands of elementary, middle-school, and high-school classrooms around the world. The guide can be used in its entirety as a semester-long computing course, or selectively as part of other curricular areas. Many educators introduce creative computing as an after-school or lunch-time program, using the activities as inspiration and scaffolding for students' open-ended explorations.

MUSEUM OR LIBRARY EDUCATOR

In addition to formal learning environments like classrooms, Scratch has been used in informal learning spaces like museums and libraries. Whether as a structured workshop experience or a drop-in play space, these learning environments are ideal for supporting explorations in creative computing, without some of the restrictions present in traditional settings.

PARENT

Parents can use the guide in a wide range of ways. From supporting homeschooling activities, to starting creative computing clubs at school, to hosting workshops at local community centers, parents are encouraged to think about how to use the guide to support the creative computing experiences of young learners.

Creative computing is for everybody!

COLLEGE INSTRUCTOR

Scratch can serve as an introduction to fundamental computational concepts and practices, often followed by a transition to more traditional text-based programming languages in computer science courses. For example, the CS50 course at Harvard University uses Scratch as an introductory programming experience before transitioning to the C programming language. The activities have also been used as part of education, art, and media literacy courses at the college level.

YOUNG LEARNER

Over the past seven years since Scratch's launch, young learners have been passionate advocates for creative computing in a variety of settings. From introducing their parents and teachers to programming, to creating learning opportunities for their peers, creative computing can be something that is done with them or by them, rather than just for them.

WHAT DO I NEED IN ORDER TO USE THIS GUIDE?

In addition to time and an openness to adventure, some important resources include:

+ **Computers with speakers** (and, optionally, microphones and webcams): for the computer-based design activities
+ **Network connection**: for connecting to Scratch online (if your environment does not offer a network connection, a downloadable version of Scratch is available)
+ **Projector or interactive whiteboard with speakers**: for sharing works-in-progress and for demonstrations
+ **Design notebooks** (physical or digital): for documenting, sketching, and brainstorming ideas and plans

WHAT IS INCLUDED IN THIS GUIDE?

This guide is organized in seven units – from an initial preparatory unit to a culminating project-based unit – with each unit typically including six activities. A summary of each unit follows:

UNIT 0 – GETTING STARTED

Prepare for the culture of creative computing by exploring possibilities and setting up technical infrastructure (e.g., creating Scratch accounts, starting design journals) and social infrastructure (e.g., establishing critique groups). Dive into an initial creative experience by making something "surprising" happen to a Scratch character.

UNIT 1 - EXPLORING

Get comfortable with the key computational concept of sequence through a series of activities that provide varying levels of structure – from a step-by-step tutorial, to a creative challenge using a limited number of blocks, to open-ended explorations through making a project about yourself.

UNIT 2 - ANIMATIONS

Play with visuals and audio in these activities focused on animation, art, and music. Explore Scratch's focus on media – and the key computational concepts of loops, events, and parallelism – by building your own band, designing animated creatures, and creating a music video for a favorite song.

UNIT 3 - STORIES

Create new interactive worlds through collaborative storytelling. Begin by developing characters, learning to code conversations, and then situating those characters and conversations in shifting scenes. Combine characters, conversations, and scenes in a larger story project that is passed along to other creators to further develop – and possibly reimagine entirely!

UNIT 4 - GAMES

Connect fundamental game mechanics such as score and levels to key computational concepts, such as variables, operators, and conditionals. Analyze your favorite games, imagine new ones, and practice game design by implementing (and extending) classic games, like Pong.

UNIT 5 – DIVING DEEPER

Before the culminating unit, take a moment to revisit work from prior units, further exploring advanced concepts or helping others by designing new activities or debugging challenges.

UNIT 6 - HACKATHON

Put all of the computational concepts and practices into action by designing and developing a project of your own through iterative cycles of planning, making, and sharing.

Assessment strategies are described throughout the guide, and several assessment instruments are included in the guide appendix. Our approach to assessment is process-oriented, with a focus on creating opportunities for students to talk about their own (and others') creations and creative practices. There are many forms of process-oriented data that could be collected and various strategies are suggested throughout the guide, such as:

+ supporting conversations with and among students about their projects, recorded through audio, video, or text
+ examining portfolios of projects
+ maintaining design journals

We view assessment as something that is done with students, to support their understanding of what they already know and what they still want to learn. Assessment can involve a variety of participants, including the creators, their peers, teachers, parents, and others.

HOW SHOULD I USE THIS GUIDE?

We encourage you to use as much or as little of the guide as you like, to design new activities, and to remix the included activities. No matter your prior experience or expertise, we think of every educator as a co-designer of the Creative Computing experience. We would love to learn about what you're doing, so we encourage you to document and share your experiences with us and with other educators via the ScratchEd community at http://scratched.gse.harvard.edu

WHERE DID THIS GUIDE COME FROM?

This guide was developed by members of the ScratchEd research team at the Harvard Graduate School of Education – Christan Balch, Michelle Chung, and Karen Brennan. Jeff Hawson provided editing support and inexhaustible enthusiasm.

The guide contents draw on a previous version of the Creative Computing Guide (released in 2011) and on the Creative Computing Online Workshop (hosted in 2013). These were made possible with support from the National Science Foundation through grant DRL-1019396, the Google CS4HS program, and the Code-to-Learn Foundation.

We are enormously appreciative of the numerous educators who have used the previous version of this guide and participated in workshops. In particular, we would like to thank the educators who extensively tested the first guide (Russell Clough, Judy Hoffman, Kara Kestner, Alvin Kroon, Melissa Nordmann, and Tyson Spraul) and the educators who extensively reviewed the current guide (Ingrid Gustafson, Megan Haddadi, Keledy Kenkel, Adam Scharfenberger, and LeeAnn Wells).

We are also greatly appreciative of our collaborators. We would like to thank Wendy Martin, Francisco Cervantes, and Bill Tally from Education Development Center's Center for Children & Technology, and Mitch Resnick from the MIT Media Lab for their extensive contributions in developing the computational thinking framework and resources. We would like to thank the many amazing Harvard Graduate School of Education interns who have contributed to the guide development over the past several years since the initial version in 2011, including Vanity Gee, Vanessa Gennarelli, Mylo Lam, Tomoko Matsukawa, Aaron Morris, Matthew Ong, Roshanak Razavi, Mary Jo Madda, Eric Schilling, and Elizabeth Woodbury.

UNIT 0
GETTING STARTED

YOU ARE HERE

0 1 2 3 4 5 6

WHAT'S INCLUDED

UNIT 0 OVERVIEW

THE "BIG IDEA"

When we shared a draft of this guide with teachers, a common initial reaction was, "Unit 0?!? Why 0?"

We hoped to communicate that this is a *preparatory* unit, supporting you in establishing a culture of creative computing through creating, personalizing, sharing, and reflecting. Our ambition to support this type of learning culture will be evident throughout the guide.

Creative computing culture has an intellectual dimension, engaging with a set of computational concepts and practices. It has a physical dimension, encouraging interactions with others through the placement of desks, chairs, and computers. Most importantly, it has an affective dimension, cultivating a sense of confidence and fearlessness.

> *It really helps if you have kind of a culture or climate in your classroom. It starts on the first day – getting kids to appreciate that they're going to make mistakes and that I'm going to be asking them to do stuff that is hard. I always just put that right out there. And they don't, at first, just because they want to succeed. Even adults don't like to fail, or make mistakes. But it is important, I feel, that when you do run into difficulties that it's not time to give up or cry. It's time to think about the strategies that you have to solve your problem, or to look for help. No reason to break down or give up – you keep at it.*
>
> *TS, Elementary School Teacher*

LEARNING OBJECTIVES

Students will:

+ be introduced to the concept of computational creation, in the context of Scratch
+ be able to imagine possibilities for their own Scratch-based computational creation
+ become familiar with resources that support their computational creation
+ prepare for creating Scratch projects by establishing Scratch accounts, exploring Scratch studios, creating design journals, and organizing critique groups

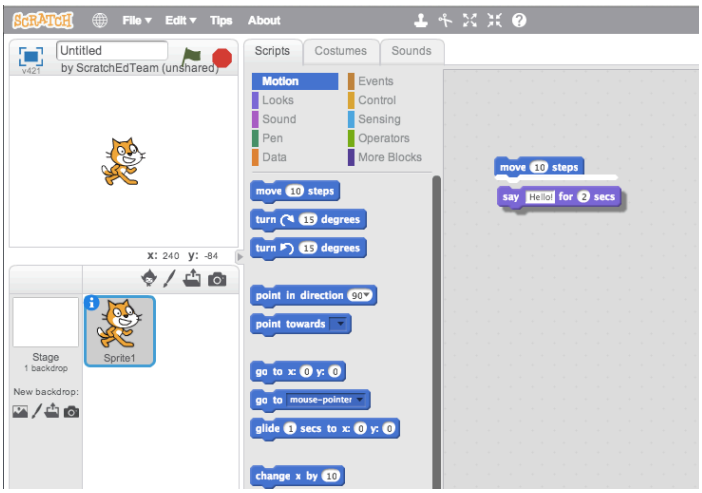
KEY WORDS, CONCEPTS, & PRACTICES

+ profile editor
+ project page
+ studio
+ critique croup
+ red, yellow, green

NOTES

+ Coordinate with your IT department to make sure your computers can access the Scratch website.
+ Don't have internet access? An offline version of Scratch is available for download: http://scratch.mit.edu/scratch2download

CHOOSE YOUR OWN ADVENTURE



Ready to get started? This unit is designed for those who are completely new to Scratch. From exploring inspiring projects, to creating a Scratch account, to having an initial experience playing with the Scratch project editor, each activity is designed to guide you and your students through the process of getting started with Scratch.

In each unit, we offer a selection of activities – but we encourage you to tinker with the choice and order of the activities. Different contexts and audiences will invite different experiences. Choose your own adventure by mixing and matching the activities in ways that are most compelling for you and the learners you support.

Not sure where to start? For more support, check out the suggested path through the activities provided below.

POSSIBLE PATH

SESSION 1

INTRODUCING SCRATCH

Watch the Scratch overview video and imagine what's possible with Scratch.

SCRATCH ACCOUNT

Set up a Scratch account to save and share your projects.

DESIGN JOURNAL

Create a design journal to write down notes and reflections on the process of designing Scratch projects.

SESSION 2

SCRATCH SURPRISE

Can you make the Scratch cat do something surprising?

SCRATCH STUDIO

Learn how to create a studio and add a project to the studio.

CRITIQUE GROUP

Gather in small peer groups to give and receive feedback on ideas and projects drafts.

UNIT 0 ACTIVITY

INTRODUCING SCRATCH

SUGGESTED TIME
5–15 MINUTES

OBJECTIVES

By completing this activity, students will:

+ be introduced to computational creation with the Scratch programming environment by watching the Scratch overview video or exploring sample projects
+ be able to imagine possibilities for their own Scratch-based computational creation

ACTIVITY DESCRIPTION

- ❑ Ask students about their experiences with computers using the reflection prompts to the right.
- ❑ Introduce students to creative computing with Scratch and the range of projects they will be able to create by showing the Scratch overview video and some sample projects that your students will find engaging and inspiring. Explain that over the next several sessions they will be creating their own interactive computational media with Scratch.
- ❑ What will you create? Ask students to imagine what types of projects they want to create with Scratch.

RESOURCES

- ❑ projector for showing Scratch overview video (optional)
- ❑ Scratch overview video
  http://vimeo.com/65583694
  http://youtu.be/-SjuiawRMU4
- ❑ sample projects studio
  http://scratch.mit.edu/studios/137903

REFLECTION PROMPTS

+ What are the different ways you interact with computers?
+ How many of those ways involve being creative with computers?

REVIEWING STUDENT WORK

+ Did students brainstorm a diverse range of project ideas? If not, try showing a wide variety of projects to give students a sense of the possibilities.

NOTES

+ If you don't have internet access, download the Scratch overview video from Vimeo before class, available at http://vimeo.com/65583694
+ Instead of writing out their answers to the reflections prompts, encourage students to get creative by drawing their responses. (e.g., "Draw different ways you interact with computers.")

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....


clicked
10
10
drum 4 for 0.2 beats
Welcome to Scratch!
for 2

UNIT 0 ACTIVITY

SCRATCH ACCOUNT

SUGGESTED TIME
5–15 MINUTES

OBJECTIVES

By completing this activity, students will:

+ create a Scratch account
+ explore the Scratch online community and review the Scratch community guidelines

ACTIVITY DESCRIPTION

- ❑ Scratch online accounts require an email address. If students cannot provide a personal or school email address, a teacher or parent/guardian email address may be used. Plan in advance if permission slips for online accounts need to be collected.
- ❑ Help students navigate to the Scratch website at http://scratch.mit.edu and click on "Join Scratch" to get started creating a Scratch account. Optionally, have the Scratch Account handout available to guide students. Give students time to register, update their Scratch profile page, and explore the Scratch online community. Encourage students to practice signing in and out of their accounts.
- ❑ To make it easier for members of the class to find and follow one another's Scratch profiles, consider creating a class list of usernames and names.
- ❑ Examine the Scratch community guidelines as a group to discuss respectful and constructive behavior. Review how to report inappropriate posts on the website.

RESOURCES

- ❑ Scratch Account handout
- ❑ Scratch community guidelines http://scratch.mit.edu/community_guidelines

REFLECTION PROMPTS

+ What is your Scratch account username?
+ What is a hint to help you remember your password?

REVIEWING STUDENT WORK

+ Were students able to create Scratch accounts and successfully sign in and out of the Scratch website?

NOTES

+ Teachers may prefer providing their email or creating a class email address, as notifications of any inappropriate behavior on the Scratch website will be sent to the email that is registered with the account.
+ Check if any students already have an online account.
+ To remember passwords while maintaining privacy, have students write down their username and password in sealed envelopes that are kept in a secure place in the classroom.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

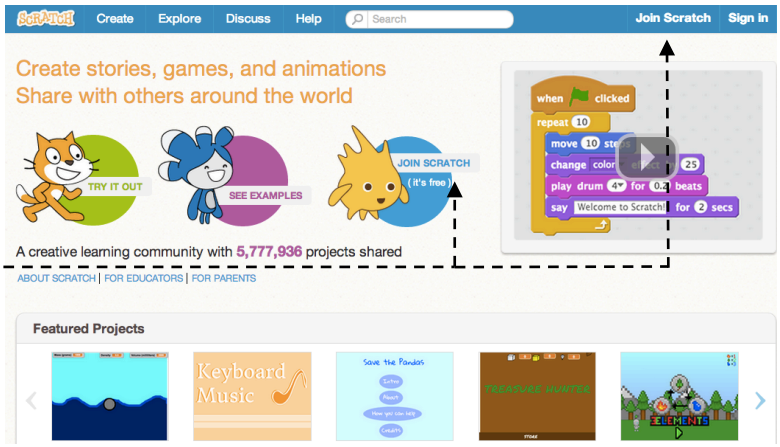
SCRATCH ACCOUNT

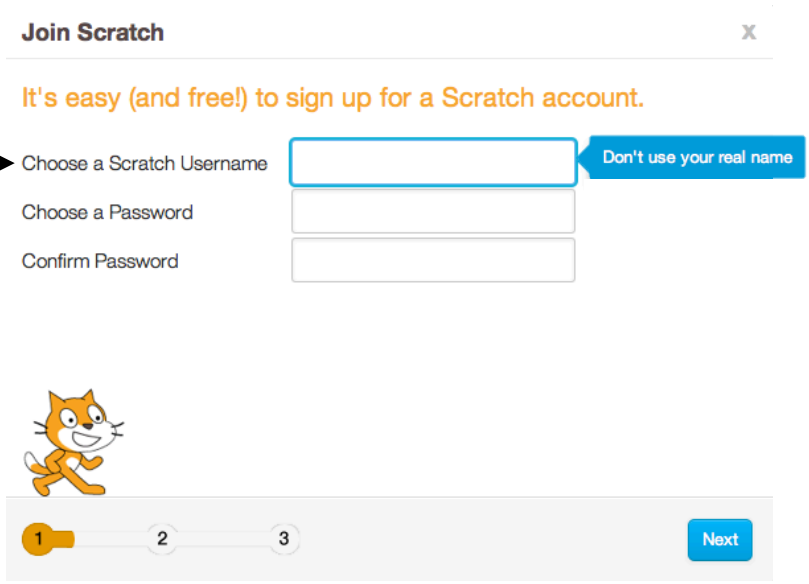
NEW TO SCRATCH? GET STARTED BY CREATING YOUR SCRATCH ACCOUNT!

You will need a Scratch account to create, save, and share your Scratch projects. The steps below will walk you through creating a new account and setting up your profile.

START HERE

- ❑ Open a web browser and navigate to the Scratch website: http://scratch.mit.edu
- ❑ On the homepage, click on "Join Scratch" at the top on the right or in the blue circle.
- ❑ Complete the three steps to sign up for your very own Scratch account!





UNIT 0 ACTIVITY

DESIGN JOURNAL

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ start a personalized design journal for documenting their design process and reflections

ACTIVITY DESCRIPTION

- ❑ Introduce students to the idea of the design journal, a physical or digital notebook where they can brainstorm ideas and share personal reflections, similar to a personal journal or diary. Explain that students will be prompted to update their design journals throughout their Scratch programming adventures, but encourage them to add to their journals anytime during the process of designing projects to capture ideas, inspiration, notes, sketches, questions, frustrations, triumphs, etc.
- ❑ Look through sample design journals to get ideas for what type of design journals (paper or digital) will work best for your students. Give students time to start and personalize their design journals.
- ❑ Ask students to create their first design journal post by responding to the reflection prompts on the right.
- ❑ Encourage students to share their design journals and initial reflections with a neighbor.

RESOURCES

- ❑ sample design journals
  http://bit.ly/designjournal-paper
  http://bit.ly/designjournal-digital
  http://bit.ly/designjournal-blog
- ❑ paper and craft materials (for paper journals)

REFLECTION PROMPTS

+ How would you describe Scratch to a friend?
+ Write or sketch ideas for three different Scratch projects you are interested in creating.

REVIEWING STUDENT WORK

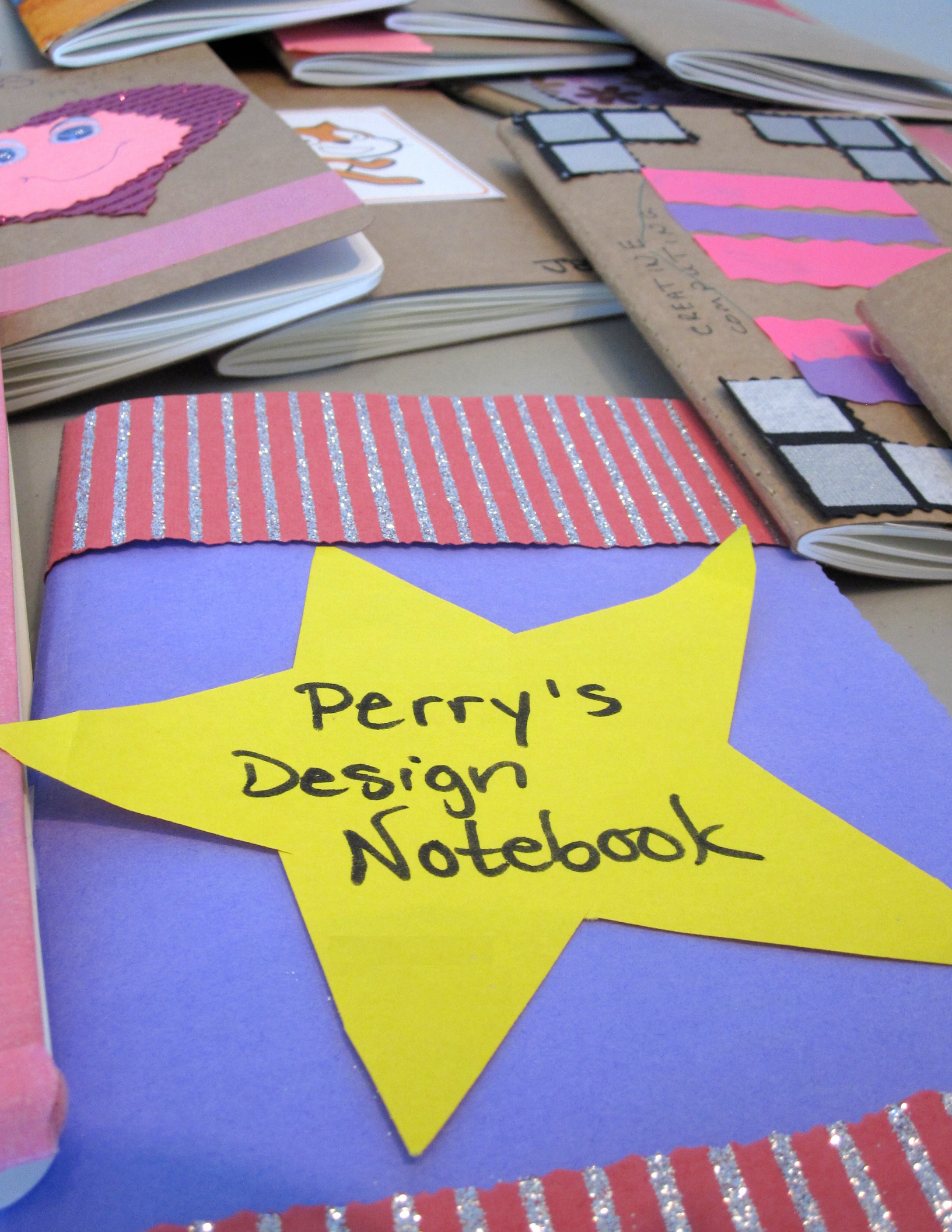
+ What do the reflection responses tell you about the types of projects students might be interested in pursuing?
+ Based on students' responses, which units in this guide might appeal to your different students?

NOTES

+ During other guide activities, facilitate group discussions around relevant reflection prompts.
+ Decide whether design journals should be private or public. For example, you could maintain one-on-one feedback with students through private journals or have students leave comments for peers on shared journals. Consider the pros and cons of each option.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....


Perry's
Design
Notebook

UNIT 0 ACTIVITY

SCRATCH SURPRISE

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ engage in an exploratory, hands-on experience with Scratch

ACTIVITY DESCRIPTION

- ❑ Help students open the Scratch project editor by navigating to the Scratch website at http://scratch.mit.edu, signing in to their Scratch accounts, and then clicking on "Create" at the top of the page. Optionally, have the Scratch Surprise handout and Scratch Cards available to guide students during their explorations.
- ❑ Give students 10 minutes to explore the Scratch interface in an open-ended way. Prompt students with, "You have 10 minutes to make something surprising happen to the Scratch cat." Or, "Take 10 minutes to explore the interface fearlessly. What do you notice?" Encourage students to work together, ask each other for help, and share what they are figuring out.
- ❑ Ask for 3 or 4 volunteers to share with the entire group one thing that they discovered. Optionally, after the volunteers have shared, offer several challenges to the students:
  - Did anyone figure out how to add sound?
  - Did anyone figure out how to change the background?
  - Did anyone figure out how to get help with blocks?

RESOURCES

- ❑ Scratch Surprise Handout
- ❑ Scratch Cards
  http://scratch.mit.edu/help/cards

REFLECTION PROMPTS

+ What did you figure out?
+ What do you want to know more about?

REVIEWING STUDENT WORK

+ Do students know how to initiate a new project?
+ Do students understand the basic mechanism of snapping Scratch blocks together?

NOTES

+ A major goal of this activity is to establish a culture of fearlessness, exploration, and peer collaboration. It is expected that students (and their teachers!) will not know everything ahead of time – and the environment becomes a space where everyone is learning together.
+ Make sure that your computers have the latest version of Flash to run Scratch:
  http://helpx.adobe.com/flash-player.html

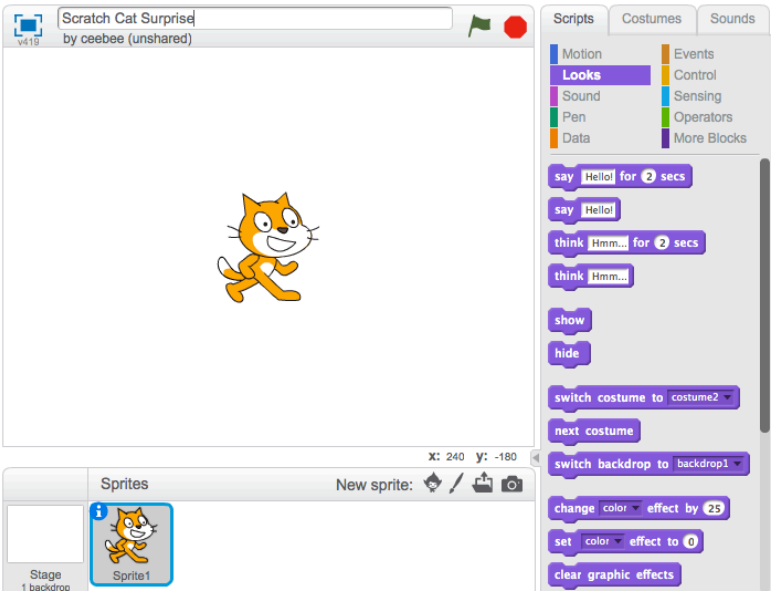
NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

SCRATCH SURPRISE

CAN YOU MAKE THE SCRATCH CAT DO SOMETHING SURPRISING?

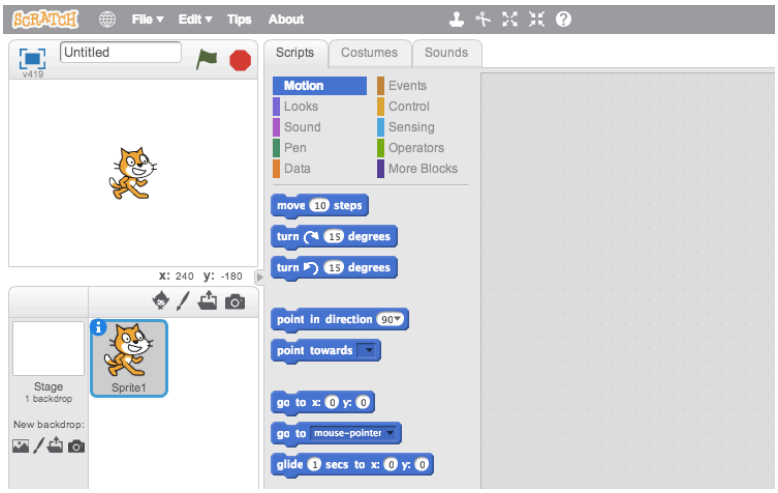
In this activity, you will create a new project with Scratch and explore different Scratch blocks to make the cat do something surprising! What will you create?



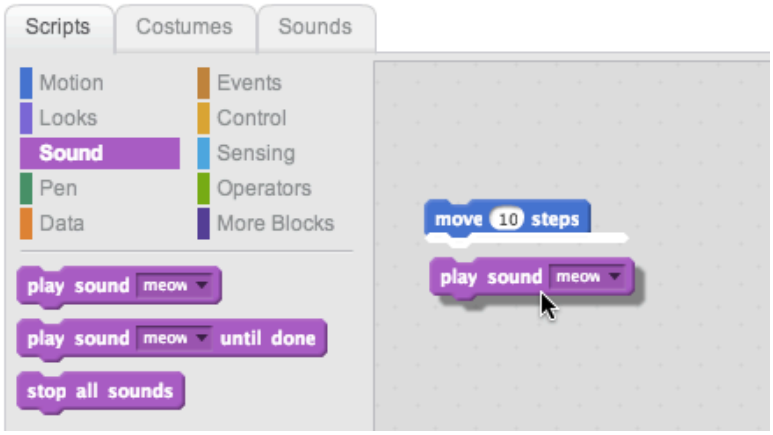
START HERE

- ❑ Go to the Scratch website: http://scratch.mit.edu

- ❑ Sign into your account.
- ❑ Click on the "Create" tab located at the top left of the browser to start a new project.



- ❑ Time to explore! Try clicking on different parts of the Scratch interface to see what happens.
- ❑ Play with different Scratch blocks! Drag and drop Scratch blocks into the scripting area. Experiment by clicking on each block to see what they do or try snapping blocks together.



UNIT 0 ACTIVITY

SCRATCH STUDIO

SUGGESTED TIME
5–15 MINUTES

OBJECTIVES

By completing this activity, students will:

+ be able to add a project to a studio
+ be able to post comments on other Scratch projects

ACTIVITY DESCRIPTION

- ❑ Scratch studios are one way to collect and organize Scratch projects online. In this activity, help students understand what studios are and how to add a project to a studio. Optionally, have the Scratch Studio handout available to guide students.
- ❑ First, have students navigate to the Scratch website and sign in to their accounts. Next, help students find the Scratch Surprise studio or a class studio you've created. Then, let students share their Scratch Surprise explorations with others by adding their programs to the studio.
- ❑ Encourage students to investigate other projects in the studio. Invite them to add a comment on the project page of two projects in the collection that they find particularly interesting or inspiring. Engage the group in a discussion about how to give appropriate and purposeful feedback.
- ❑ Ask students to think back on their creative explorations by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Scratch Studio handout
- ❑ Scratch Surprise studio
  http://scratch.mit.edu/studios/460431

REFLECTION PROMPTS

+ What are Scratch studios for?
+ What did you find interesting or inspiring about looking at other projects?
+ What two comments did you share?
+ What is "good" feedback?

REVIEWING STUDENT WORK

+ Did students successfully add their projects to the studio?
+ Did students comment appropriately on others' work?

NOTES

+ Create your own studio(s) to collect student work. Start a class Scratch Surprise studio using your Scratch account and then give students the studio link to "turn in" projects. Create one dedicated studio to gather all class projects or distribute activities across separate studios to track student progress.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

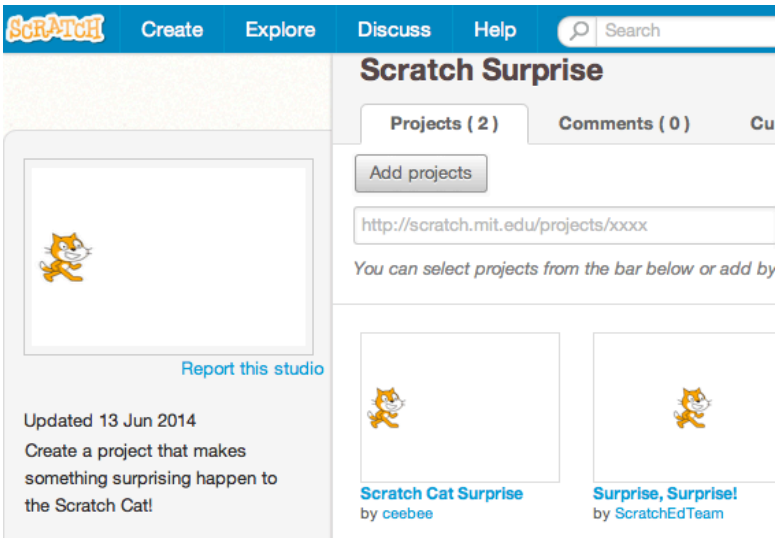
SCRATCH STUDIO

LEARN HOW TO ADD YOUR PROJECT TO AN ONLINE SCRATCH STUDIO!

Studios are collections of Scratch projects. Follow along with the steps below to add your Scratch Surprise program to the Scratch Surprise studio on the Scratch website.

START HERE

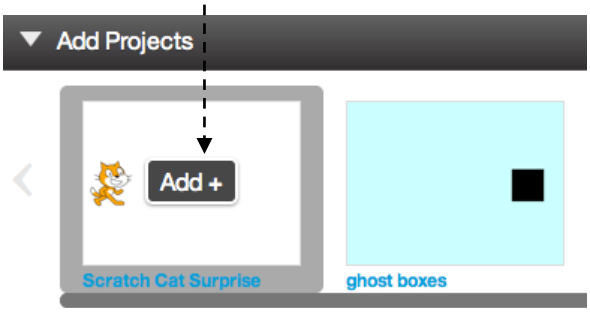
- ❑ Go to the Scratch Surprise studio using this link: http://scratch.mit.edu/studios/460431



- ❑ Sign into your account.

- ❑ Click on "Add Projects" at the bottom of the page to show your your projects, favorite projects, and recently viewed projects.

- ❑ Use the arrows to find your Scratch Surprise project and then click "Add + " to add your project to the studio.



UNIT 0 ACTIVITY

CRITIQUE GROUP

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ divide into small critique groups in order to give and get feedback on design ideas and works-in-progress

ACTIVITY DESCRIPTION

- ❑ Introduce students to the idea of a critique group, a small group of designers who share ideas and projects-in-progress with one another in order to get feedback and suggestions for further development.
- ❑ Optionally, have the Critique Group handout available to guide students in giving feedback.
- ❑ Divide students in smaller groups of 3-4 people. In these critique groups, ask students to take turns sharing their ideas, drafts, or prototypes, for example, Scratch Surprise projects.
- ❑ Let students gather feedback by having their critique group members respond to the Red, Yellow, Green reflection prompts or using the Critique Group handout. Encourage students to record other notes, feedback, and suggestions in their design journals.

RESOURCES

- ❑ Critique Group handout

REFLECTION PROMPTS

+ RED: What is something that doesn't work or could be improved?
+ YELLOW: What is something that is confusing or could be done differently?
+ GREEN: What is something that works well or you really like about the project?

REVIEWING STUDENT WORK

+ Did all students have a chance to share their work and get feedback?

NOTES

+ It can be valuable to have a dedicated group of peers to give you encouragement and feedback on your design iterations. Provide opportunities for students to continue meeting with their critique groups during Units 1-6.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

CRITIQUE GROUP

FEEDBACK FOR: _____

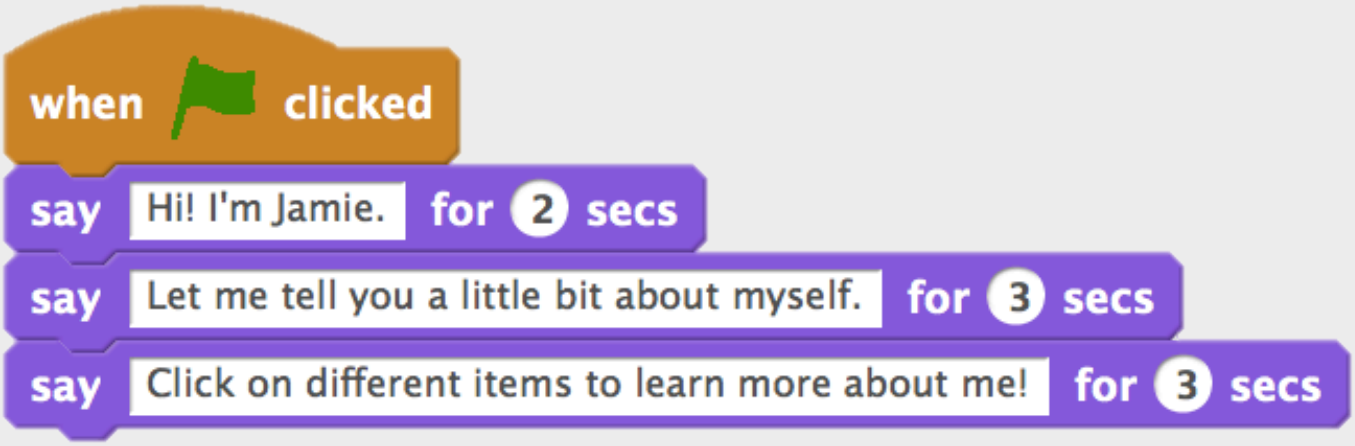
PROJECT TITLE: _____

FEEDBACK BY	[RED] What is something that doesn't work or could be improved?	[YELLOW] What is something that is confusing or could be done differently?	[GREEN] What is something that works well or you really like about the project?

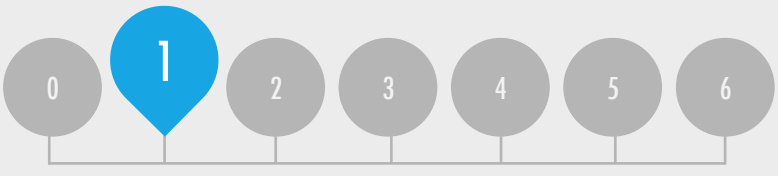
PARTS OF THE PROJECT THAT MIGHT BE HELPFUL TO THINK ABOUT:

+ Clarity: Did you understand what the project is supposed to do?
+ Features: What features does the project have? Does the project work as expected?
+ Appeal: How engaging is the project? Is it interactive, original, sophisticated, funny, or interesting? How did you feel as you interacted with it?

UNIT 1
EXPLORING



YOU ARE HERE



WHAT'S INCLUDED

UNIT 1 OVERVIEW

THE "BIG IDEA"

Many of the educators that we have worked with over the years wrestle with two questions when getting started with creative computing: "What's the best way of helping learners get started?" and "What do I, as teacher, need to know?" The writings of Seymour Papert (a renowned mathematician, educator, and major influence on the development of Scratch through the Logo programming language) serve as inspiration for thinking about these questions.

With respect to the first question, two extreme positions tend to be taken up. Either learners need to be told what to do and should have highly structured experiences – or learners need to be left totally alone to explore under their own direction. Papert, a proponent of the notion that young learners should act as advocates for and explorers of their own thinking and learning, encouraged teachers to seek a balance between teaching and learning. Throughout the guide, we vary the amount of structure in the activities in an effort to provide balance.

With respect to the second question, educators sometimes worry that they don't "know" enough about Scratch to be able to help others. We encourage you to take a broad view of what it means to "know" Scratch. You don't need to know everything about the Scratch interface or how to solve every problem that a learner encounters. But, as Papert noted, educators can serve as cognitive guides, asking questions and helping break down problems into manageable pieces.

> *As they puzzled together the child had a revelation: "Do you mean," he said, "that you really don't know how to fix it?" The child did not yet know how to say it, but what had been revealed to him was that he and the teacher had been engaged together in a research project. The incident is poignant. It speaks of all the times this child entered into teachers' games of "let's do that together" all the while knowing that the collaboration was a fiction. Discovery cannot be a setup; invention cannot be scheduled.*
>
> *(Papert, 1980, p. 115)*

KEY WORDS, CONCEPTS, & PRACTICES

+ experimenting and iterating
+ testing and debugging
+ sequence
+ sprite
+ motion
+ looks
+ sound
+ costume
+ backdrop
+ tips window
+ remix
+ interactive collage
+ pair-share

LEARNING OBJECTIVES

Students will:

+ build on initial explorations of the Scratch environment by creating an interactive Scratch project
+ be introduced to a wider range of Scratch blocks
+ become familiar with the concept of sequence
+ practice experimenting and iterating while creating projects

NOTES

+ Make sure students already have a Scratch account for saving and sharing their projects online.
+ Think about how you plan to access your students' work. For example, you can create class studios to collect projects, have students email you project links, or start a class blog.

CHOOSE YOUR OWN ADVENTURE

This unit includes a mix of structured and open-ended activities that engage students in exploration of the key concept of sequence – identifying and specifying an ordered series of instructions. This is often a powerful moment for students: they're telling the computer what to do, by translating their ideas into blocks of computer code.

From a step-by-step tutorial, to playing with a constrained number of blocks, to a debugging challenge, each activity helps learners build the skills needed to create an About Me project. In the culminating project, learners will explore and experiment with sprites, costumes, looks, backdrops, and sounds to create a personalized, interactive collage in Scratch.

Take advantage of all the activities or pick a few that cater to your students' specific needs and interests; the choice is up to you. If you're not sure where to start, a possible order for the activities is suggested below.

POSSIBLE PATH

SESSION 1

PROGRAMMED TO DANCE

How can you express a sequence of dance moves using simple verbal instructions?

SESSION 2

STEP-BY-STEP

New to Scratch? Create your first Scratch project!

10 BLOCKS

What can you create with only 10 Scratch blocks?

SESSION 3

MY STUDIO

What can be created with Scratch?

DEBUG IT!

Help! Can you debug these five Scratch programs?

SESSION 4 & SESSION 5

ABOUT ME

How can you combine images and sounds to make an interactive collage about yourself?

UNIT 1 ACTIVITY

PROGRAMMED TO DANCE

SUGGESTED TIME
45-60 MINUTES

OBJECTIVES

By completing this activity, students will:

+ learn to express a complex activity using a sequence of simple instructions

ACTIVITY DESCRIPTION

- ❑ Ask for 8 volunteers – four people who don't mind being bossy and four people who don't mind being bossed. Create four bossy/bossed pairs. Optionally, have a projector ready to present the Programmed to Dance videos.
- ❑ For each bossy/bossed pair:
  1. Have the bossed partner facing away from the display and the bossy partner (and the rest of the group) facing the display.
  2. Show the video to the bossy partner and the group, but NOT to the bossed partner.
  3. Ask the bossy partner to describe to their partner (using only words!) how to perform the sequence of dance moves shown in the video.
- ❑ Use this activity to start a discussion about the importance of sequence in specifying a set of instructions. You can let students reflect individually in their design journals or facilitate a group discussion by inviting different bossy/bossed pairs and observers to share their thoughts.

RESOURCES

- ❑ projector (optional)
- ❑ Programmed to Dance videos
  http://vimeo.com/28612347
  http://vimeo.com/28612585
  http://vimeo.com/28612800
  http://vimeo.com/28612970

REFLECTION PROMPTS

+ What was easy/difficult about being the bossy partner?
+ What was easy/difficult about being the bossed partner?
+ What was easy/difficult about watching?
+ How does this activity relate to what we're doing with Scratch?

REVIEWING STUDENT WORK

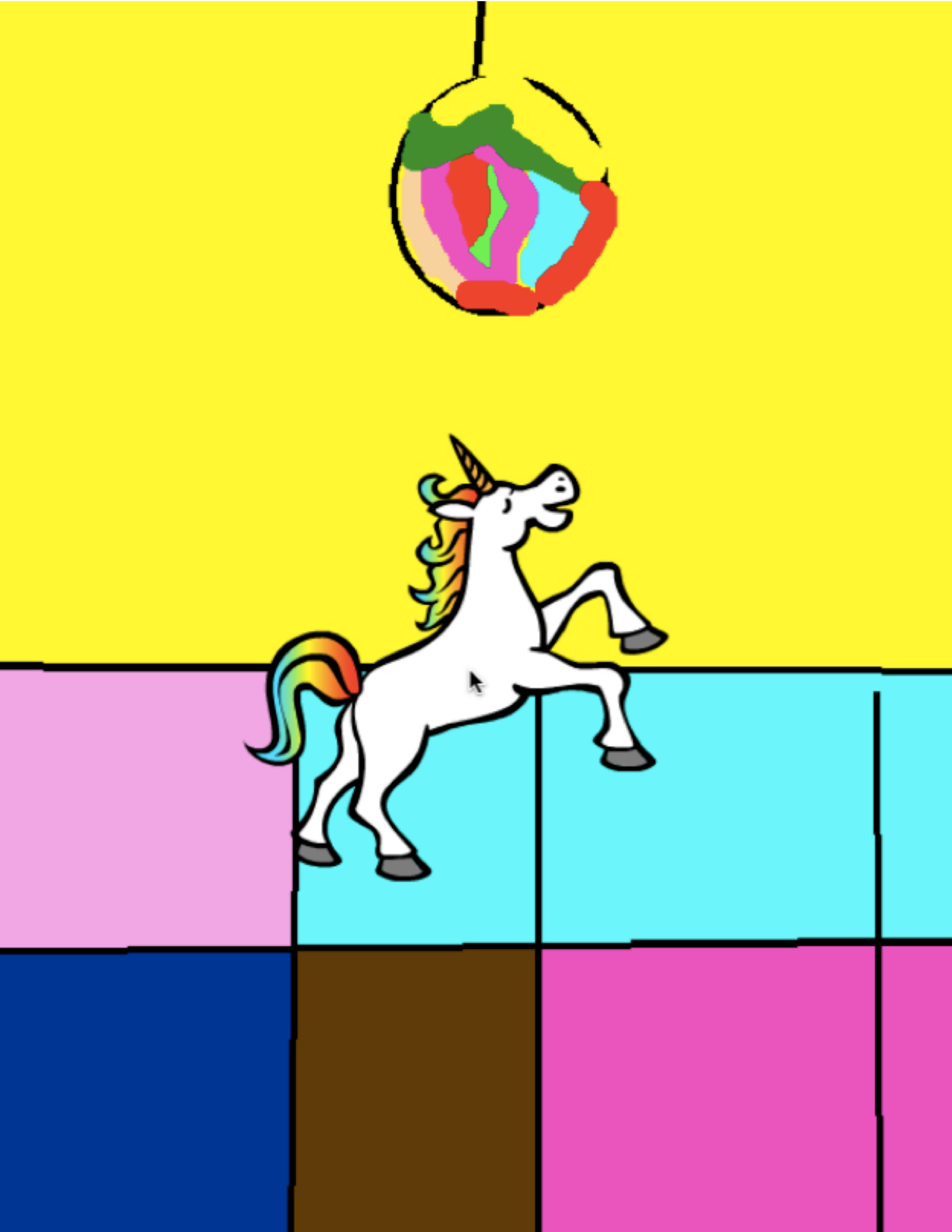
+ Can students explain what is important about sequence when specifying instructions?

NOTES

+ This is one of several activities in this guide that are computer-free. Stepping back from the computer can support fresh perspectives on and new understandings of computational concepts, practices, and perspectives.
+ Have students write down step-by-step instructions for one of the dances. In programming, this is called "pseudocode".

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....



UNIT 1 ACTIVITY

STEP-BY-STEP

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ create a dancing cat in Scratch by following a step-by-step tutorial
+ experience building up a program by experimenting and iterating

ACTIVITY DESCRIPTION

- ❑ Help students sign in to their Scratch accounts and click on the Create button at the top of the Scratch website to open the project editor. Optionally, have the Step-by-Step handout and Scratch Cards available to guide students during the activity.
- ❑ Have students open the Tips window and follow the Getting Started with Scratch step-by-step tutorial to create a dancing cat program. Encourage students to add other blocks and experiment with motion, sprites, looks, costumes, sound, or backdrops to make the project their own.
- ❑ Let students share their first Scratch creations with one another! Optionally, help students share and add their projects to the Step-by-Step studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or as a group discussion.

RESOURCES

- ❑ Step-by-Step handout
- ❑ Step-by-Step studio
  http://scratch.mit.edu/studios/475476
- ❑ Scratch Cards
  http://scratch.mit.edu/help/cards

REFLECTION PROMPTS

+ What was surprising about the activity?
+ How did it feel to be led step-by-step through the activity?
+ When do you feel most creative?

REVIEWING STUDENT WORK

+ Were students able to open Scratch and find the Tips Window?
+ Were students able to create a dancing cat?
+ Were students able to save and share projects?

NOTES

+ If they don't have one already, help learners create a Scratch account using the Unit 0 Scratch Account activity, so that students can save and share their first Scratch project with friends and family.
+ Remind students how to add a project to a studio with the Unit 0 Scratch Studio activity or handout.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

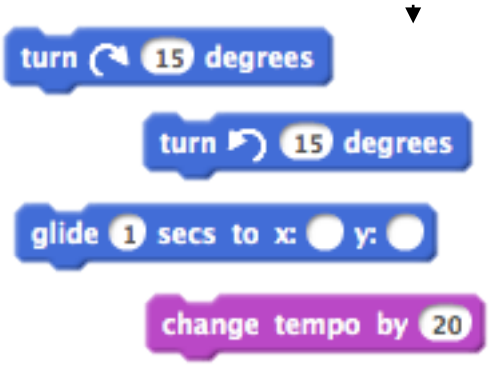
STEP-BY-STEP

NEW TO SCRATCH? CREATE YOUR FIRST SCRATCH PROJECT!

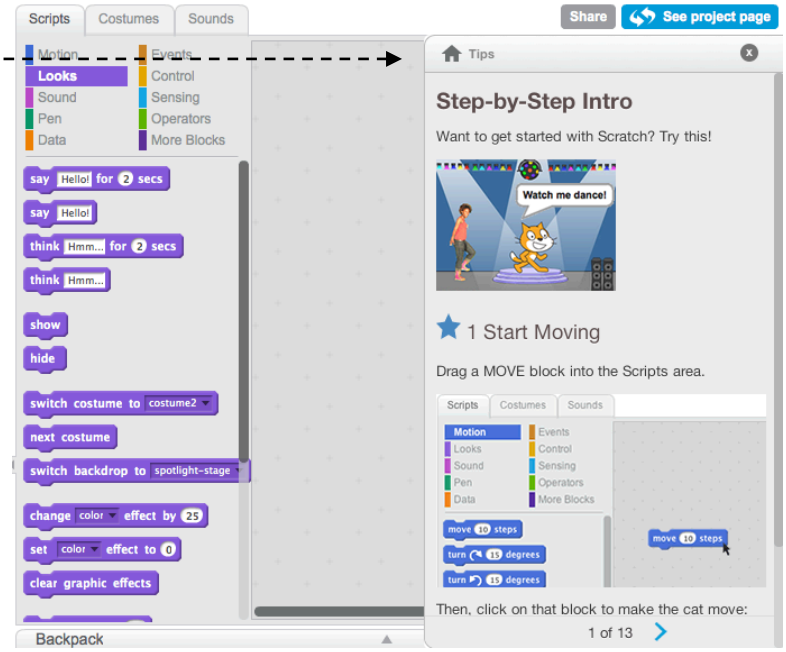
In this activity, you will follow the Step-by-Step Intro in the Tips Window to create a dancing cat in Scratch. Once you have completed the steps, experiment by adding other Scratch blocks to make the project your own.

START HERE

- ❑ Follow the Step-by-Step Intro in the Tips Window.
- ❑ Add more blocks.
- ❑ Experiment to make it your own!



What blocks do you want to experiment with?



THINGS TO TRY

- ❑ Try recording your own sounds.
- ❑ Create different backdrops.
- ❑ Turn your project into a dance party by adding more dancing sprites!
- ❑ Try designing a new costume for your sprite.

FINISHED?

- + Add your project to the Step-by-Step Studio: http://scratch.mit.edu/studios/475476
- + Challenge yourself to do more! Play with adding new blocks, sound, or motion.
- + Help a neighbor!
- + Choose a few new blocks to experiment with. Try them out!

UNIT 1 ACTIVITY

10 BLOCKS

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ create a project with the constraint of only being able to use 10 blocks

ACTIVITY DESCRIPTION

- ❑ Help students sign in to their Scratch accounts and click on the Create button at the top of the Scratch website to start a new project. Optionally, have the 10 Blocks handout available to guide students during the activity.
- ❑ Give students time to create a project with only these 10 Scratch blocks: go to, glide, say, show, hide, set size to, play sound until done, when this sprite clicked, wait, and repeat. Remind students to use each block at least once in their project and encourage them to experiment with different sprites, costumes, or backdrops.
- ❑ Invite students to share their projects in their critique groups (see the Unit 0 Critique Group activity). Optionally, have students add their projects to the 10 Blocks studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ 10 Blocks handout
- ❑ 10 Blocks studio
  http://scratch.mit.edu/studios/475480

REFLECTION PROMPTS

+ What was difficult about being able to use only 10 blocks?
+ What was easy about being able to use only 10 blocks?
+ How did it make you think of things differently?

REVIEWING STUDENT WORK

+ Do projects include all 10 blocks?
+ How do different students react to the idea of creating with constraints? What might this tell you about how this student learns?

NOTES

+ It's surprising how much one can do with just 10 blocks! Take this opportunity to encourage different ideas and celebrate creativity by inviting a few students to present their projects in front of the class or by exploring other projects online in the 10 Blocks studio.

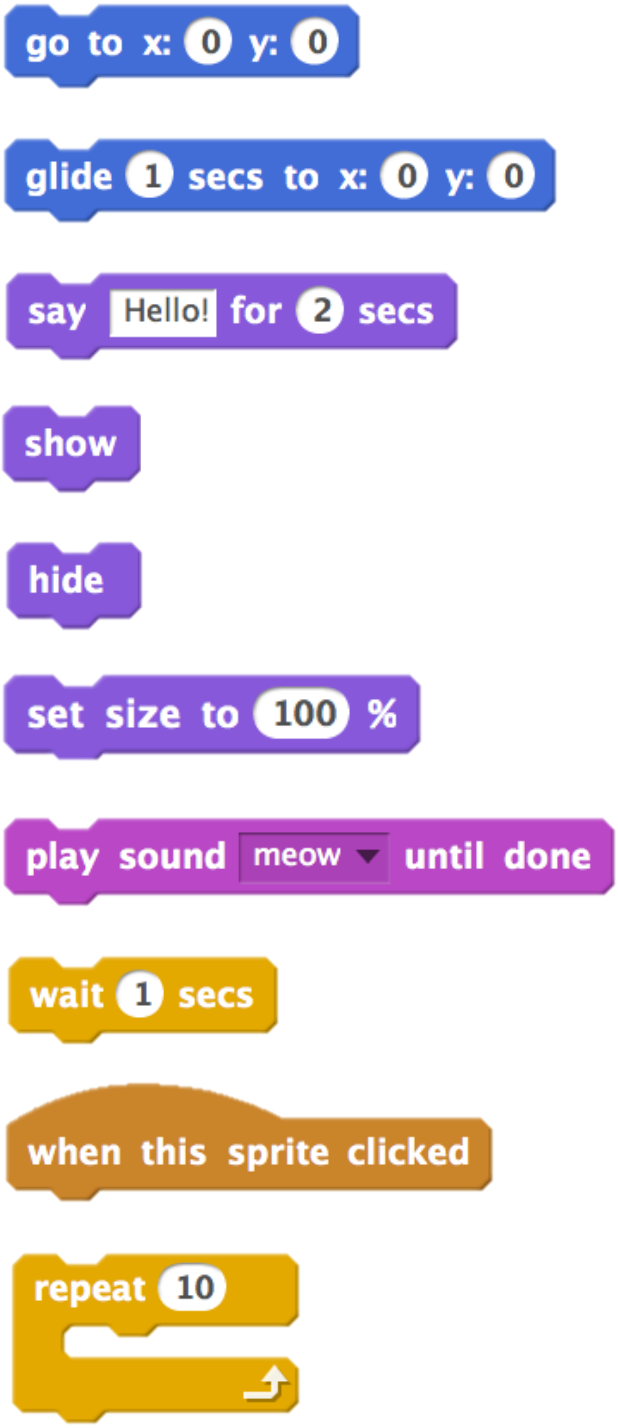
NOTES TO SELF

- ❑ _____
- ❑ _____
- ❑ _____
- ❑ _____

10 BLOCKS

WHAT CAN YOU CREATE WITH ONLY 10 SCRATCH BLOCKS?

Create a project using only these 10 blocks. Use them once, twice, or multiple times, but use each block at least once.



START HERE

- ❑ Test ideas by experimenting with each block.
- ❑ Mix and match blocks in various ways.
- ❑ Repeat!

FEELING STUCK?

THAT'S OKAY! TRY THESE THINGS...

- ❑ Test ideas by trying out different block combinations. Mix and match blocks until you find something that interests you!
- ❑ Try brainstorming ideas with a neighbor!
- ❑ Explore other projects to see what others are doing in Scratch. This can be a great way to find inspiration!

FINISHED?

- + Add your project to the 10 Blocks Studio: http://scratch.mit.edu/studios/475480
- + Play with different sprites, costumes, or backdrops.
- + Challenge yourself to do more! See how many different projects you can create with these 10 blocks.
- + Swap projects with a partner and remix each others' creations.

UNIT 1 ACTIVITY

MY STUDIO

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ investigate the range of creative possibilities with Scratch by exploring some of the millions of projects on the Scratch website
+ curate a collection of 3 or more Scratch projects in a Scratch studio

ACTIVITY DESCRIPTION

- ❑ Optionally, demonstrate how to create a new studio or have the My Studio handout available to guide students.
- ❑ Optionally, show example inspiration studios using the links provided. Give students 10 minutes to browse existing Scratch projects on the Scratch homepage and search for interesting programs using the Explore page.
- ❑ Ask students to identify three or more Scratch projects that can be used to inform and inspire a project of their own. Help students create a new studio from their My Stuff page and add the inspirational projects to the studio.
- ❑ Invite students to share their approaches for finding inspirational programs. We suggest pair-share: have students share studios and discuss search strategies in pairs.
- ❑ Ask students to think back on the process of discovery by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ My Studio handout
- ❑ example studios
  http://scratch.mit.edu/studios/211580
  http://scratch.mit.edu/studios/138296
  http://scratch.mit.edu/studios/138297
  http://scratch.mit.edu/studios/138298

REFLECTION PROMPTS

+ What search strategies did you use to find interesting projects?
+ How might each example project help with future work?
+ It's important to give credit to sources of inspiration. How can you give credit for inspiration from these projects?

REVIEWING STUDENT WORK

+ Are there three or more projects in the studio?
+ What do these projects tell you about your students' design interests?

NOTES

+ If students don't have individual Scratch accounts, create a class studio that students can curate.
+ A variety of studios can be created - students could collect Scratch projects that are similar in theme or topic to what they want to create or gather programs that include techniques or assets to incorporate in a future creation.

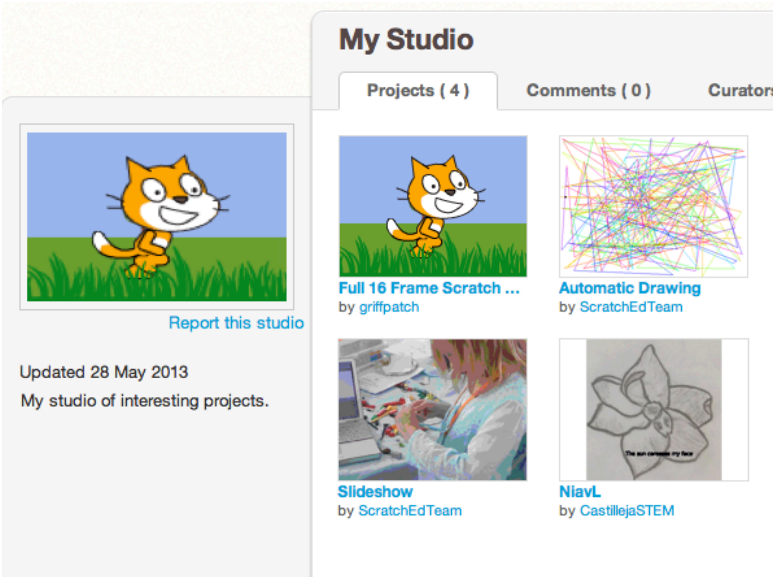
NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

MY STUDIO

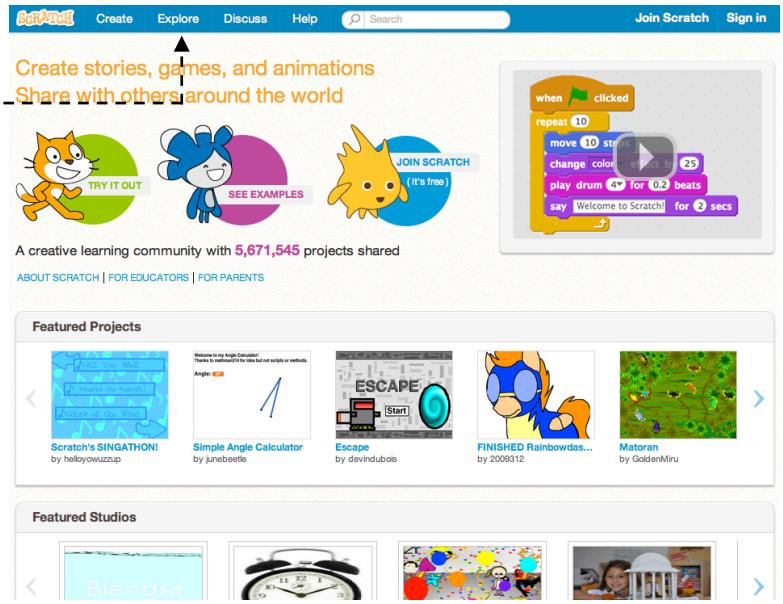
WHAT CAN BE CREATED WITH SCRATCH?

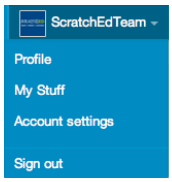
In this activity, you will investigate the range of creative possibility with Scratch by exploring some of the millions of projects on the Scratch website – and start a collection of favorites in a Scratch studio!

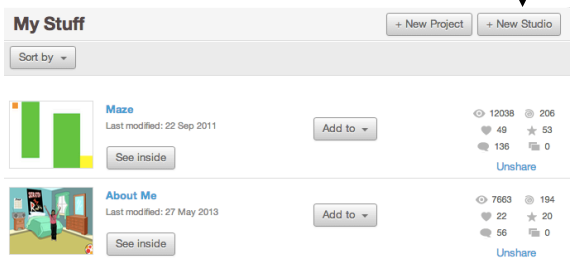


START HERE

- ❑ Browse projects on the Scratch homepage OR click on "Explore" to search for specific types of projects.
- ❑ Create a new studio from your My Stuff page.
- ❑ Add three (or more!) inspiring projects to your studio.







THINGS TO TRY

- ❑ Use the search bar to find projects that relate to your interests.
- ❑ Explore each of the Animations, Art, Games, Music, & Stories categories on the Explore page.
- ❑ Look through the Featured Studios on the homepage for ideas.

FINISHED?

- + Challenge yourself to do more! The more Scratch projects you explore, the more you learn about what can be accomplished in Scratch!
- + Find studios created by other Scratchers that you find interesting!
- + Ask a neighbor what strategies they used to find interesting projects.
- + Share your newly created studio with a neighbor!

UNIT 1 ACTIVITY

DEBUG IT!

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ investigate the problem and find a solution to five debugging challenges
+ explore a range of concepts (including sequence) through the practices of testing and debugging
+ develop a list of strategies for debugging projects

ACTIVITY DESCRIPTION

- ❑ Optionally, have the Unit 1 Debug It! handout available to guide students during the activity.
- ❑ Help students open the Debug It! programs from the Unit 1 Debug It! studio or by following the project links listed on the Unit 1 Debug It! handout. Encourage students to click on the "Look Inside" button to investigate the buggy program, tinker with problematic code, and test possible solutions.
- ❑ Give students time to test and debug each Debug It! challenge. Optionally, have students use the remix function in Scratch to fix the bugs and save corrected programs.
- ❑ Ask students to reflect back on their testing and debugging experiences by responding to the reflection prompts in their design journals or in a group discussion.
- ❑ Create a class list of debugging strategies by collecting students' problem finding and problem solving approaches.

RESOURCES

- ❑ Unit 1 Debug It! handout
- ❑ Unit 1 Debug It! studio
  http://scratch.mit.edu/studios/475483

REFLECTION PROMPTS

+ What was the problem?
+ How did you identify the problem?
+ How did you fix the problem?
+ Did others have alternative approaches to fixing the problem?

REVIEWING STUDENT WORK

+ Were students able to solve all five bugs? If not, how might you clarify the concepts expressed in the unsolved programs?
+ What different testing and debugging strategies did students employ?

NOTES

+ This activity works well in groups! Get students working in teams of 2-4 people to collectively problem solve and share debugging strategies.
+ Testing and debugging is probably the most common activity of programmers. Things rarely work as planned, so developing a set of testing and debugging strategies will be beneficial to any computational creator.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

DEBUG IT!

HELP! CAN YOU DEBUG THESE FIVE SCRATCH PROGRAMS?

In this activity, you will investigate what is going awry and find a solution for each of the five Debug It! challenges.

START HERE

- ❑ Go to the Unit 1 Debug It! studio: http://scratch.mit.edu/studios/475483
- ❑ Test and debug each of the five debugging challenges in the studio.
- ❑ Write down your solution or remix the buggy program with your solution.

❑ **DEBUG IT! 1.1** http://scratch.mit.edu/projects/10437040

When the green flag is clicked, both Gobo and Scratch Cat should start dancing. But only Scratch Cat starts Dancing! How do we fix the program?

❑ **DEBUG IT! 1.2** http://scratch.mit.edu/projects/10437249

In this project, when the green flag is clicked, the Scratch Cat should start on the left side of the stage, say something about being on the left side, glide to the right side of the stage, and say something about being on the right side. It works the first time the green flag is clicked, but not again. How do we fix the program?

❑ **DEBUG IT! 1.3** http://scratch.mit.edu/projects/10437366

The Scratch Cat should do a flip when the space key is pressed. But when the space key is pressed, nothing happens! How do we fix the program?

❑ **DEBUG IT! 1.4** http://scratch.mit.edu/projects/10437439

In this project, the Scratch Cat should pace back and forth across the stage, when it is clicked. But the Scratch Cat is flipping out – and is walking upside down! How do we fix the program?

❑ **DEBUG IT! 1.5** http://scratch.mit.edu/projects/10437476

In this project, when the green flag is clicked, the Scratch Cat should saw 'Meow, meow, meow!' in a speech bubble and as a sound. But the speech bubble happens before the sound – and the Scratch Cat only makes one 'Meow' sound! How do we fix the program?

FEELING STUCK?

THAT'S OKAY! TRY THESE THINGS...

- ❑ Make a list of possible bugs in the program.
- ❑ Keep track of your work! This can be a useful reminder of what you have already tried and point you toward what to try next.
- ❑ Share and compare your problem finding and problem solving approaches with a neighbor until you find something that works for you!

FINISHED?

- + Discuss your testing and debugging practices with a partner. Make note of the similarities and differences in your strategies.
- + Add code commentary by right clicking on blocks in your scripts. This can help others understand different parts of your program!
- + Help a neighbor!

UNIT 1 ACTIVITY

ABOUT ME

SUGGESTED TIME
45-60 MINUTES

OBJECTIVES

By completing this activity, students will:

+ become familiar with a wider range of Scratch blocks
+ be able to create an open-ended Scratch project that is an interactive digital representation of their personal interests

ACTIVITY DESCRIPTION

- ❑ Introduce students to the concept of the interactive collage, a Scratch project that represents aspects of themselves through clickable sprites. Optionally, show interactive project examples from the About Me studio.
- ❑ Have students sign in to their Scratch accounts and open a new project. Optionally, have the About Me handout and Scratch Cards available to provide guidance. Give students time to create an About Me interactive collage Scratch project, encouraging them to build up their programs by experimenting and iterating.
- ❑ Allow students to share their works-in-progress with others. We suggest pair-share: have students share and discuss their projects in pairs. Optionally, invite students to add their projects to the About Me studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ About Me handout
- ❑ About Me studio
  http://scratch.mit.edu/studios/475470
- ❑ Scratch Cards
  http://scratch.mit.edu/help/cards

REFLECTION PROMPTS

+ What are you most proud of? Why?
+ What did you get stuck on? How did you get unstuck?
+ What might you want to do next?
+ What did you discover from looking at others' About Me projects?

REVIEWING STUDENT WORK

+ Do projects make creative use of sprites, costumes, looks, backdrops, or sound?
+ Are projects interactive? Can users interact with various elements within the project?

NOTES

+ Example projects can simultaneously inspire and intimidate, open the creative space and constrain it. Encourage a wide range of creations; diversity is great!
+ Students can further personalize projects by using a camera or webcam to bring images into the project.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

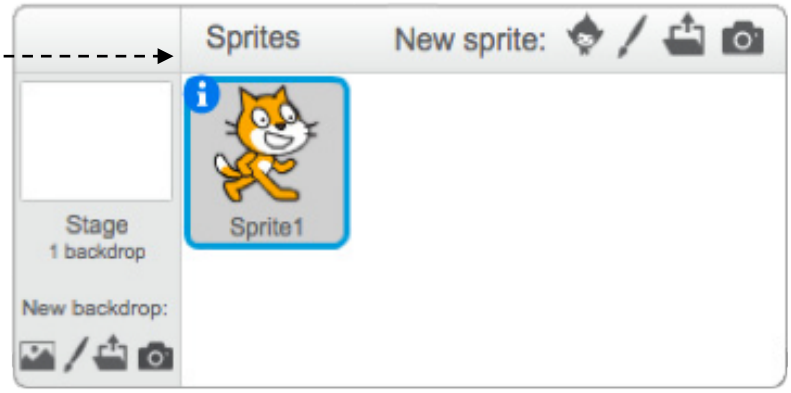
ABOUT ME

HOW CAN YOU COMBINE INTERESTING IMAGES AND SOUNDS TO MAKE AN INTERACTIVE COLLAGE ABOUT YOURSELF?

Experiment with sprites, costumes, backdrops, looks, and sounds to create an interactive Scratch project – a project that helps other people learn more about YOU and the ideas, activities, and people that you care about.

START HERE

- ❑ Create a sprite.
- ❑ Make it interactive.
- ❑ Repeat!

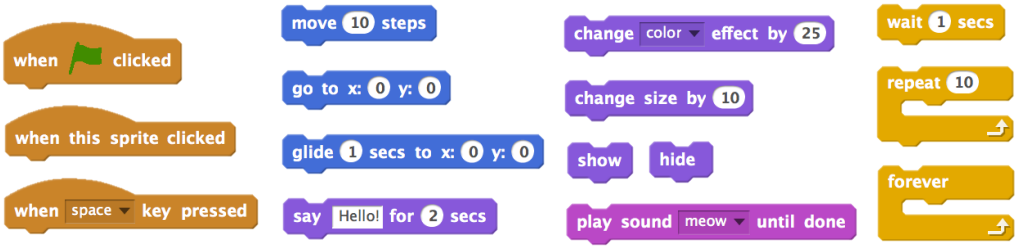


Make your sprite interactive by adding scripts that have the sprite respond to clicks, key presses, and more!

THINGS TO TRY

- ❑ Use costumes to change how your sprite looks.
- ❑ Create different backdrops.
- ❑ Try adding sound to your project.
- ❑ Try adding movement into your collage.

BLOCKS TO PLAY WITH



FINISHED?

+ Add your project to the About Me Studio: http://scratch.mit.edu/studios/475470
+ Challenge yourself to do more! Play with adding new blocks, sound, or motion!
+ Help a neighbor!

UNIT 2
ANIMATIONS

YOU ARE HERE

WHAT'S INCLUDED

UNIT 2 OVERVIEW

THE "BIG IDEA"

Kids have shared more than six million projects in the Scratch online community – animations, stories, games, and beyond – and one of our goals with the guide is to reflect this enormous diversity of creations. Within activities, we support opportunities to personalize and avoid presenting challenges that have only one "right" answer; across activities, we engage learners in a variety of genres. In this unit, we start to explore this creative diversity with a deep dive into animation, art, and music.

Creative diversity in Scratch has often been highlighted by learners. Here are a few quotes from learners who were asked, "If you had to explain what Scratch is to one of your friends, how would you describe it?"

It's a program that lets you explore your imagination. You can do whatever you want in it. You can create anything. There really is no limit to what you can make. You design your own stuff, and once you start you just don't want to stop because as you learn more, you can see there's more possibilities, and the more possibilities there are, the more you want to expand on what you just learned.

Bradley, 12 years old

It's just that there's endless possibilities. It's not like you can just make this project or this project and that's all that you can make.

Nevin, 9 years old

It's really great to express yourself creatively. You could do anything with it. You can make video games, music, art, videos, anything. The possibilities are endless, no limitations, really.

Lindsey, 12 years old

Well, I like that you can sort of do anything on it. It's like you can do whatever you want, really. You can be as creative as you want to be.

Aaron, 10 years old

LEARNING OBJECTIVES

Students will:

+ be introduced to the computational thinking concepts of loops, events, and parallelism
+ become more familiar with the concepts of sequence
+ experiment with new blocks in the Events, Control, Sound, and Looks categories
+ explore various arts-themed Scratch programs
+ create an animated music video project

KEY WORDS, CONCEPTS, & PRACTICES

+ loops
+ events
+ parallelism
+ control
+ broadcast
+ scripts
+ presentation mode
+ bitmap
+ vector
+ animation
+ gallery walk

NOTES

+ Many activities in this unit include elements of sound and music. We recommend having headphones readily available for students.

CHOOSE YOUR OWN ADVENTURE

Programming in Scratch is like directing theatre. In theatre, just as in Scratch, there are characters (sprites, in Scratch parlance), costumes, backdrops, scripts, and a stage. Scratch programming utilizes cues called "events", which signal when things should occur in a project, such as: activating a project (when green flag clicked), triggering sprites' actions (when this sprite clicked), or even sending a silent cue across sprites or backdrops (broadcast).

Inspired by the theatre metaphor, this unit's arts-themed activities are designed to help students explore the computational concepts of loops, events, and parallelism, culminating in the design of personalized music videos.

POSSIBLE PATH

SESSION 1

PERFORMING SCRIPTS

Play the part of a sprite by acting out different Scratch blocks and scripts.

BUILD-A-BAND

Create your own musical group by making interactive instruments.

SESSION 2

ORANGE SQUARE, PURPLE CIRCLE

What project can you create that includes an orange square and a purple circle?

SESSION 3

IT'S ALIVE!

Can you animate it? Experiment with multiple costumes to bring an image to life.

SESSION 4

DEBUG IT!

Help! Can you debug these five Scratch programs?

SESSION 4 & SESSION 5

MUSIC VIDEO

How can you combine animation with music to create your own Scratch-inspired music video?

UNIT 2 ACTIVITY

PERFORMING SCRIPTS

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ be introduced to the concepts of events (one thing causing another thing to happen) and parallelism (things happening at the same time) through performance
+ be able to explain what events are and how they work in Scratch
+ be able to explain what parallelism is and how it works in Scratch

ACTIVITY DESCRIPTION

- ❑ Optionally, have a projector connected to a computer with Scratch open to display which blocks and scripts will be performed.
- ❑ Ask for two volunteers.
- ❑ Prompt the two volunteers to act out a series of instructions (either by "programming" the volunteers through the Scratch interface or through printed-out physical versions of the Scratch blocks).
  - Have one person do one thing (like walk across the room).
  - Have that person "reset".
  - Have that person do two things simultaneously (like walk across the room and talk).
  - Add the second person, by having the second person simultaneously (but independently) do a task, like talking.
  - Have the second person do a dependent task, like responding to the first person instead of talking over.
- ❑ Reflect on the experience as a group to discuss the concepts of events and parallelism using the reflection prompts to the right.

RESOURCES

- ❑ projector (optional)
- ❑ physical Scratch blocks (optional)

REFLECTION PROMPTS

+ What are the different ways that actions were triggered?
+ What are the mechanisms for events in Scratch?
+ What were the different ways in which things were happening at the same time?
+ What are the mechanisms that enable parallelism in Scratch?

REVIEWING STUDENT WORK

+ Can students explain what events and parallelism are and how they work in Scratch?

NOTES

+ This activity highlights the notion of "reset", which is something Scratchers often struggle with as they get started. If they want things to start in a particular location, with a particular look, etc., students need to understand that they are completely responsible for programming those setup steps.
+ This activity can be useful for demonstrating the broadcast and when I receive block pair.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....


SCRATCH'S
CENTRAL
THEATRICAL
METAPHOR
SPRITE
COSTUME
SCRIPT
STAGE

UNIT 2 ACTIVITY

BUILD-A-BAND

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ create a program that combines interactive sprites with interesting sounds
+ develop greater fluency with sequence, loops, events, and parallelism
+ practice experimenting and iterating in building up project creations

ACTIVITY DESCRIPTION

- ❑ Optionally, show example projects from the Build-a-Band studio and have the Build-a-Band handout available to guide students.
- ❑ Give students time to create interactive instruments by pairing sprites with sounds. Encourage them to experiment with different ways to express sounds in Scratch by exploring other blocks in the Sounds category or using the editing tools within the Sounds tab.
- ❑ Allow students to demonstrate their bands to one another or let students walk around to interact with classmates' instruments. We recommend a gallery walk: have students put their projects in presentation mode and then invite them to walk around and explore each other's projects. Optionally, have students add their projects to the Build-a-Band studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Build-a-Band handout
- ❑ Build-a-Band studio
  http://scratch.mit.edu/studios/475523

REFLECTION PROMPTS

+ What did you do first?
+ What did you do next?
+ What did you do last?

REVIEWING STUDENT WORK

+ Do projects make creative use of sounds?
+ Are the sprites in the projects interactive?

NOTES

+ To share as a whole group, have students perform their Scratch instruments together to form a class band!

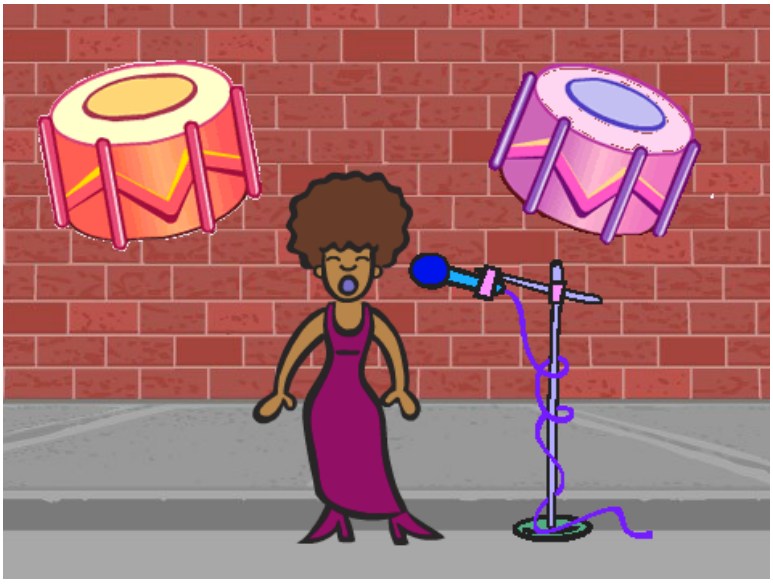
NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

BUILD-A-BAND

HOW CAN YOU UTILIZE SCRATCH TO CREATE SOUNDS, INSTRUMENTS, BANDS, OR STYLES OF MUSIC THAT REPRESENT THE MUSIC YOU LOVE MOST?

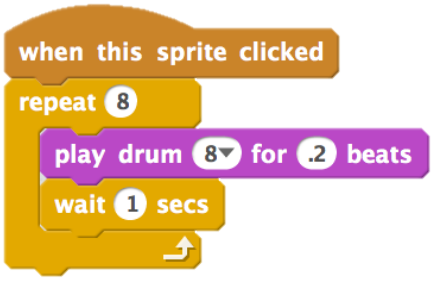
In this activity, you will build your own music-inspired Scratch project by pairing sprites with sounds to design interactive instruments.



START HERE

- ❑ Create a sprite.
- ❑ Add sound blocks.
- ❑ Experiment with ways to make your instruments interactive.

Choose instruments from the sprite library or create your own.



THINGS TO TRY

- ❑ Use repeat blocks to make a sound play more than once.
- ❑ Import or record your own sounds or experiment with the Sounds editor.
- ❑ Try playing with the tempo blocks to speed up or slow down the rhythm.

FINISHED?

- + Add your project to the Build-A-Band Studio: http://scratch.mit.edu/studios/475523
- + Challenge yourself to do more! Invent a new instrument or record your own sounds.
- + Help a neighbor!

UNIT 2 ACTIVITY

ORANGE SQUARE, PURPLE CIRCLE

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ express their creativity by completing an arts-themed challenge
+ gain more fluency with Looks blocks and the paint editor

ACTIVITY DESCRIPTION

- ❑ Optionally, show example projects from the Orange Square, Purple Circle studio and have the Orange Square, Purple Circle handout available to guide students.
- ❑ Give students time to create a project that includes an orange square and a purple circle. Invite students to experiment with Looks blocks and the paint editor to explore their artistic abilities.
- ❑ Encourage students to share their creative work with others. We recommend gallery walk: have students put their projects in presentation mode and then invite them to walk around and explore each other's projects. Optionally, have students add their projects to the Orange Square, Purple Circle studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Orange Square, Purple Circle handout
- ❑ Orange Square, Purple Circle studio
  http://scratch.mit.edu/studios/475527

REFLECTION PROMPTS

+ How did you incorporate an orange square and a purple circle into your project? Where did this idea come from?
+ What was challenging about this activity?
+ What was surprising about this activity?

REVIEWING STUDENT WORK

+ Do projects include an orange square and a purple circle?

NOTES

+ If students have questions, remind them that they can open the Tips Window to learn more about specific blocks or different parts of the Scratch editor.
+ Scratch supports both bitmap and vector graphics. Help students navigate to the vector mode or bitmap mode button in the paint editor to design and manipulate different types of images and text.

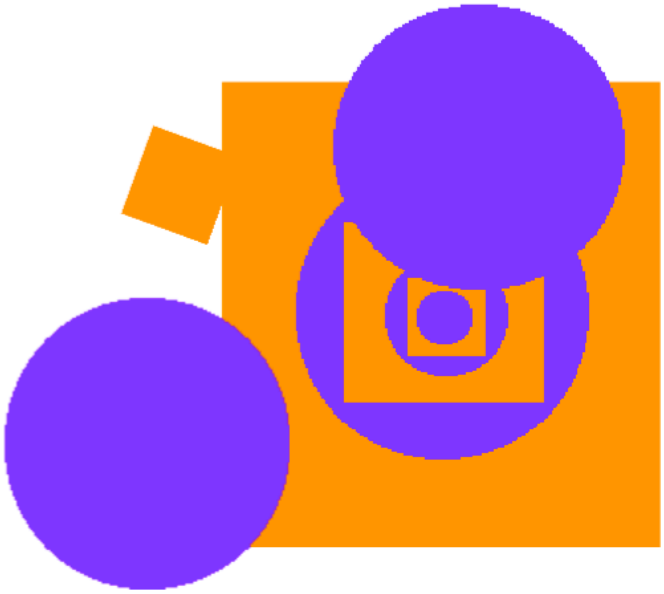
NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

ORANGE SQUARE, PURPLE CIRCLE

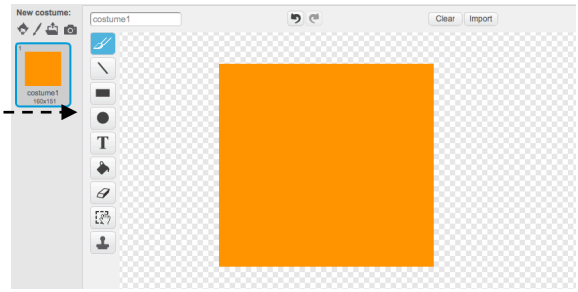
WHAT PROJECT CAN YOU CREATE THAT INCLUDES AN ORANGE SQUARE AND A PURPLE CIRCLE?

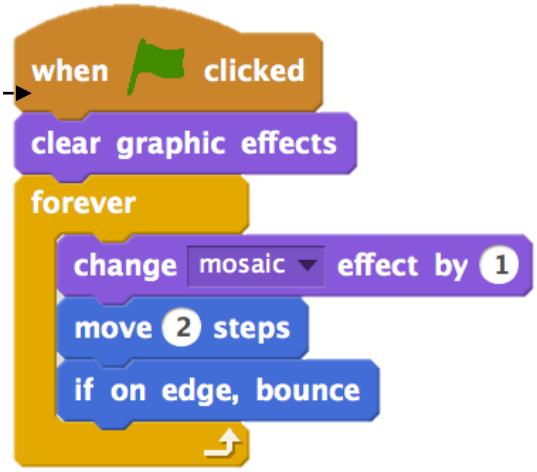
In this challenge, you'll create a project that includes an orange square and a purple circle. What will you create?



START HERE

- ❑ Draw your sprites using the Paint Editor.
- ❑ Add different Looks and Motion blocks to bring your sprites to life.
- ❑ Repeat!





FEELING STUCK?

THAT'S OKAY! TRY THESE THINGS...

- ❑ Try brainstorming with a neighbor!
- ❑ Create a list of things you would like to try before you start building your project in Scratch!
- ❑ Explore other projects to see what others are doing in Scratch – this can be a great way to find inspiration!

FINISHED?

- + Add your project to the Orange Square, Purple Circle Studio: http://scratch.mit.edu/studios/475527
- + Explore the difference between bitmap mode and vector mode, located at the bottom of the paint editor.
- + Challenge yourself to do more! Add another shape and color.
- + Swap projects with a partner and remix each other's creations.
- + Help a neighbor!

UNIT 2 ACTIVITY

IT'S ALIVE!

SUGGESTED TIME
30–45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ become more familiar with the computational concepts of sequence and loops by experimenting with Control blocks
+ be able to explain the difference between sprites and costumes
+ practice experimenting and iterating through developing an animation project

ACTIVITY DESCRIPTION

- ❑ Optionally, show example projects from the It's Alive! studio and have the It's Alive! handout available to guide students.
- ❑ Introduce the concept of an animation as looping through a series of incrementally different pictures, such as in a flipbook or a claymation film. Encourage students to explore loops by changing costumes or backdrops to create an animation.
- ❑ Invite students to share their work with others by hosting a gallery walk: have students put their projects in presentation mode and then invite them to walk around and explore each other's projects. Optionally, have students add their projects to the It's Alive studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ It's Alive! handout
- ❑ It's Alive! studio
  http://scratch.mit.edu/studios/475529

REFLECTION PROMPTS

+ What is the difference between a sprite and a costume?
+ What is an animation?
+ List three ways you experience loops in real life (e.g., going to sleep every night).

REVIEWING STUDENT WORK

+ Can students distinguish sprites and costumes?
+ Some Scratchers are particularly interested in developing animation projects and prefer to spend their time drawing and designing sprites, costumes, or backdrops. How might you engage students in both the aesthetic and technical aspects of projects?

NOTES

+ The difference between sprites and costumes is often a source of confusion for Scratchers. The metaphor of actors wearing multiple costumes can help clarify the difference.
+ Students can animate their own image by taking pictures of themselves using a camera or webcam.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

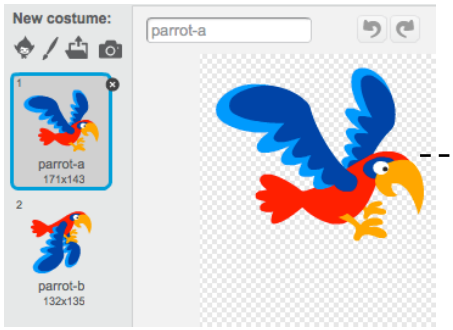
IT'S ALIVE!

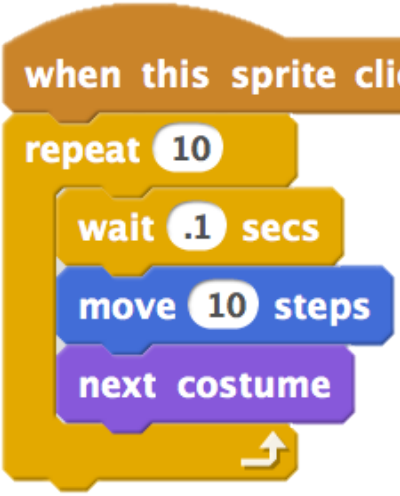
HOW CAN YOU TAKE AN IMAGE OR A PHOTO AND MAKE IT COME ALIVE?

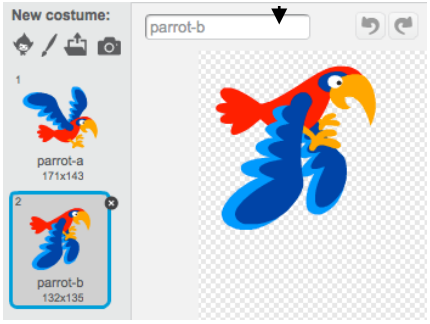
In this activity, you will explore ways of bringing sprites, images, and ideas to life as an animation by programming a series of costume changes.

START HERE

- ❑ Choose a sprite.
- ❑ Add a different costume.
- ❑ Add blocks to make the image come alive.
- ❑ Repeat!







THINGS TO TRY

- ❑ Try sketching your animation ideas on paper first – like a flipbook.
- ❑ Experiment with different blocks and costumes until you find something you enjoy.
- ❑ Need some inspiration? Find projects in the Animation section of the Explore page.

FINISHED?

- + Add your project to the It's Alive studio: http://scratch.mit.edu/studios/475529
- + Challenge yourself to do more! Add more features to your project to make your animations look even more lifelike.
- + Help a neighbor!
- + Share your project with a partner and walk them through your design process.
- + Find an animated project you're inspired by and remix it!

UNIT 2 ACTIVITY

DEBUG IT!

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ investigate the problem and find a solution to five debugging challenges
+ explore a range of concepts (including sequence and loops) through the practices of testing and debugging
+ develop a list of strategies for debugging projects

ACTIVITY DESCRIPTION

- ❑ Optionally, have the Unit 2 Debug It! handout available to guide students during the activity.
- ❑ Help students open the Debug It! programs from the Unit 2 Debug It! studio or by following the project links listed on the Unit 2 Debug It! handout. Encourage students to click on the "Look Inside" button to investigate the buggy program, tinker with problematic code, and test possible solutions.
- ❑ Give students time to test and debug each Debug It! challenge. Optionally, have students use the remix function in Scratch to fix the bugs and save corrected programs.
- ❑ Ask students to reflect back on their testing and debugging experiences by responding to the reflection prompts in their design journals or in a group discussion.
- ❑ Create a class list of debugging strategies by collecting students' problem finding and problem solving approaches.

RESOURCES

- ❑ Unit 2 Debug It! handout
- ❑ Unit 2 Debug It! studio
  http://scratch.mit.edu/studios/475539

REFLECTION PROMPTS

+ What was the problem?
+ How did you identify the problem?
+ How did you fix the problem?
+ Did others have alternative approaches to fixing the problem?

REVIEWING STUDENT WORK

+ Were students able to solve all five bugs? If not, how might you clarify the concepts expressed in the unsolved programs?
+ What different testing and debugging strategies did students employ?

NOTES

+ Facilitate this activity in a whole group by having students act out the Debug It! programs in a similar way to the Performing Scripts activity, or introduce performing scripts as a new strategy for testing and debugging projects.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

DEBUG IT!

HELP! CAN YOU DEBUG THESE FIVE SCRATCH PROGRAMS?

In this activity, you will investigate what is going awry and find a solution for each of the five Debug It! challenges.

START HERE

- ❑ Go to the Unit 2 Debug It! Studio: http://scratch.mit.edu/studios/475539
- ❑ Test and debug each of the five debugging challenges in the studio.
- ❑ Write down your solution or remix the buggy program with your solution.

❑ **DEBUG IT! 2.1** http://scratch.mit.edu/projects/23266426

In this project, Scratch Cat wants to show you a dance. When you click on him, he should do a dance while a drum beat plays along with him. However, as soon as he starts to dance he stops but the drumming continues without him! How do we fix this program?

❑ **DEBUG IT! 2.2** http://scratch.mit.edu/projects/24268476

In this project, when the green flag is clicked Pico should move towards Nano. When Pico reaches Nano, Pico should say "Tag, you're it!" and Nano says "My turn!" But something is wrong! Pico doesn't say anything to Nano. How do we fix the program?

❑ **DEBUG IT! 2.3** http://scratch.mit.edu/projects/24268506

This project is programmed to draw a happy face but something is not quite right! The pen continues to draw from one of the eyes to the smile when it should not be doing so. How do we fix the program?

❑ **DEBUG IT! 2.4** http://scratch.mit.edu/projects/23267140

In this project, when the green flag is clicked an animation of a flower growing begins and stops once it has fully bloomed. But something is not quite right! Instead of stopping when all the petals have bloomed, the animation starts all over. How do we fix this program?

❑ **DEBUG IT! 2.5** http://scratch.mit.edu/projects/23267245

In this project, the Happy Birthday song starts playing when the green flag is clicked. Once the song finishes, instructions should appear telling us to "click on me to blow out the candles!" But something is not working! The instructions to blow out the candles are shown while the birthday song is playing rather than after it finishes. How do we fix this program?

FEELING STUCK?

THAT'S OKAY! TRY THESE THINGS...

- ❑ Make a list of possible bugs in the program.
- ❑ Keep track of your work! This can be a useful reminder of what you have already tried and point you toward what to try next.
- ❑ Share and compare your problem finding and problem solving approaches with a neighbor until you find something that works for you!

FINISHED?

- + Add code commentary by right clicking on blocks in your scripts. This can help others understand different parts of your program!
- + Discuss your testing and debugging practices with a partner – make notes of the similarities and differences in your strategies.
- + Help a neighbor!

UNIT 2 ACTIVITY

MUSIC VIDEO

SUGGESTED TIME
45-60 MINUTES

OBJECTIVES

By completing this activity, students will:

+ be able to create a project that combines animation and music by working on a self-directed music video project
+ gain more familiarity with sprites, costumes, and sounds

ACTIVITY DESCRIPTION

- ❑ Introduce students to the idea of creating a music video in Scratch that combines music with animation. Optionally, show a few project examples from the Music Video studio.
- ❑ Give students open-ended time to work on their projects, with the Music Video handout available to provide guidance and inspiration. Encourage students to give credit on the project page for using others' ideas, music, or code.
- ❑ Help students give and receive feedback while developing their projects. We suggest checking in with a neighbor: have students stop midway and share their works-in-progress with one other person or in their critique groups (see the Unit 0 Critique Group activity) to ask for feedback. Optionally, invite students to add their projects to the Music Video studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Music Video handout
- ❑ Music Video studio
  http://scratch.mit.edu/studios/475517

REFLECTION PROMPTS

+ What was a challenge you overcame? How did you overcome it?
+ What is something you still want to figure out?
+ How did you give credit for ideas, music, or code that you borrowed to use in your project?

REVIEWING STUDENT WORK

+ Did the projects combine sprites and sound?
+ What parts of the projects did students choose to animate?
+ Are there certain blocks or concepts introduced up until now that students might still be struggling with? How might you help?

NOTES

+ To further personalize projects, help students include a favorite song or record themselves singing or playing an instrument, using features under the Sounds tab.
+ Questions about remixing and plagiarism may arise during this activity. Take this opportunity to facilitate a discussion about giving credit and attribution using the Scratch FAQ about remixing: http://scratch.mit.edu/help/faq/#remix

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

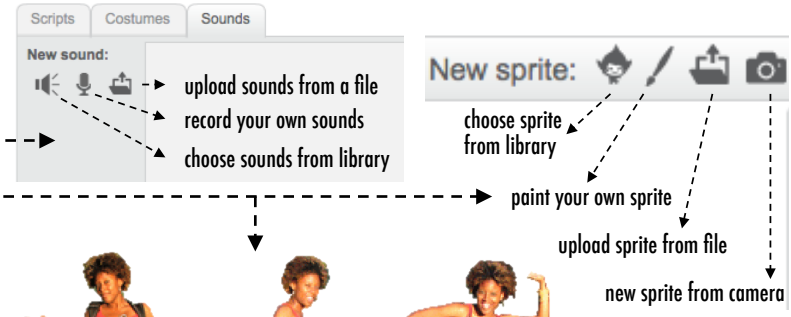
MUSIC VIDEO

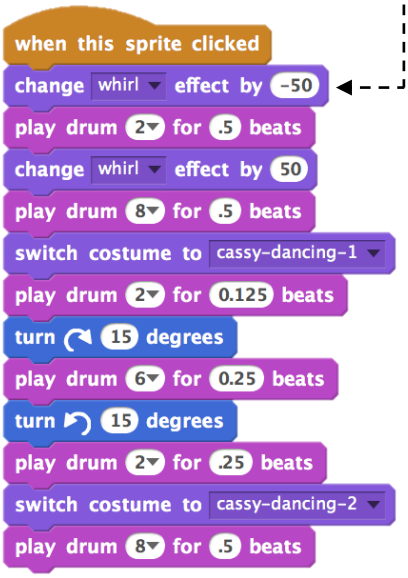
HOW CAN YOU COMBINE ANIMATION WITH MUSIC TO CREATE YOUR OWN SCRATCH-INSPIRED MUSIC VIDEO?

In this project, you will explore ideas related to theatre, song, dance, music, drawing, illustration, photography, and animation to create a personalized music video!

START HERE

- ❑ Add sound.
- ❑ Create and animate a sprite.
- ❑ Make them interact together!





THINGS TO TRY

- ❑ Use costumes to help bring your animations to life!
- ❑ Make your sprite interactive by adding scripts that have the sprite respond to clicks, key presses, and more.
- ❑ Add instructions on the project page to explain how people can interact with your program.

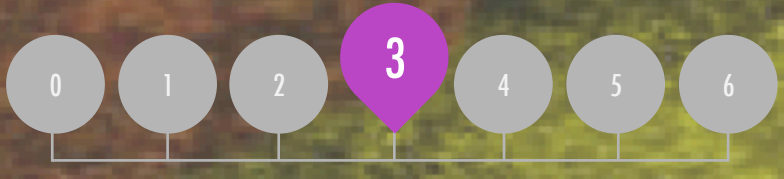
BLOCKS TO PLAY WITH

FINISHED?

+ Add your project to the Music Video studio: http://scratch.mit.edu/studios/475517
+ Be sure to give credit to any music, code, or other work used in your project.
+ Challenge yourself to do more! Create your own sprites, sounds, or costumes!

UNIT 3 STORIES

YOU ARE HERE



WHAT'S INCLUDED

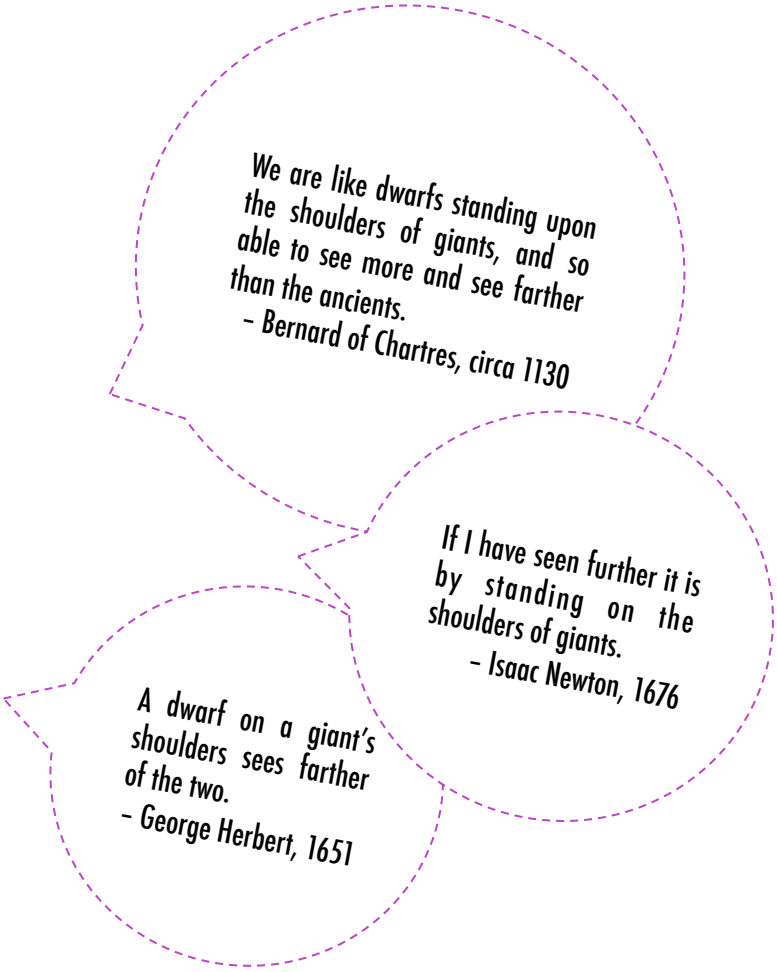
UNIT 3 OVERVIEW

THE "BIG IDEA"

In the introduction to his doctoral dissertation exploring remix culture, Andres Monroy-Hernandez (the lead designer of the initial version of the Scratch online community) included three quotes:

Building on other people's work has been a longstanding practice in programming, and has only been amplified by network technologies that provide access to a wide range of other people's work. An important goal of creative computing is to support connections between learners through reusing and remixing. The Scratch authoring environment and online community can support young designers in this key computational practice by helping them find ideas and code to build upon, enabling them to create more complex projects than they could have created on their own.

The activities in this unit offer initial ideas and strategies for cultivating a culture that supports reusing and remixing. How can you further support sharing and connecting?



LEARNING OBJECTIVES

Students will:

+ gain familiarity in and build understandings of the benefits of reusing and remixing while designing
+ develop greater fluency with computational concepts (events and parallelism) and practices (experimenting and iterating, testing and debugging, reusing and remixing)
+ explore computational creation within the genre of stories by designing collaborative narratives

KEY WORDS, CONCEPTS, & PRACTICES

+ reusing and remixing
+ make a block
+ backpack
+ stage
+ pass-it-on story
+ pair programming
+ scratch screening
+ design demo

NOTES

+ Reusing and remixing support the development of critical code-reading capacities and provoke important questions about ownership and authorship. Consider different strategies for how you might facilitate, discuss, and assess cooperative and collaborative work.

CHOOSE YOUR OWN ADVENTURE

This unit focuses on helping students develop their storytelling and remixing abilities through a variety of hands-on and off-computer design activities, providing opportunities for students to work collaboratively and build on the creative work of others. Building on initial experiences from Unit 2, the activities in this unit are designed to help students develop deeper fluency in the computational concepts of events and parallelism and the computational practices of experimenting and iterating and reusing and remixing. Each capacity-building activity is designed to help students build up storytelling projects by discovering new blocks and methods for programming interactions between sprites and backdrops, culminating in a Pass It On project.

POSSIBLE PATH

SESSION 1 → **CHARACTERS**

Create your own Scratch blocks using Make a Block.

SESSION 2 → **CONVERSATIONS**

How do you coordinate interactions between sprites using timing and broadcasting?

SESSION 3 → **SCENES**

What's the difference between the Stage and sprites?

SESSION 4 → **DEBUG IT!**

Help! Can you debug these five Scratch programs?

SESSION 5 → **CREATURE CONSTRUCTION**

What can we create by building on others' work?

SESSION 5 → **PASS IT ON**

What can we create by building on others' work?

UNIT 3 ACTIVITY

CHARACTERS

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ experiment with defining behaviors for characters using Scratch's Make a Block feature
+ gain more familiarity with the computational concepts of events and parallelism and the practice of experimenting and iterating

ACTIVITY DESCRIPTION

- ❑ Optionally, show example projects from the Characters studio and have the Characters handout available to guide students.
- ❑ Give students time to create their own Scratch blocks using the Make a Block feature found in the More Blocks category. Help them design two sprites or "characters" that each have two behaviors. Optionally, conduct a walkthrough of the Make a Block feature together as a class.
- ❑ Allow students to share their characters and behaviors with one another. We suggest the design demo activity: invite a few students to present their work to the class and demonstrate how they implemented the Make a Block feature. Optionally, have students add their projects to the Characters studio or a separate class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Characters handout
- ❑ Characters studio
  http://scratch.mit.edu/studios/475545

REFLECTION PROMPTS

+ How would you explain Make a Block to someone else?
+ When might you use Make a Block?

REVIEWING STUDENT WORK

+ Do projects include two sprites that each have two behaviors using the Make a Block feature?
+ Can students explain how to use the Make a Block feature to each other and to you?

NOTES

+ If students are struggling with figuring out how to use the Make a Block feature, invite them to explore how others implemented the feature by investigating the code of projects in the Characters studio.
+ Learn more about the Make a Block feature in this video tutorial: http://bit.ly/makeablock

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

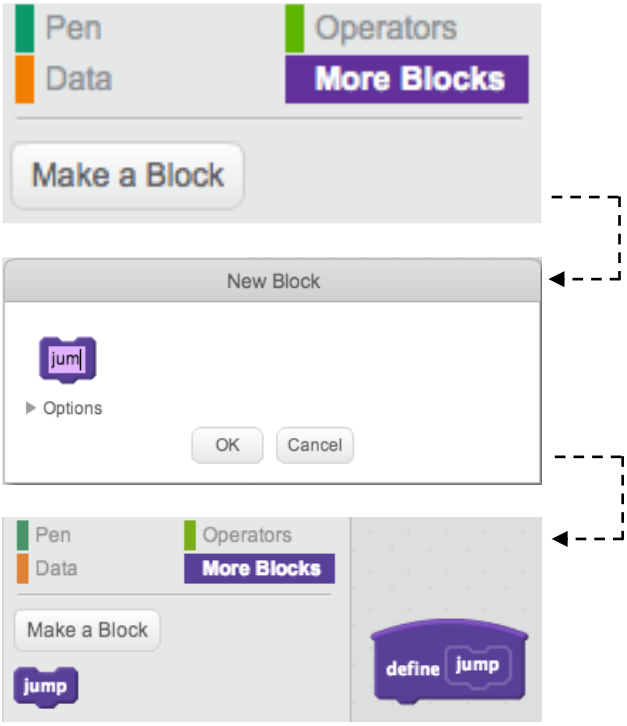
CHARACTERS

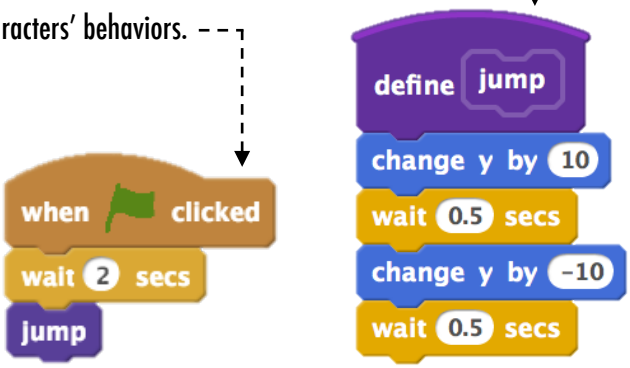
DO YOU WANT TO CREATE YOUR OWN SCRATCH BLOCKS?

Experiment with the Make a Block feature in Scratch! In this project, you will create your own blocks that define two behaviors for two different characters.

START HERE

- ❑ Choose from the library, paint, or upload two sprite characters.
- ❑ Click on the Make a Block button in the More Blocks category to create and name your block.
- ❑ Add blocks under the Define block to control what your custom block will do.
- ❑ Experiment with using your block to program your characters' behaviors.
- ❑ Repeat!





THINGS TO TRY

- ❑ Feeling stuck? That's okay! Check out this video to get started with the Make a Block feature: http://bit.ly/makeablock
- ❑ Explore other projects in the Characters Studio to see what new blocks others have created.
- ❑ Sometimes there can be more than one way of defining the same behavior. Experiment with different block combinations to try out multiple options and outcomes.

FINISHED?

- + Add your project to the Characters Studio: http://scratch.mit.edu/studios/475545
- + Challenge yourself to do more! Experiment with adding different characters and behaviors using the Make a Block feature.
- + Help a neighbor!

UNIT 3 ACTIVITY

CONVERSATIONS

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ explore two different strategies for synchronizing interactions between sprites (timing and broadcasting) by remixing a joke project
+ develop greater familiarity with the computational concept of events and parallelism and the practice of reusing and remixing

ACTIVITY DESCRIPTION

- ❑ Optionally, explore the Penguin Joke starter project as a group and have the Conversations handout available to guide students.
- ❑ Invite students to see inside the Penguin Joke starter project to observe how the conversation is animated using wait blocks. Have students use the remix function and redesign the Penguin Joke project to coordinate the conversation using the broadcast, broadcast and wait, and when I receive blocks.
- ❑ Encourage students to share their joke projects with one another. We suggest the design demo activity: invite a few students to present their work to the class and demonstrate how they implemented broadcast. Optionally, have students add their projects to the Conversations studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Conversations handout
- ❑ Penguin Joke starter project
  http://scratch.mit.edu/projects/10015800
- ❑ Conversations studio
  http://scratch.mit.edu/studios/475547

REFLECTION PROMPTS

+ How would you describe broadcast to someone else?
+ When would you use timing in a project? When would you use broadcasting?

REVIEWING STUDENT WORK

+ Do projects use the broadcast and when I receive blocks?
+ Can students explain how to use the broadcast, broadcast and wait, and when I receive blocks?

NOTES

+ If students are having trouble understanding how to use the broadcast and when I receive block pair, invite them to explore the code of example projects in the Broadcast Examples studio: http://scratch.mit.edu/studios/202853

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

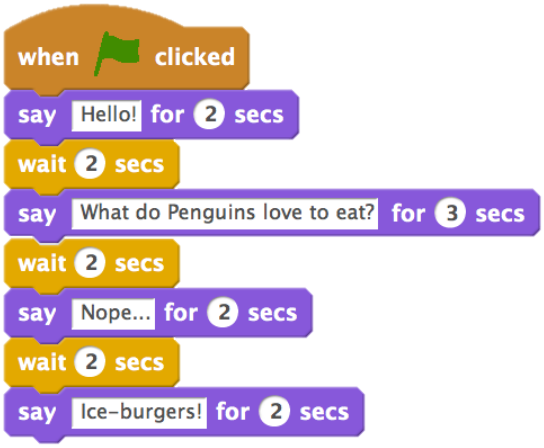
CONVERSATIONS

WHAT ARE DIFFERENT WAYS TO COORDINATE INTERACTIONS BETWEEN SPRITES?

In this activity, you'll explore different ways to program sprites to have conversations! Experiment with timing and explore using broadcast by remixing a joke project.

START HERE

- ❑ Look inside the Penguin Jokes project: http://scratch.mit.edu/projects/10015800
- ❑ Investigate the code to see how the wait and say blocks are used to coordinate the conversation.
- ❑ Remix the project to use the broadcast and when I receive blocks instead of wait blocks.



FEELING STUCK?

THAT'S OKAY! TRY THESE THINGS...

- ❑ Brainstorm ideas with a neighbor! Generate a list of possible solutions and test them out together.
- ❑ Try using the broadcast and when I receive blocks in different parts of your project.
- ❑ Explore projects in the Conversations studio to get inspiration for different ways to coordinate conversations between sprites.

FINISHED?

- + Add your project to the Conversations studio: http://scratch.mit.edu/studios/475547
- + Challenge yourself to do more! Add other characters and conversations.
- + Share your project with a neighbor and walk them through your process of exploration and design.
- + Help a neighbor!

UNIT 3 ACTIVITY

SCENES

SUGGESTED TIME
30–45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ be able to create a project that experiments with changing backdrops, like a story with multiple scenes or a slideshow
+ gain more familiarity with the computational concepts of events and parallelism and the practice of experimenting and iterating

ACTIVITY DESCRIPTION

- ❑ Optionally, show example projects from the Scenes studio and have the Scenes handout available to guide students.
- ❑ Give students time to develop a project that includes multiple scene changes using different backdrops, such as in a slideshow. Challenge students to explore and manipulate scripts in the Stage to initiate backdrop changes.
- ❑ Allow students to share their projects with one another. We suggest the design demo activity: invite a few students to present their work to the class and demonstrate how they implemented switching backdrops. Optionally, have students add their projects to the Scenes studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Scenes handout
- ❑ Scenes studio
  http://scratch.mit.edu/studios/475550

REFLECTION PROMPTS

+ What does the Stage have in common with sprites?
+ How is the Stage different from sprites?
+ How do you initiate a sprite's actions in a scene?
+ What other types of projects (beyond animations) use scene changes?

REVIEWING STUDENT WORK

+ Do projects successfully coordinate multiple scenes using changing backdrops?

NOTES

+ If students are having trouble figuring out how to switch backdrops, encourage them to tinker with blocks under the Looks category, especially the switch backdrop to, switch backdrop to and wait, and next backdrop blocks.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

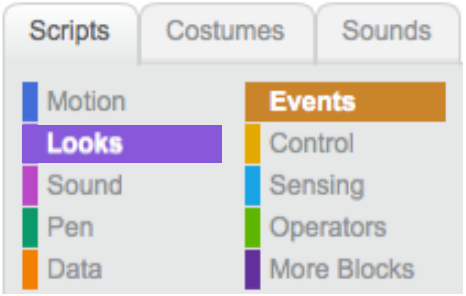
SCENES

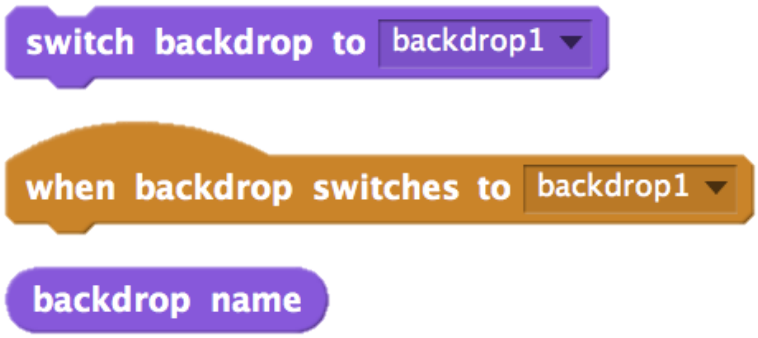
WHAT IS THE DIFFERENCE BETWEEN THE STAGE AND SPRITES?

In this activity, you will create a project that experiments with backdrops, like a story with multiple scenes or a slideshow.

START HERE

- ❑ Choose from the library, paint, or upload multiple backdrops into your project.
- ❑ Experiment with blocks from the Looks and Events categories to initiate switching backdrops.
- ❑ Add scripts to the stage and sprites to coordinate what happens when the backdrop changes in your project!





THINGS TO TRY

- ❑ Look for blocks under the sprites and the stage related to backdrop and test them out to see what they do!
- ❑ Need more inspiration? Explore the Scratch online community to discover projects that use multiple backdrops.

FINISHED?

- + Add your project to the Scenes Studio: http://scratch.mit.edu/studios/475550
- + Challenge yourself to do more! Add more backdrop changes to your project.
- + Help a neighbor!
- + Return to one of your previous projects or find a project you are inspired by and remix it by adding switching backdrops.

UNIT 3 ACTIVITY

DEBUG IT!

SUGGESTED TIME
15–30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ investigate the problem and find a solution to five debugging challenges
+ explore a range of concepts (including events and parallelism) through the practices of testing and debugging

ACTIVITY DESCRIPTION

- ❑ Optionally, have the Unit 3 Debug It! handout available to guide students during the activity.
- ❑ Help students open the Debug It! programs from the Unit 3 Debug It! studio or by following the project links listed on the Unit 3 Debug It! handout. Encourage students to click on the "Look Inside" button to investigate the buggy program, tinker with problematic code, and test possible solutions.
- ❑ Give students time to test and debug each Debug It! challenge. Optionally, have students use the remix function in Scratch to fix the bugs and save corrected programs.
- ❑ Ask students to reflect back on their testing and debugging experiences by responding to the reflection prompts in their design journals or in a group discussion.
- ❑ Create a class list of debugging strategies by collecting students' problem finding and problem solving approaches.

RESOURCES

- ❑ Unit 3 Debug It! handout
- ❑ Unit 3 Debug It! studio
  http://scratch.mit.edu/studios/475554

REFLECTION PROMPTS

+ What was the problem?
+ How did you identify the problem?
+ How did you fix the problem?
+ Did others have alternative approaches to fixing the problem?

REVIEWING STUDENT WORK

+ Were students able to solve all five bugs? If not, how might you clarify the concepts expressed in the unsolved programs?
+ What different testing and debugging strategies did students employ?

NOTES

+ Being able to read others' code is a valuable skill and is critical for being able to engage in the practices of reusing and remixing.
+ This activity is a great opportunity for pair programming. Divide students into pairs to work on the debugging challenges.
+ Students can explain their code revisions by right-clicking on Scratch blocks to insert code comments.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

DEBUG IT!

HELP! CAN YOU DEBUG THESE FIVE SCRATCH PROGRAMS?

In this activity, you will investigate what is going awry and find a solution for each of the five Debug It! challenges.

START HERE

- ❑ Go to the Unit 3 Debug It! Studio: http://scratch.mit.edu/studios/475554
- ❑ Test and debug each of the five debugging challenges in the studio.
- ❑ Write down your solution or remix the buggy program with your solution.

❑ **DEBUG IT! 3.1** http://scratch.mit.edu/projects/24269007

In this project, the Scratch Cat teaches Gobo to meow. But when it's Gobo's turn to try – Gobo stays silent. How do we fix the program?

❑ **DEBUG IT! 3.2** http://scratch.mit.edu/projects/24269046

In this project, the Scratch Cat is supposed to count from 1 to the number the user provides. But the Scratch Cat always counts to 10. How do we fix the program?

❑ **DEBUG IT! 3.3** http://scratch.mit.edu/projects/24269070

In this project, the Scratch Cat is doing roll call with Gobo's friends: Giga, Nano, Pico, and Tera. But everything is happening all at once! How do we fix the program?

❑ **DEBUG IT! 3.4** http://scratch.mit.edu/projects/24269097

In this project, the Scratch Cat and Gobo are practicing their jumping routine. When Scratch Cat says "Jump!", Gobo should jump up and down. But Gobo isn't jumping. How do we fix the program?

❑ **DEBUG IT! 3.5** http://scratch.mit.edu/projects/24269131

In this project, the scene changes when you press the right arrow key. The star of the project – a dinosaur – should be hidden in every scene except when the scene transitions to the auditorium backdrop. In the auditorium, the dinosaur should appear and do a dance. But the dinosaur is always present and is not dancing at the right time. How do we fix the program?

FEELING STUCK?

THAT'S OKAY! TRY THESE THINGS...

- ❑ Make a list of possible bugs in the program.
- ❑ Keep track of your work! This can be a useful reminder of what you have already tried and point you toward what to try next.
- ❑ Share and compare your problem finding and problem solving approaches with a neighbor until you find something that works for you!

FINISHED?

- + Add code commentary by right clicking on blocks in your scripts. This can help others understand different parts of your program!
- + Discuss your testing and debugging practices with a partner, and make note of the similarities and differences in your strategies.
- + Help a neighbor!

UNIT 3 ACTIVITY

CREATURE CONSTRUCTION

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ be introduced to the computational practice of reusing and remixing by contributing to a collaborative drawing

ACTIVITY DESCRIPTION

- ❑ In this activity, students will draw a "creature" in three parts.
- ❑ Give each student a tri-folded sheet of blank paper and one minute to draw a "head" for their creature. Next, have them fold the paper over so that the head is hidden, with little prompts for where to continue the drawing. After the head is hidden, students will pass the creature to another student. Then, give students one minute to draw a "middle" for their creature, using the guides from the head, but without peeking! After the middles are hidden (and prompts drawn), pass the creatures. Finally, give students one minute to draw a "bottom" for their creature. When finished, unfold the papers to reveal the collaboratively constructed creatures!
- ❑ Post drawings on a wall or board and let students explore the outcome of their creative contributions.
- ❑ Facilitate a group discussion about co-authorship, collaboration, and reusing and remixing work.

RESOURCES

- ❑ blank paper (approximately 8.5" by 11"), folded into thirds
- ❑ things to sketch with (pencils, pens, markers, etc.)

REFLECTION PROMPTS

+ What is your definition of remixing?
+ Think about the creature you started (drew the "head" for). How did your ideas become extended or enhanced by others' contributions?
+ Considering the creatures you extended (drew the "middle" or "bottom" sections for), how did your contributions extend or enhance others' ideas?

REVIEWING STUDENT WORK

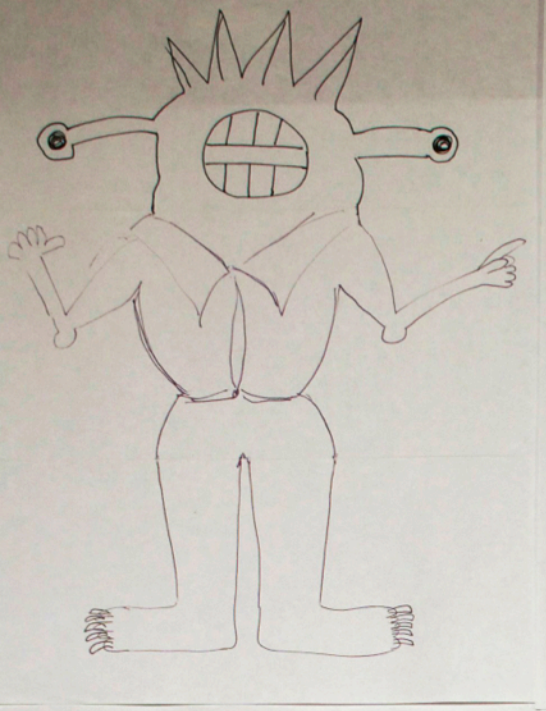
+ Can students explain remixing and its benefits?

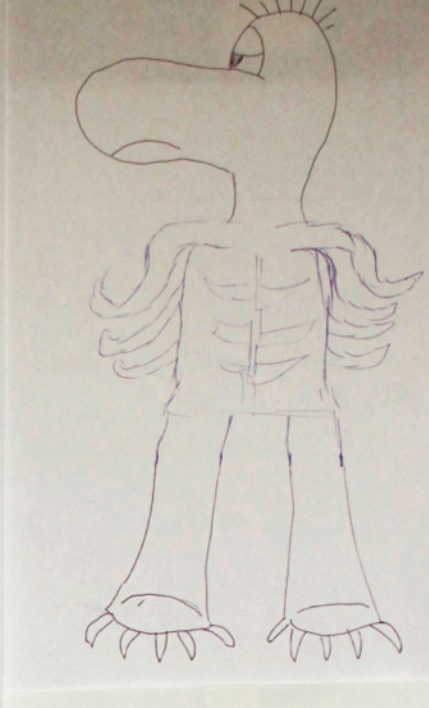
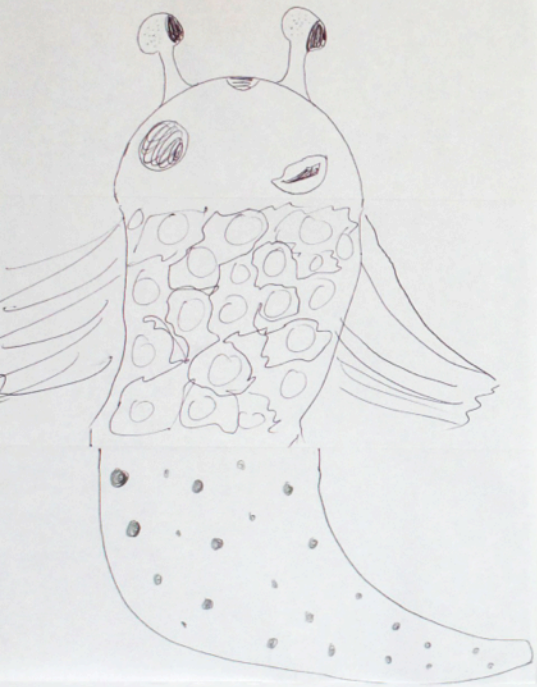
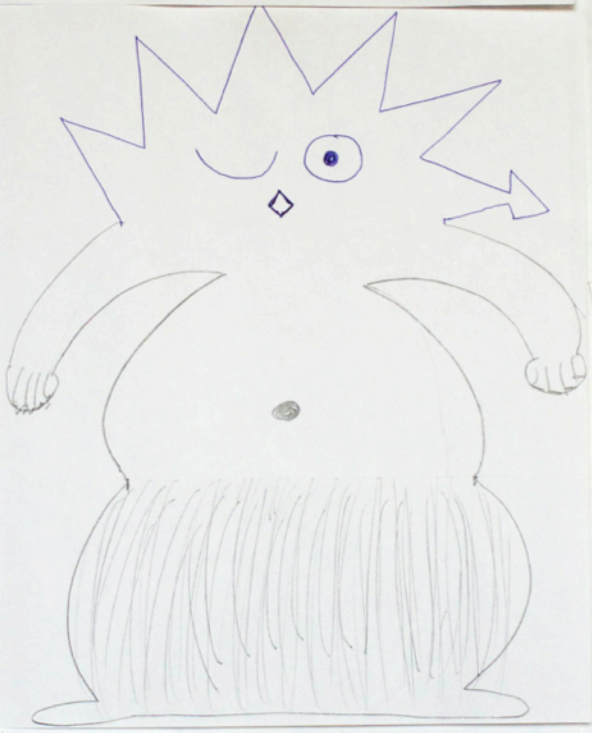
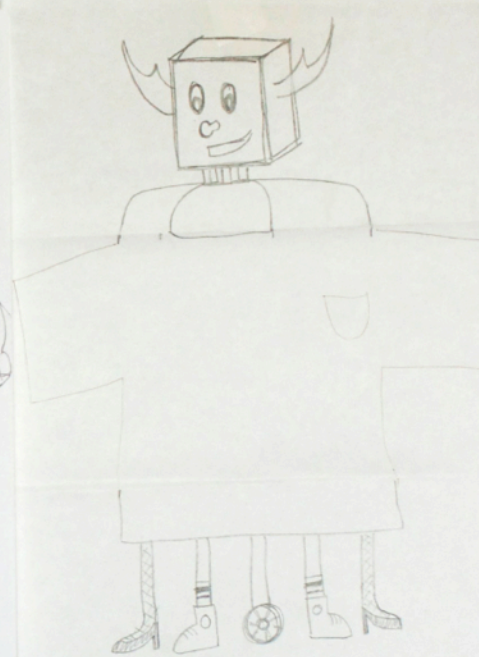
NOTES

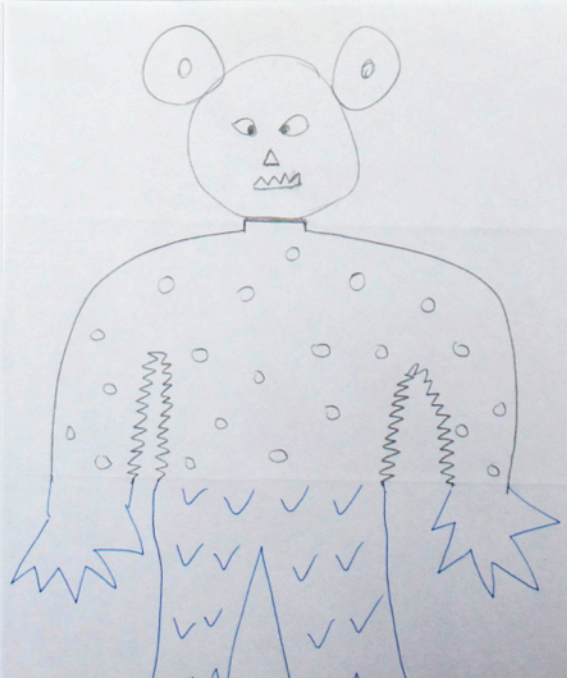
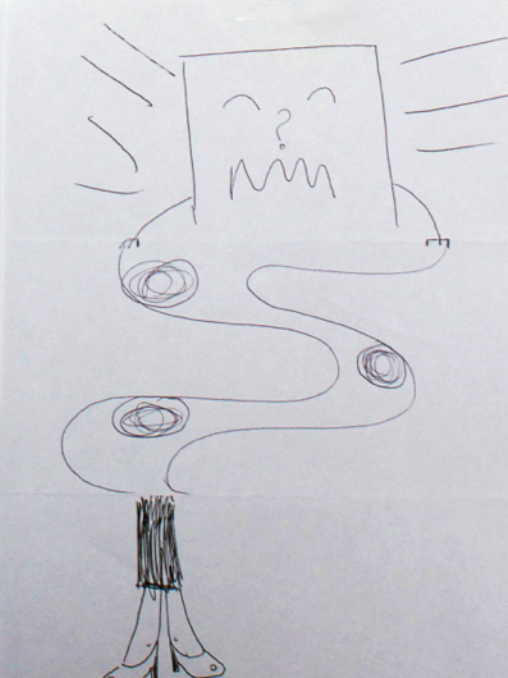
+ This activity is a perfect warm-up activity for the Pass It On project! We recommend facilitating Creature Construction directly before Pass It On.
+ Optionally, have students sign their names at the bottom of each creature drawing they worked on to identify the contributing artists.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....



UNIT 3 ACTIVITY

PASS IT ON

SUGGESTED TIME
45-60 MINUTES

OBJECTIVES

By completing this activity, students will:

+ be able to create a Scratch project that tells a story by reusing and remixing the work of others
+ experience pair programming by working in pairs to develop a collaborative storytelling project

ACTIVITY DESCRIPTION

- ❑ Divide the group into pairs. Introduce students to the concept of a pass-it-on-story, a Scratch project that is started by a pair of people, and then passed on to two other pairs to extend and reimagine. Optionally, print out the Pass It On handout.
- ❑ Encourage students to start in whatever way they want – focusing on characters, scene, plot, or whatever element excites them. Give each pair 10 minutes to work on their collaborative story before having them rotate to extend another story by remixing the project. Encourage students to give credit for reusing or remixing content.
- ❑ After two rotations, allow students to revisit story projects with their contributions. We suggest hosting a Scratch screening: with projector and screen, present the story projects with students gathered around to watch. Optionally, invite students to add their projects to the Pass It On studio or a class studio.
- ❑ Ask students to respond to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Pass It On handout
- ❑ Pass It On studio
  http://scratch.mit.edu/studios/475543
- ❑ Projector and screen to present student work (optional)

REFLECTION PROMPTS

+ How did it feel to remix and build on others' work? How did it feel to be remixed?
+ Where else in your life have you seen or experienced reusing and remixing? Share two examples.
+ How was working with someone else different from your prior experiences of designing your Scratch projects?

REVIEWING STUDENT WORK

+ What parts of projects did students contribute to?
+ Do students seem comfortable with the concepts of events and parallelism and practices of reusing and remixing? If not, in what ways can these be further clarified?

NOTES

+ Consider organizing your Scratch screening as an event! Invite students from other classes to the viewing, offer snacks and drinks, or host the event in an auditorium or room with a large wall or screen for displaying projects.
+ Introduce students to the backpack (located at the bottom of the Scratch project editor) as another way to remix projects. Learn more about this tool in the Backpack video tutorial: http://bit.ly/scratchbackpack

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

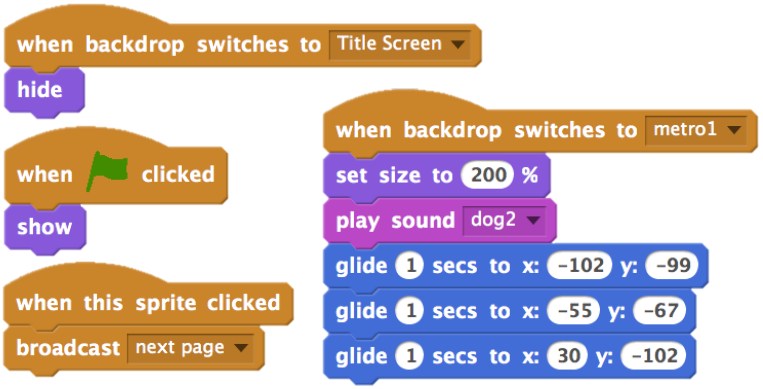
PASS IT ON

WHAT CAN WE CREATE BY BUILDING ON OTHERS' WORK?

In this project, you will start developing an animated story project, and then you will pass the story on to others to remix, extend, or reimagine!

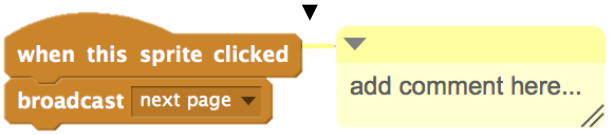
START HERE

- ❑ Work on a story project that focuses on characters, scene, plot, or whatever element excites you.
- ❑ After 10 minutes, save and share your project online.
- ❑ Rotate & extend another story project by remixing it.
- ❑ Repeat!

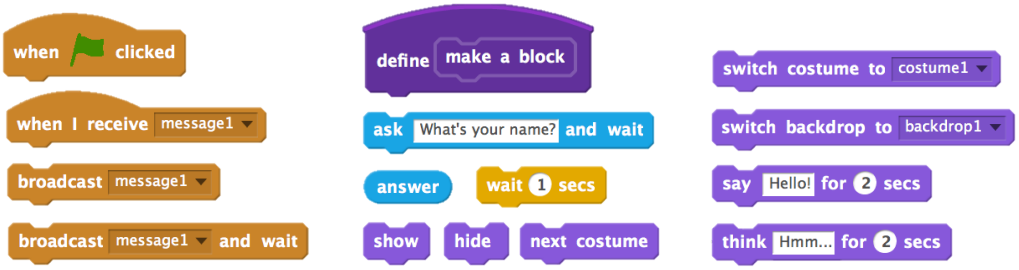


THINGS TO TRY

- ❑ Brainstorm different possibilities for remixing, extending, or reimagining a story. Do you want to add a new scene to the end? Could you imagine what happens before the story begins? What if a new character was added? How about inserting a plot twist? What else?
- ❑ Adding comments in your code can help others understand different parts of your program. To attach a comment to a script, right click on a block and add a description.



BLOCKS TO PLAY WITH

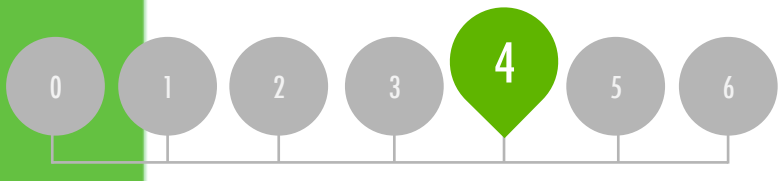


FINISHED?

+ Add your project to the Pass It On studio: http://scratch.mit.edu/studios/475543
+ Help a neighbor!
+ Return to all the projects you contributed to and check out how the stories evolved!

UNIT 4
GAMES

YOU ARE HERE



WHAT'S INCLUDED

UNIT 4 OVERVIEW

THE "BIG IDEA"

Personalization is an important guiding principle in the design of the creative computing experience. By "personalization", we mean both connecting to personal interests and acknowledging that personal interests can vary considerably. There are many ways of knowing and doing – and exploring these multiple ways can help support interest, motivation, and persistence among young learners. In this unit, learners explore some of the advanced concepts and challenging problems associated with game design. An advanced concept or challenging problem can be made more accessible if rooted in activities that are personally meaningful. As an example of the power of context, we turn to a story shared by Mitch Resnick – the director of the Scratch project at MIT.

A few years ago I was at one of our Computer Clubhouse after school centers and I saw a 13-year-old boy working on creating his own game. He was able to control a character, in this case, a fish. He wanted the game to keep track of the score, so you could see how many little fish had been eaten by the big fish, but he didn't know how.

I saw this as an opportunity to introduce the idea of variables. I showed this to him and he immediately saw how he could use this block to keep track of how many fish had been eaten in his game. He took the block and put it in the script right where the big fish eats the little fish. He quickly tried it. Sure enough, every time the big fish ate a little fish, the score goes up by 1.

I think that he really got a deep understanding of variables because he really wanted to make use of it. That's one of our overall goals of Scratch. It's not just about variables, but for all types of concepts. We see that kids get a much deeper understanding of the concepts they learn when they are making use of the concepts in a meaningful and motivating way.

LEARNING OBJECTIVES

Students will:

+ be introduced to the computational concepts of conditionals, operators, and data (variables and lists)
+ become more familiar with the computational practices of experimenting and iterating, testing and debugging, reusing and remixing, and abstracting and modularizing by building and extending a self-directed maze, pong, or scrolling game project
+ identify and understand common game mechanics

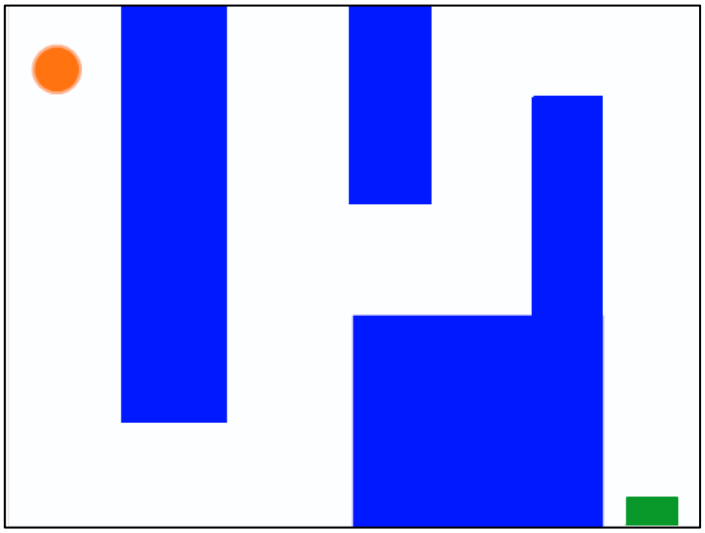
KEY WORDS, CONCEPTS, & PRACTICES

+ abstracting and modularizing
+ conditionals
+ operators
+ data
+ variables and lists
+ sensing
+ feedback fair
+ arcade day
+ puzzle jar
+ brain dump

NOTES

+ Many new concepts are explored in this unit, so we've included added support in the form of example project studios, new programming puzzles for extra practice, and starter game projects that we encourage you to remix and reuse as needed.

CHOOSE YOUR OWN ADVENTURE



In this unit, learners will become game designers and experience creating their own game project. Guided by the activities in this unit, students will be introduced to game mechanics and game development while building understandings of computational concepts (conditionals, operators, data) and computational practices (abstracting and modularizing).

You could get students started on their game projects with the Starter Games activity and then support further development through other activities. From learning common game mechanics such as keeping score and side-scrolling, to the creation of multiplayer games (e.g., Pong), Unit 4 activities offer students multiple opportunities to practice game development.

POSSIBLE PATH

SESSION 1
DREAM GAME LIST
What do all games have in common?

SESSIONS 1 - 5
STARTER GAMES
How can you use Scratch to build an interactive game?

SESSION 2
SCORE
How can you add score to a game using variables?

SESSION 3
EXTENSIONS
What are different ways of extending and increasing difficulty in a game?

SESSION 4
INTERACTIONS
Tackle nine Scratch programming puzzles.

SESSION 5
DEBUG IT!
Help! Can you debug these five Scratch programs?

UNIT 4 ACTIVITY

DREAM GAME LIST

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES
By completing this activity, students will:
+ identify common design elements of games

ACTIVITY DESCRIPTION

- ❑ Divide students into small groups of 2-3 people.
- ❑ In their small groups, ask students to generate a list of games that they enjoy playing. They can compose the list using their design journals or a sheet of paper. We suggest facilitating the brain dump brainstorming activity: give students a short time period (1-2 minutes) to write down as many games as they can. Then, have students narrow down their favorites from the brain dump list.
- ❑ After a few minutes, ask groups about their list of games:
  What do the games have in common?
  What features of their design make them a game?
- ❑ Facilitate a class discussion about what characteristics make up a game and generate a class list of common game mechanics. Next, ask students to imagine their dream game and write a list of design elements for that game.
- ❑ Invite students to share their dream game lists in their small groups or critique groups (see Unit 0 Critique Group activity) to get feedback and suggestions.

RESOURCES

- ❑ paper to write down game design elements (optional)
- ❑ things to sketch with (pencils, pens, markers, etc.)

REFLECTION PROMPTS

+ Make a list of your favorite games.
+ What do the games have in common?
+ What features of their design make them a game?
+ Create a list of design elements for your dream game.

REVIEWING STUDENT WORK

+ Do the dream game lists include features of games?
+ What design elements are similar or different from the class group list?
+ What do the lists tell you about the kinds of games and the types of play your students enjoy?

NOTES

+ Invite students to refer back to this dream game list while programming games in other Unit 4 activities.

NOTES TO SELF

- ❑ _____
- ❑ _____
- ❑ _____
- ❑ _____


Mario
monopoly
Chess
Candyland
football
Clue
Tennis
Baseball
Jump Rope
Pac Man
Flappy Bird
Wheel of Fortune
Four Square

UNIT 4 ACTIVITY

STARTER GAMES

SUGGESTED TIME
45-60 MINUTES

OBJECTIVES

By completing this activity, students will:

+ develop greater fluency with computational concepts (conditionals, operators, data) and practices (experimenting and iterating, testing and debugging, reusing and remixing, abstracting and modularizing) by working on a self-directed game project

ACTIVITY DESCRIPTION

- ❑ In this activity, students will create a starter game project that can be revisited and extended during the Score, Extensions, and Interactions activities. Optionally, show the Maze, Pong, and Scrolling example starter projects, and have the Maze, Pong, and Scrolling handouts available to guide students.
- ❑ Choose one game project to facilitate as a class or let students choose which game they want to create: maze, pong, or scrolling. Give students time to start building their games or let them remix one of the starter projects.
- ❑ Encourage students to get feedback on their games-in-progress. We suggest the feedback fair activity: half of the students stay in their seats with their projects open while the other half walks around exploring projects, asking questions, and giving feedback, then switch sides. Optionally, have students add their final game projects to the Games studio or a class studio.
- ❑ Ask students to respond to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Maze handout
- ❑ Maze example starter project
  http://scratch.mit.edu/projects/11414041
- ❑ Pong handout
- ❑ Pong example starter project
  http://scratch.mit.edu/projects/10128515
- ❑ Scrolling handout
- ❑ Scrolling example starter project
  http://scratch.mit.edu/projects/22162012
- ❑ Games studio
  http://scratch.mit.edu/studios/487504

REFLECTION PROMPTS

+ What was challenging about designing your game?
+ What are you proud of?

REVIEWING STUDENT WORK

+ Do games include conditionals, operators, and data?

NOTES

+ To celebrate and share final game creations, we recommend hosting an Arcade Day. Final game projects are placed in presentation mode; students walk around and play each other's games.
+ The Scrolling game option introduces cloning. Help students learn more about the cloning blocks with the Cloning handout from Unit 5 Advanced Features.

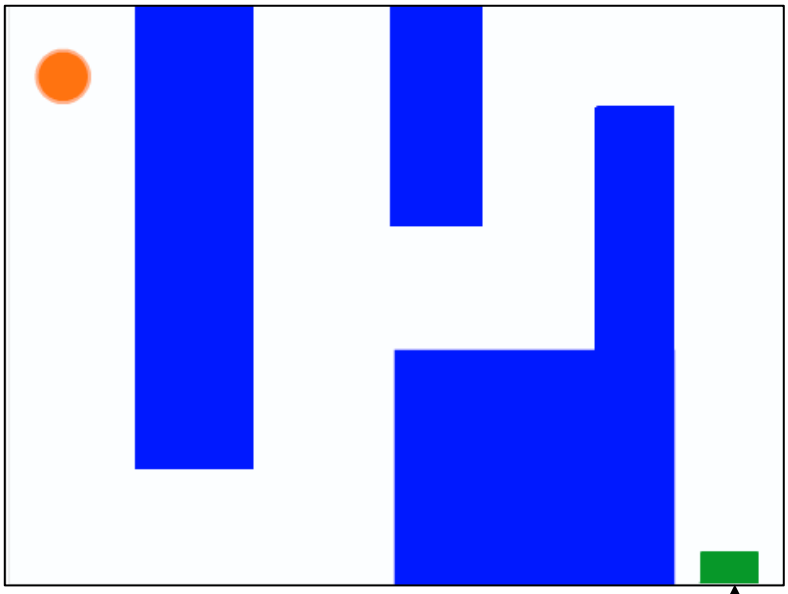
NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

MAZE

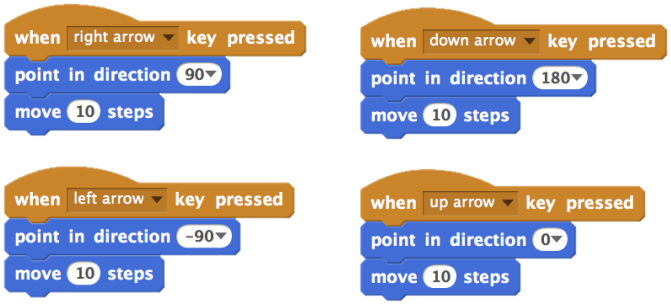
HOW CAN YOU USE SCRATCH TO BUILD AN INTERACTIVE GAME?

In this project, you will create a game. This game includes interactions between sprites, score, and levels. You move a sprite from the start of a maze to the end without touching the walls.



START HERE

- ❑ Draw a maze-like background and use different colors for the walls and end-of-maze marker.
- ❑ Add a sprite.
- ❑ Make your game interactive!



These scripts give the player control over sprite movement in the maze.

This tells your sprite where to begin and marks the start of the maze.

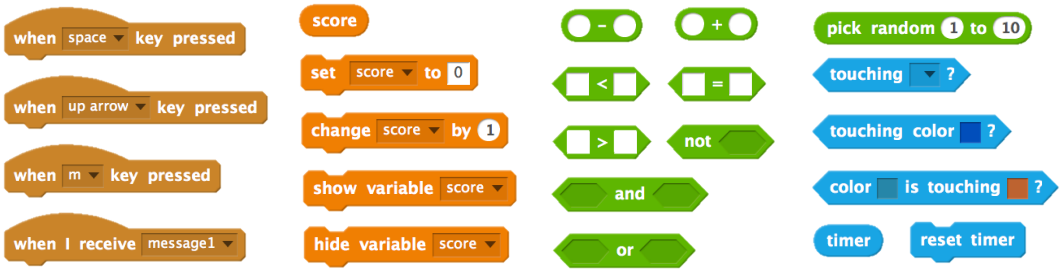
THINGS TO TRY

- ❑ Add multiple levels to your game! This can be done through the use of different backdrops and using braodcast blocks to trigger the next level.
- ❑ Use the make a variable block to keep score!
- ❑ Experiment with timer blocks to add new challenges to your maze!

This will cause your sprite to bounce off the blue walls of the maze.

This tells the end-of-maze sprite that players win when the ball touches this sprite.

BLOCKS TO PLAY WITH



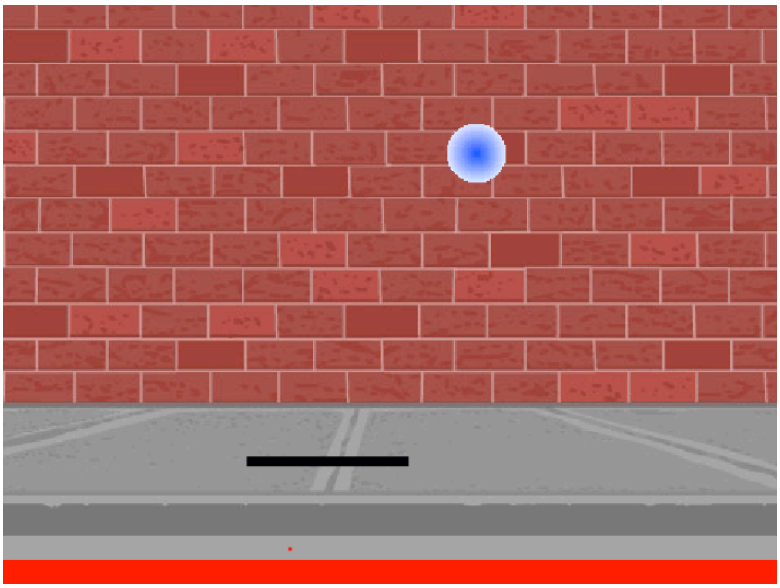
FINISHED?

+ Add your project to the Games Studio: http://scratch.mit.edu/studios/487504
+ Swap games with a partner and walk each other through your creations.

PONG

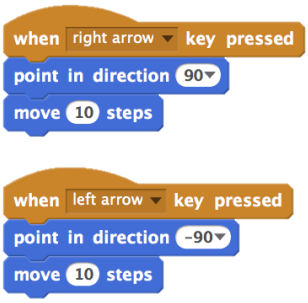
HOW CAN YOU USE SCRATCH TO BUILD AN INTERACTIVE GAME?

In this project, you will create a game. This game includes interactions between sprites, score, and levels. The game is similar to the classic game of pong, where the goal is to keep the sprite from getting past you.



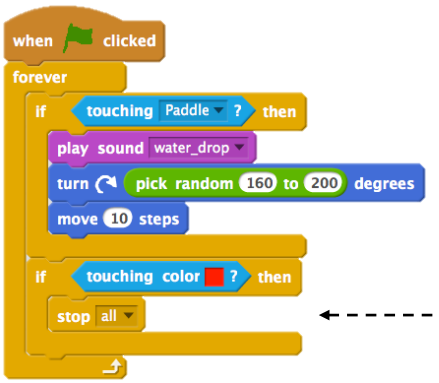
START HERE

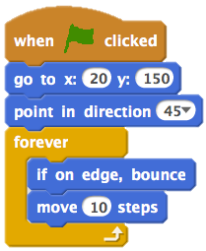
- ❑ Create two sprites: a paddle for the user to control and a ball the user will be playing with.
- ❑ Make your paddle sprite interactive.
- ❑ Bring your game to life!



THINGS TO TRY

- ❑ How do you add difficulty to your game? Creating different levels, using a timer, or keeping score are a few examples of things you could do.
- ❑ Experiment with changing the look of your game by editing the backdrops!
- ❑ Explore using different key presses to control your sprites!

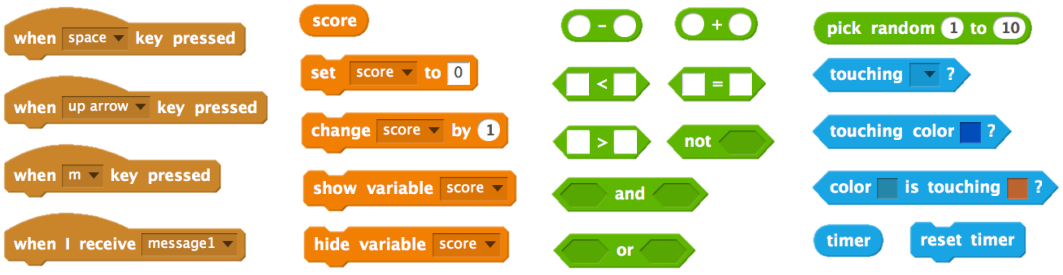




Interacts with the walls
Interacts with the paddle

These control the ball - if touching the paddle or a wall, it continues moving. If touching red (meaning the ball moved past the paddle) the game ends.

BLOCKS TO PLAY WITH



FINISHED?

+ Add your project to the Games Studio: http://scratch.mit.edu/studios/487504
+ Swap games with a partner and walk each other through your creations.

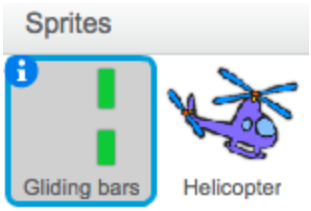
SCROLLING

HOW CAN YOU USE SCRATCH TO BUILD AN INTERACTIVE GAME?

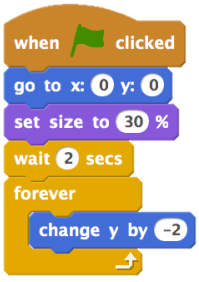
In this project, you will create a game. This game includes interactions between sprites, score, and levels. The game is similar to Flappy Bird, where the goal is to keep an object from falling to the ground or touching certain objects.

START HERE

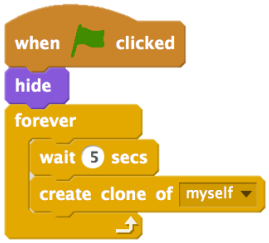
- ❑ Create two sprites: one for the player to control (helicopter) and one to avoid (gliding bars).
- ❑ Make the helicopter interactive.
- ❑ Bring your game to life by adding scripts to make the gliding bars scroll across the stage!



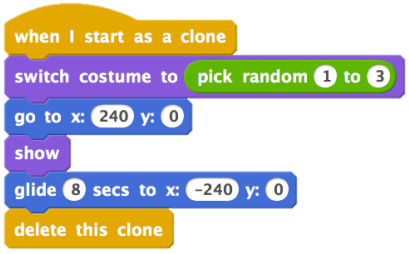
Controls sprite movement

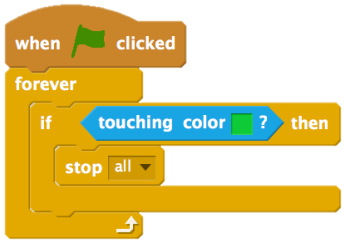


Causes sprite to constantly fall downward



This creates clones, which are used in the script below to make the bars scroll across the screen:



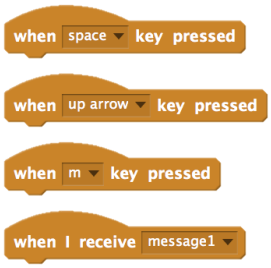


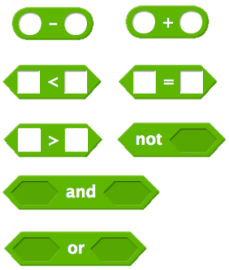
Specifies when the game ends

THINGS TO TRY

- ❑ How do you add difficulty to your game? Creating different levels, using a timer, or keeping score are a few examples of things you could do.
- ❑ Experiment with changing the look of your game by editing the backdrops!
- ❑ Explore using different key presses to control your sprites!

BLOCKS TO PLAY WITH





FINISHED?

+ Add your project to the Games Studio: http://scratch.mit.edu/studios/487504
+ Swap games with a partner and walk each other through your creations.

UNIT 4 ACTIVITY

SCORE

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ be able to describe what a variable is and why variables are useful
+ be introduced to the computational concept of data
+ experience remixing and reusing a project or part of a project

ACTIVITY DESCRIPTION

- ❑ Optionally, explore the Fish Chomp starter project as a group and have the Score handout available to guide students.
- ❑ Help students open the Fish Chomp starter project. Give students time to explore variables by remixing the Fish Chomp Starter Project to add score to the game. Optionally, give students time to incorporate score into their previously started maze, pong, or scrolling game projects.
- ❑ Allow students to share their Fish Chomp remixes or game projects with added score. We suggest the Design Demo activity: invite a few students to present their projects to the group and demonstrate how they implemented score using variables. Optionally, have students add their remixes to the Fish Chomp Remix studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Score handout
- ❑ Score examples studio
  http://scratch.mit.edu/studios/218313
- ❑ Fish Chomp starter project
  http://scratch.mit.edu/projects/10859244
- ❑ Fish Chomp remix studio
  http://scratch.mit.edu/studios/475615

REFLECTION PROMPTS

+ How would you explain variables to someone else?
+ What are variables good for?

REVIEWING STUDENT WORK

+ Can students explain what a variable is and what variables are good for?

NOTES

+ Encourage students to clarify their understanding of variables by exploring code from sample projects in the Score examples studio.
+ Variables are an important mathematical and computational concept. Students are taught about variables in their math and science classes, but many students have a difficult time learning them. Games are one way to make the usefulness of variables more concrete.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

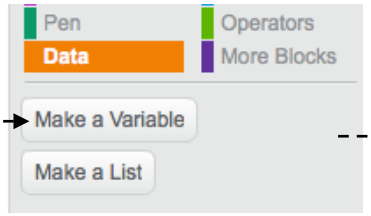
SCORE

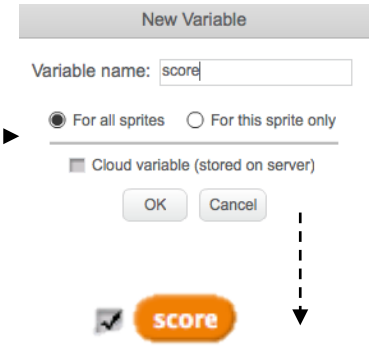
HOW CAN YOU KEEP SCORE IN A SCRATCH PROJECT?

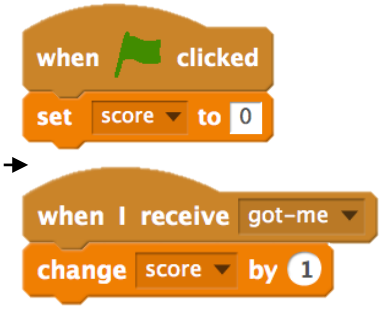
Fish Chomp is a game where players try to catch as many fish as they can by guiding a sprite with the mouse. In this activity, you will remix Fish Chomp by adding a score with variables.

START HERE

- ❑ Go to the Fish Chomp project page: http://scratch.mit.edu/projects/10859244
- ❑ Click on the Make a Variable button in the Data category to create and name a variable for score.
- ❑ Experiment with your new variable blocks to incorporate score into your project!







FEELING STUCK?

THAT'S OKAY! TRY THESE THINGS...

- ❑ Not sure how to work with variables? Check out this project for more information: http://scratch.mit.edu/projects/2042755
- ❑ Or take a look at this video: http://youtu.be/uXq379XkhVw
- ❑ Explore and study code in games that use score to learn more about creating variables and incorporating score into a project.

FINISHED?

- + Add your project to the Fish Chomp Remix studio: http://scratch.mit.edu/studios/475615
- + Challenge yourself to do more! How can you use score to add difficulty to your game design?
- + Find a game you are inspired by and remix it!

UNIT 4 ACTIVITY

EXTENSIONS

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ become more familiar with the concepts of conditionals, operators, and data by exploring programs that illustrate common game mechanics

ACTIVITY DESCRIPTION

- ❑ Optionally, show example projects from the Extensions studio and have the Extensions handout available to guide students.
- ❑ Give students time to explore the code of programs in the Extensions studio to investigate different ways games can be increased in difficulty or extended. Ask students to select one or more extensions to add to their previously started maze, pong, or scrolling game projects. Give students time to experiment and incorporate the extension(s) into their games.
- ❑ Allow students to share their extended game projects with one another. We suggest facilitating the pair-share or design demo activity to let students share their games and demonstrate what they learned.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Extensions handout
- ❑ Extensions studio
  http://scratch.mit.edu/studios/475619

REFLECTION PROMPTS

+ What are different ways of increasing difficulty in a game?
+ Which extensions did you add to your game project?
+ Describe your process for including the extension(s) in your game?

REVIEWING STUDENT WORK

+ Were students able to incorporate extensions into their original game projects?

NOTES

+ To provide more scaffolding for students needing extra support, we suggest walking through one extension sample program (e.g., levels) as a class and helping students add the extension to their game projects.
+ The backpack tool is one way students can incorporate parts of the extension projects into their starter games. Learn more about backpack at http://bit.ly/scratchbackpack

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

EXTENSIONS

HOW CAN YOU EXTEND AND REIMAGINE GAMES IN SCRATCH?

Get into game design by adding extended features within your Scratch project! Choose at least one (or more!) of the following extensions and add it to your previously started maze, pong, or scrolling games.

START HERE

- ❑ Go to the Extensions studio: http://scratch.mit.edu/studios/475619
- ❑ Choose one (or more) of the extensions to explore.
- ❑ Incorporate your choice into your previously started game projects!

+ **SCORE** http://scratch.mit.edu/projects/1940443
Demonstrates how to set and change a score. Receive 10 points every time the Scratch cat is clicked.

+ **LEVELS** http://scratch.mit.edu/projects/1940453
Demonstrates how to change levels. Score increases by 1 every time the space bar is pressed. Level increases by 1 for every 10 points.

+ **TIMER** http://scratch.mit.edu/projects/1940445
Demonstrates how to use a timer. Use the mouse to navigate the Scratch cat to Gobo.

+ **ENEMIES** http://scratch.mit.edu/projects/1940450
Demonstrates how to add an enemy. Avoid the tennis ball by using the up and down arrow keys.

+ **REWARDS** http://scratch.mit.edu/projects/1940456
Demonstrates how to collect items. Use the arrow keys to move the Scratch cat around to collect quest items.

+ **MOUSE** http://scratch.mit.edu/projects/25192659
Demonstrates how to program the mouse to control game play. Move the mouse to move the paddle.

+ **RESTART** http://scratch.mit.edu/projects/25192935
Demonstrates how to make a button to restart the game. Click on the RESTART button to restart.

+ **MENU** http://scratch.mit.edu/projects/25192991
Demonstrates how to display a menu screen at the beginning of the game. Click START or DIRECTIONS on the menu screen.

+ **MULTIPLAYER** http://scratch.mit.edu/projects/25192711
Demonstrates how to add another player to the game. Player 1 uses the arrow keys to navigate Pico through the maze, and player 2 uses the W, A, S, D keys to navigate Nano through the maze.

THINGS TO TRY

+ The backpack can be an extremely useful tool while programming in Scratch. It can store everything from lines of code, to music files, to sprites, and more. Try using it to incorporate extensions into your game projects.
+ Alternatively, sketching out ideas and bits of code in your design journal is another great method for planning how to incorporate your extensions.

FINISHED?

+ Add another extension to your maze, pong, or scrolling game.
+ Challenge yourself to do more! Continue going through each of the extensions and add them to your games.
+ Help a neighbor!
+ Share your project with a neighbor and give each other feedback on your games.

UNIT 4 ACTIVITY

INTERACTIONS

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ explore different approaches to making projects interactive by solving a series of nine programming puzzles
+ gain more fluency in the concepts of conditionals, operators, and data, and the practice of testing and debugging

ACTIVITY DESCRIPTION

- ❑ On their own or in small groups of 2-3 people, challenge students to further explore Scratch by creating Scratch programs that solve each of the nine Interactions programming puzzles. These Interactions puzzles explore Sensing blocks, engaging some of the more advanced concepts in Scratch related to interactivity. Optionally, have the Interactions handout available to guide students during the activity.
- ❑ Each puzzle can have several possible solutions. Invite students or groups to share different solutions and strategies. We suggest the Pair-Share or Design Demo activity to allow students to share their work and describe their process. Optionally, have students add their projects to the Interactions studio or a class studio.
- ❑ Ask students to think back on the challenge by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Interactions handout
- ❑ Interactions studio
  http://scratch.mit.edu/studios/487213

REFLECTION PROMPTS

+ Which puzzles did you work on?
+ What was your strategy for solving the puzzles?
+ Which puzzles helped you think about your game project?

REVIEWING STUDENT WORK

+ Are the puzzles solved?
+ Did students explore other approaches for solving the puzzles?
+ Are there certain blocks or concepts students are still struggling with? How might you help?

NOTES

+ Choose particular challenges that highlight new blocks or concepts that you would like students to explore. Or let students invent their own interaction puzzle prompts.
+ Repurpose these puzzles as an unstructured activity for students who finish other activities early or as a warm-up challenge. Create a puzzle jar: print out, cut, fold, and place copies of each puzzle description in a jar. Then, let students pick puzzles from the jar to solve.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

INTERACTIONS

WHAT DIFFERENTIATES A SCRATCH PROJECT FROM A STILL IMAGE OR A VIDEO?

Tackle these nine puzzles that engage some of the more advanced concepts in Scratch related to interactivity. Each of these challenges has several possible solutions.

START HERE

- ❑ Create a Scratch program for each of the nine interactivity puzzles.

- ❑ **PUZZLE 1:** Whenever you press the B key, the sprite gets a little bigger. Whenever you press the S key, the sprite gets a little smaller.
- ❑ **PUZZLE 2:** Whenever the sprite hears a loud sound, it changes color.
- ❑ **PUZZLE 3:** Whenever the sprite is in the top 25% of the screen, it says "I like it up here."
- ❑ **PUZZLE 4:** When the sprite touches something blue, it plays a high note. When the sprite touches something red, it plays a low note.
- ❑ **PUZZLE 5:** Whenever two sprites collide, one of them says: "Excuse me."
- ❑ **PUZZLE 6:** Whenever the cat sprite gets near the dog sprite, the dog turns and runs from the cat.
- ❑ **PUZZLE 7:** Whenever you click on the background, a flower appears at that spot.
- ❑ **PUZZLE 8:** Whenever you click on a sprite, all other sprites do a dance.
- ❑ **PUZZLE 9:** Whenever you move the mouse-pointer, the sprite follows but doesn't touch the mouse-pointer.

FEELING STUCK?

THAT'S OKAY! TRY THESE THINGS...

- ❑ Before getting started in Scratch, write down ideas in your design journal for possible ways of programming each of the interactivity puzzles.
- ❑ Work with a neighbor. Collaborating with a partner can be a great way to solve problems and gain new perspectives on ways of programming in Scratch!

FINISHED?

- + Add each of the projects you create to the Interaction Studio: http://scratch.mit.edu/studios/487213
- + Help a neighbor!
- + Discuss your strategies for approaching each puzzle with a partner. Take notes about the similarities and differences in your methods.

UNIT 4 ACTIVITY

DEBUG IT!

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ investigate the problem and find a solution to five debugging challenges
+ explore a range of concepts (conditionals, operators, and data) through the practices of testing and debugging

ACTIVITY DESCRIPTION

- ❑ Optionally, have the Unit 4 Debug It! handout available to guide students during the activity.
- ❑ Help students open the Debug It! programs from the Unit 4 Debug It! studio or by following the project links listed on the Unit 4 Debug It! handout. Encourage students to click on the "Look Inside" button to investigate the buggy program, tinker with problematic code, and test possible solutions.
- ❑ Give students time to test and debug each Debug It! challenge. Optionally, have students use the remix function in Scratch to fix the bugs and save corrected programs.
- ❑ Ask students to reflect back on their testing and debugging experiences by responding to the reflection prompts in their design journals or in a group discussion.
- ❑ Create a class list of debugging strategies by collecting students' problem finding and problem solving approaches.

RESOURCES

- ❑ Unit 4 Debug It! handout
- ❑ Unit 4 Debug It! studio
  http://scratch.mit.edu/studios/475634

REFLECTION PROMPTS

+ What was the problem?
+ How did you identify the problem?
+ How did you fix the problem?
+ Did others have alternative approaches to fixing the problem?

REVIEWING STUDENT WORK

+ Were students able to solve all five bugs? If not, how might you clarify the concepts expressed in the unsolved programs?
+ What different testing and debugging strategies did students employ?

NOTES

+ This activity provides an opportunity to check in with students who might need some additional attention or support, particularly around the concepts of conditionals (e.g., if), operators (e.g., arithmetic, logical), and data (e.g., variables, lists).

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

DEBUG IT!

HELP! CAN YOU DEBUG THESE FIVE SCRATCH PROGRAMS?

In this activity, you will investigate what is going awry and find a solution for each of the five Debug It! challenges.

START HERE

- ❑ Go to the Unit 4 Debug It! Studio: http://scratch.mit.edu/studios/475634/
- ❑ Test and debug each of the five debugging challenges in the studio.
- ❑ Write down your solution or remix the buggy program with your solution.

❑ **DEBUG IT! 4.1** http://scratch.mit.edu/projects/24271192

In this project, the "Inventory" list should be updated every time Scratch Cat picks up a new item. But Scratch Cat can only pick up the laptop. How do we fix the program?

❑ **DEBUG IT! 4.2** http://scratch.mit.edu/projects/24271303

In this project, Scratch Cat gets 10 points for collecting Yellow Gobos and loses 10 points for colliding with Pink Gobos. But something isn't working. How do we fix the program?

❑ **DEBUG IT! 4.3** http://scratch.mit.edu/projects/24271446

In this project, Scratch Cat is thinking of a number between 1 and 10. But something is wrong with the guess checking – it doesn't work consistently. How do we fix the program?

❑ **DEBUG IT! 4.4** http://scratch.mit.edu/projects/24271475

In this project, the "# of hits" display should increase by 1 every time the Scratch Cat is hit by a tennis ball. But the "# of hits" increases by more than 1 when Scratch Cat is hit. How do we fix the program?

❑ **DEBUG IT! 4.5** http://scratch.mit.edu/projects/24271560

In this project, Scratch Cat is navigating a maze to get to the yellow rectangle. But Scratch Cat can walk through walls. How do we fix the program?

FEELING STUCK?

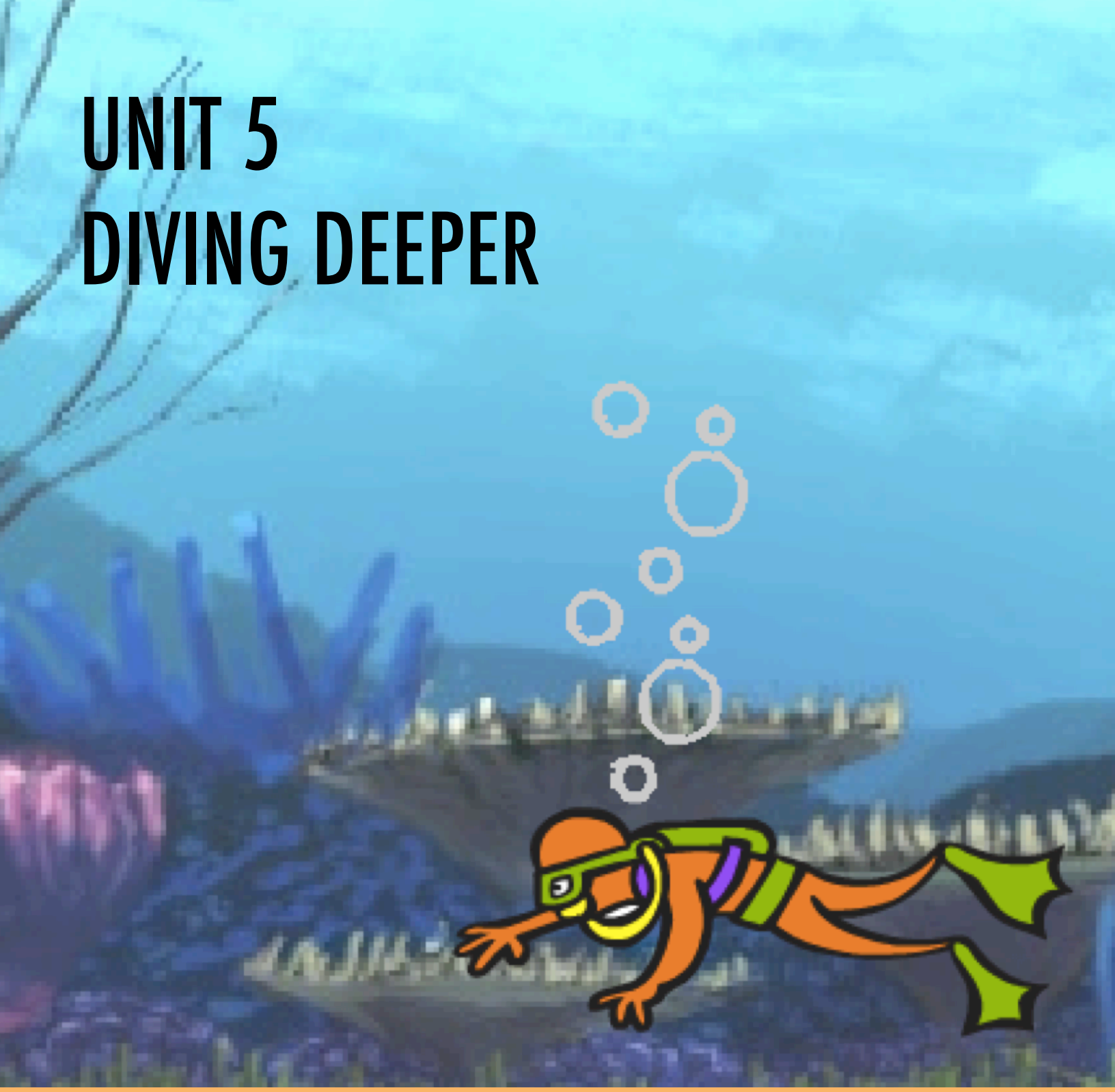
THAT'S OKAY! TRY THESE THINGS...

- ❑ Make a list of possible bugs in the program.
- ❑ Keep track of your work! This can be a useful reminder of what you have already tried and point you toward what to try next.
- ❑ Share and compare your problem finding and problem solving approaches with a neighbor until you find something that works for you!

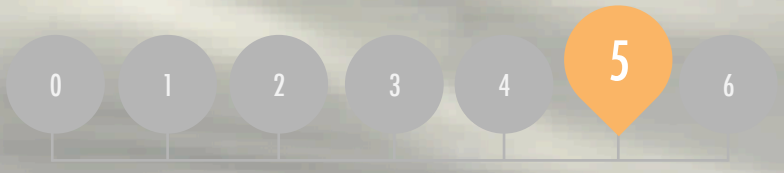
FINISHED?

- + Add code commentary by right clicking on blocks in your scripts. This can help others understand different parts of your program!
- + Discuss your testing and debugging practices with a partner. Make note of the similarities and differences in your strategies.
- + Help a neighbor!

UNIT 5
DIVING DEEPER



YOU ARE HERE



WHAT'S INCLUDED

UNIT 5 OVERVIEW

THE "BIG IDEA"

After the release of the previous version of the guide, a common piece of feedback that we received from teachers was that they (and the learners they support) wanted more "catch-up" time, time to linger, revisit, and extend the ideas and projects they had created in previous units. In response, we added this "Diving Deeper" unit.

Whether pushing ahead with advanced concepts and practices or revisiting previous experiences, this is an opportunity for learners to engage in a moment of contemplation and reflection. What isn't as clear as it could be? What do they still want to know about Scratch? How might others help them – and how might they help others?

This is also an opportunity for you, as educator, to engage in similar acts of contemplation and reflection. What has surprised you? What has made you uncomfortable? What would you want to do differently next time? Why?

LEARNING OBJECTIVES

Learners will:

+ reflect on past experiences to self-assess current learning goals and needs
+ create a self-remix by extending a previously started project
+ be introduced to various hardware extensions that connect Scratch to the physical world
+ gain more fluency in computational concepts and practices by exploring the newest Scratch features (video sensing, cloning)
+ experiment with designing learning experiences for others

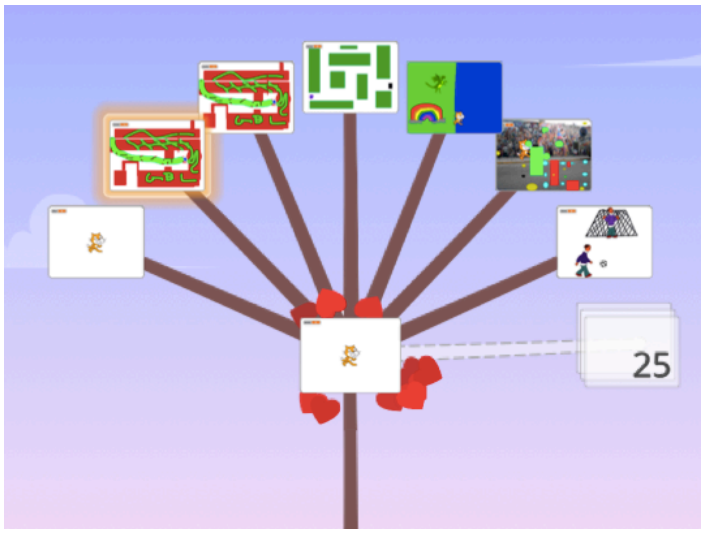
KEY WORDS, CONCEPTS, & PRACTICES

+ video sensing
+ cloning
+ peer interviews
+ hardware
+ extensions

NOTES

+ Not finding what you're looking for? Feel free to remix, reuse, and reimagine any of the activities in this guide to make it work best for you and your learners.
+ Search for lesson plans, activities, and resources designed for a specific curricular area on the ScratchEd website: http://scratched.gse.harvard.edu

CHOOSE YOUR OWN ADVENTURE



Rather than focusing on a particular theme or genre like the three previous units, this unit is intended to create a space for reviewing and reflecting on prior work. This unit's activities are especially flexible, diving deeper into creative computing by revisiting challenges, extending skills, or refining practices.

Begin by inviting students to review their past work and engage in self-assessment of their learning goals in the Know Want Learn activity.

Then, encourage students to dive deeper into Scratch by choosing which follow-up activities to pursue.

POSSIBLE PATH

SESSION 1

KNOW WANT LEARN

What do you know? What do you want to know? What have you learned?

SESSION 2

ROUND TWO

Remix a past project, go back to a missed activity, or challenge yourself to learn something new.

SESSION 3

HARDWARE & EXTENSIONS

Creating with Scratch can go beyond what happens on the screen.

ADVANCED CONCEPTS

Create a project that explores video sensing or cloning.

SESSION 4

ACTIVITY DESIGN

Design a learning experience for others to try.

SESSION 5

MY DEBUG IT!

Create your own Debug It! program and see if others can solve it.

UNIT 5 ACTIVITY

KNOW WANT LEARN

SUGGESTED TIME
30–45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ reflect on past projects and experiences
+ self-assess current knowledge and learning goals
+ pursue personal learning interests in a self-directed research activity

ACTIVITY DESCRIPTION

- ❑ In this self-directed learning activity, students will reflect on current understandings and build new knowledge based on their interests. Optionally, have the Know Want Learn worksheet available to guide students.
- ❑ Ask students to reflect on what they know already and what they want to know next about Scratch and creative computing. Guide students in answering the first two reflection prompts in their design journals or using the Know Want Learn self-assessment worksheet. Next, give students time to pursue learning interests from their "What do you want to know?" responses. Finally, have students respond to the third and fourth reflection prompts in their design journals or using the Know Want Learn worksheet.
- ❑ Help students share their reflections and learning interests with one another. We recommend peer interviews: divide students into pairs and have them take turns interviewing one another about their processes of reflection, self-assessment, and research.

RESOURCES

- ❑ Know Want Learn worksheet
- ❑ Scratch Wiki
  http://wiki.scratch.mit.edu
- ❑ Scratch Discussion Forums
  http://scratch.mit.edu/discuss
- ❑ Scratch FAQ
  http://scratch.mit.edu/help/faq

REFLECTION PROMPTS

+ What do you know?
+ What do you want to know?
+ What did you learn?
+ What were your strategies for investigating what you wanted to know?

REVIEWING STUDENT WORK

+ Were students able to learn what they wanted to know?
+ What strategies and resources did they employ?

NOTES

+ Help students find and use other resources during their research such as leveraging knowledgeable peers, posing questions to family members and friends, or posting a question in the Scratch discussion forums.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

KNOW WANT LEARN

NAME: _____

What do you know about creative computing & scratch? What do you want to know next? This activity is an opportunity for you to consider which areas of Scratch you feel comfortable navigating (What do I know?) and which areas you would like to explore further (What do I want to know?). Use different resources around you to investigate what you want to know, and then share your findings (What did I learn?).

WHAT DO I KNOW?

Reflecting on your design experiences so far, write down what you know about Scratch and creative computing.

WHAT DO I WANT TO KNOW?

Based on your personal interests, generate a list of things you want to find out more about or discover next.

WHAT DID I LEARN?

Gather resources to investigate items from the list you created above, and then share what you learned from your research.

UNIT 5 ACTIVITY

ROUND TWO

SUGGESTED TIME
45-60 MINUTES

OBJECTIVES

By completing this activity, students will:

+ have the opportunity to create a self-remix of past work or spend time on a unit activity that was previously skipped or not completed

ACTIVITY DESCRIPTION

- ❑ Optionally, have the activity handouts from Units 0-5 available to guide students.
- ❑ Give students self-directed time to:
  1. reimagine or extend a past project by creating a self-remix: a remix of one's own project.
  2. revisit and work on a previous unit activity that was either skipped or not completed.
- ❑ Encourage students to share their self-remixes or activity outcomes with one another. We suggest using pair-share or design demo.
- ❑ Invite students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Units 0-5 handouts

REFLECTION PROMPTS

+ Why did you choose that project or activity to work on?
+ What would you do if you had more time?

REVIEWING STUDENT WORK

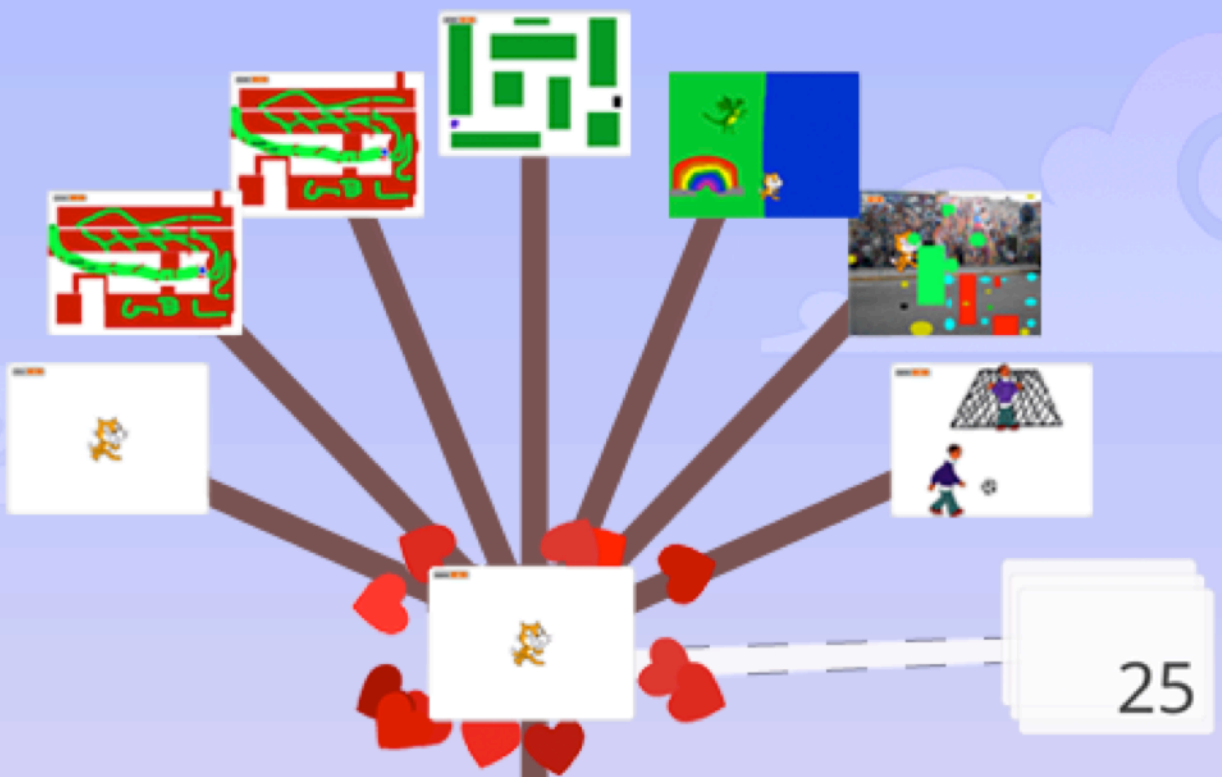
+ Did students create self-remixes or work on activities?
+ What did you learn about your students' interests?
+ What further support might your students need?

NOTES

+ Invite students to review their design journals and Scratch profiles to reflect back on previous work and activities.
+ Encourage students to review their Unit 1 My Studio inspiration projects for ideas.

NOTES TO SELF

- ❑ _____
- ❑ _____
- ❑ _____
- ❑ _____


25

UNIT 5 ACTIVITY

ADVANCED CONCEPTS

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ gain more fluency with computational concepts (events, parallelism, data) and practices (experimenting and iterating, testing and debugging, reusing and remixing, abstracting and modularizing) by creating a project exploring video sensing or cloning

ACTIVITY DESCRIPTION

- ❑ Use the Advanced Concepts, Video Sensing examples, and Cloning examples studios to show examples and help students get familiar with blocks that control video sensing and cloning. Optionally, have the Advanced Concepts handout available to guide students.
- ❑ Give students time to explore the code of example programs to create a project that experiments with one or more of the advanced concepts (video sensing, cloning).
- ❑ Encourage students to share their explorations with others. We suggest hosting design demo presentations. Optionally, have students add their projects to the Advanced Concepts studio or a class studio.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Advanced Concepts studio http://scratch.mit.edu/studios/221311
- ❑ Video Sensing handout
- ❑ Video Sensing examples studio http://scratch.mit.edu/studios/201435
- ❑ Cloning handout
- ❑ Cloning examples studio http://scratch.mit.edu/studios/201437

REFLECTION PROMPTS

+ Which advanced concept(s) did you choose to explore?
+ What was your strategy for learning more about the concept(s) you selected?

REVIEWING STUDENT WORK

+ Do projects explore one or more of the advanced concept(s)?

NOTES

+ Students who want to explore the video sensing feature will require a computer with a webcam.
+ Remind students that the backpack tool can be used to borrow and remix code from example projects.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....


Motion
move 10 steps
turn 15 degrees
turn 15 degrees
point in direction 90
point towards
go to x: 0 y: 0
go to
glide 1 secs to x: 0 y: 0
change x by 10
set x to 0
change y by 10
set y to 0
when clicked
show
say
hide
forever
when clicked
show
point in direction 90
move 10 steps
Sprites
New sprite
Sprite1
Sprite2

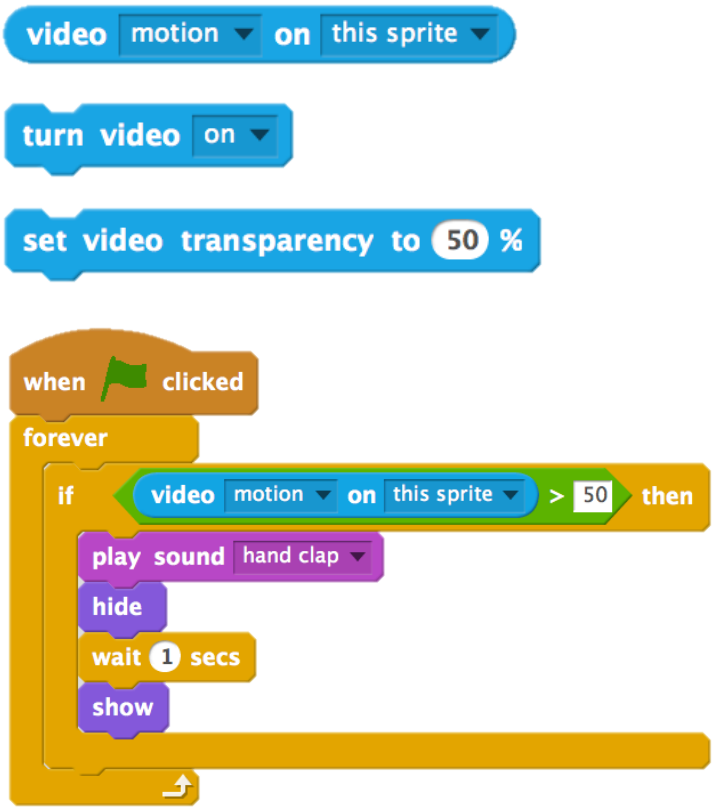
VIDEO SENSING

HOW CAN YOU USE VIDEO SENSING IN YOUR SCRATCH PROJECTS?

Did you know that you can make your Scratch projects interactive through a webcam? Explore this advanced Scratch concept by creating a project that incorporates the video sensing feature.

START HERE

- ❑ Open an existing Scratch project or start a new project to add video sensing.
- ❑ Check out blocks for video sensing in the Sensing category.
- ❑ Experiment with video on, turn video, and set video transparency to blocks to program your project to sense video motion.



THINGS TO TRY

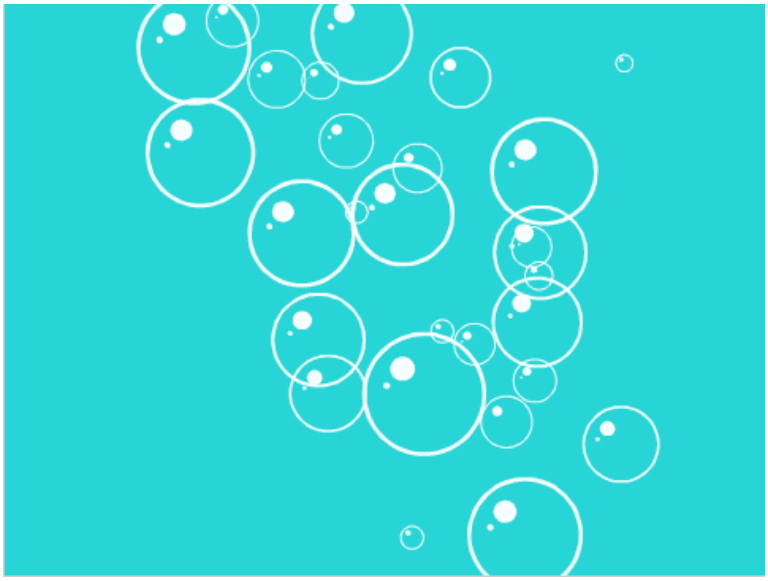
- ❑ Make sure your webcam is connected! Test it out using the turn video on block.
- ❑ If you're feeling a little stuck, that's okay! Explore some of the other projects in the Video Sensing studio to see how they use the video blocks or use the Tips Window to learn more about video sensing.

FINISHED?

- + Add your project to the Advanced Concepts studio: http://scratch.mit.edu/studios/221311
- + Add video sensing to one of your past projects!
- + Help a neighbor!
- + Remix a project in the Video Sensing studio.

CLONING

HOW CAN YOU USE CLONING IN YOUR SCRATCH PROJECTS?

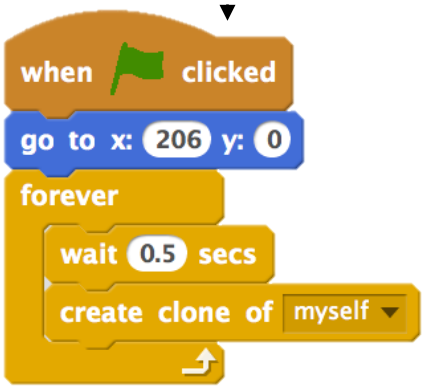


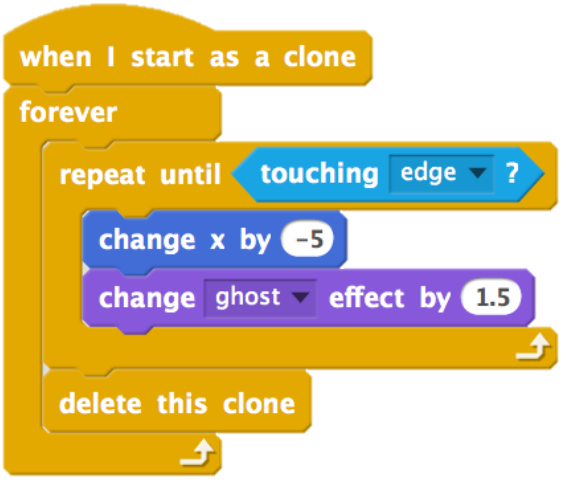
Cloning is an easy way to create multiples of the same sprite. You can use cloning to make many objects and create cool effects in a project.

Explore this advanced Scratch concept by creating a project that incorporates the cloning feature.

START HERE

- ❑ Open an existing Scratch project or start a new project to experiment with cloning.
- ❑ Check out blocks for cloning in the Control category.
- ❑ Experiment with the blocks to create clones of your sprite. Define behaviors for what your cloned sprites will do.





THINGS TO TRY

- ❑ If you can't see your clone initially, check if the original sprite is in the same location – it might be covering the clone! Program your original sprite or the clone to move or go to different locations so you can see them.
- ❑ Stuck? That's okay! Explore some of the other projects in the Cloning Studio to see how they use cloning or search in the Tips Window to learn more about the Create Clone and When I start as a Clone blocks.

FINISHED?

- + Add your project to the Cloning studio: http://scratch.mit.edu/studios/201437
- + Add cloning to one of your past projects!
- + Help a neighbor!
- + Remix a project in the Cloning studio.

UNIT 5 ACTIVITY

HARDWARE & EXTENSIONS

SUGGESTED TIME
45-60 MINUTES

OBJECTIVES
By completing this activity, students will:
+ be introduced to various hardware extensions that connect the digital world of Scratch with the physical world

ACTIVITY DESCRIPTION

- ❑ (IMPORTANT: This activity requires access to one or more of these hardware products.) Introduce students to ways Scratch can connect to other technologies and hardware extensions including the LEGO WeDo, MaKey MaKey, and PicoBoard. Optionally, show examples from the How can I connect Scratch with other technologies? video playlist.
- ❑ Divide students into small groups of 2-4 people. Give the groups time to explore creating a Scratch project that incorporates a physical world component using one or more available hardware extensions.
- ❑ Allow each group to share their creations with others. We suggest facilitating a gallery walk or feedback fair.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ LEGO WeDo construction set
  http://bit.ly/LEGOWeDo
- ❑ MaKey MaKey
  http://makeymakey.com
- ❑ PicoBoard
  https://www.sparkfun.com/products/10311
- ❑ How can I connect Scratch with other technologies? videos
  http://bit.ly/hardwareandextensions

REFLECTION PROMPTS

+ Which hardware or extension did you explore?
+ How did you incorporate the digital and the physical?
+ What was difficult?
+ What was surprising?

REVIEWING STUDENT WORK

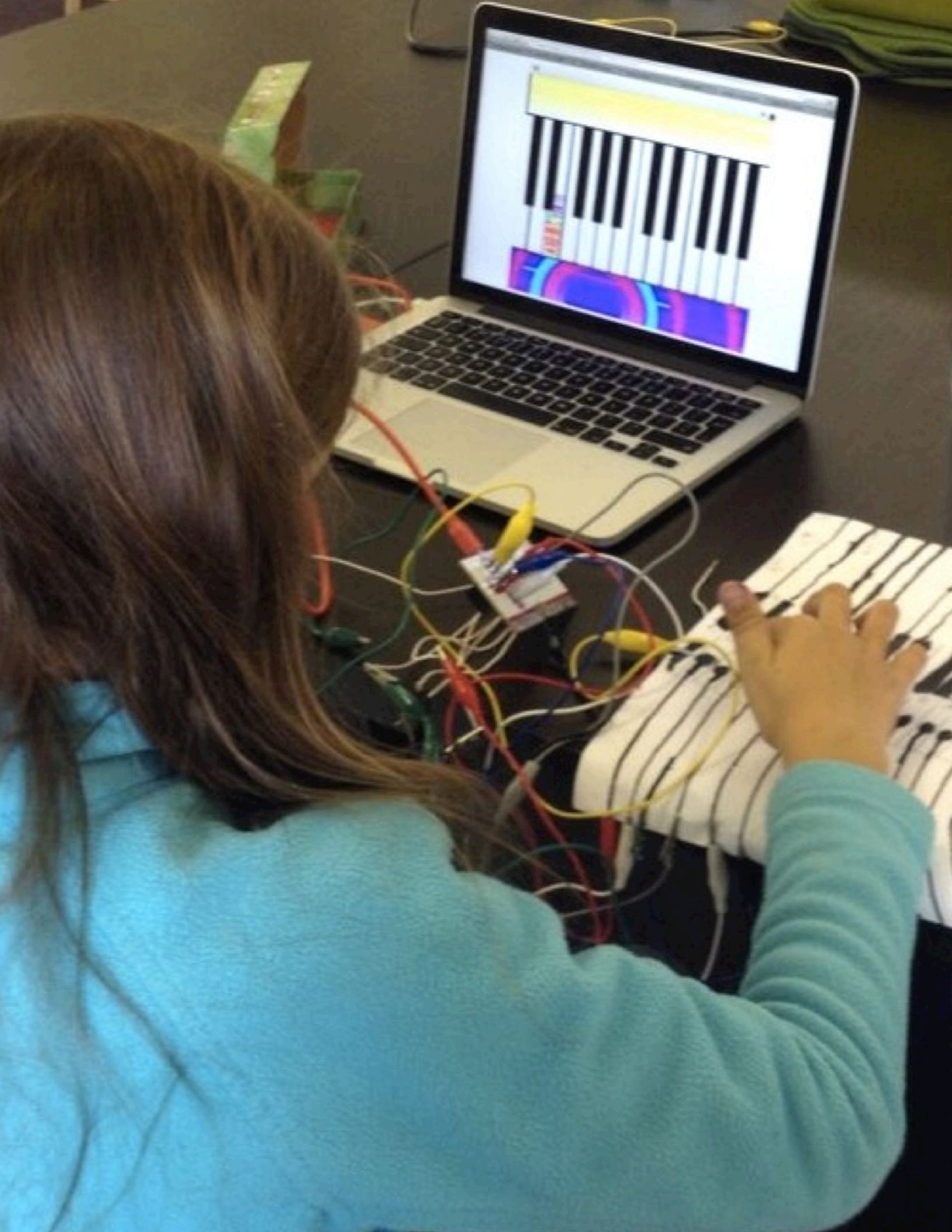
+ Does the work have a digital and a physical component?

NOTES

+ Make this a group-wide activity! Using the LEGO WeDo and Scratch, challenge students to connect their projects to create a chain of reactions in the style of a Rube Goldberg machine. See this video for an example: http://bit.ly/ScratchChainReaction
+ Activate the Scratch blocks that control hardware extensions by clicking on the Add an Extension button located under the More Blocks category in the Scratch project editor.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....



UNIT 5 ACTIVITY

ACTIVITY DESIGN

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ design an activity or resource for supporting others in learning more about Scratch and computational creativity

ACTIVITY DESCRIPTION

- ❑ Let students experience what it's like to be in your teaching shoes! Challenge students to create, remix, or reimagine an activity or resource designed to support others' explorations of creative computing. Optionally, have the Activity Design handout available for additional support.
- ❑ Help students brainstorm and imagine different kinds of creative learning experiences. Optionally, review example project ideas and activities from this guide, or encourage students to explore the Scratch Cards resource and Scratch Design Studio list for inspiration. Then, give students time to design their own learning activity or resource.
- ❑ Give students opportunities to test out their activity or resource with learners. Encourage them to share their activity or resource with family or friends, or invite students to be peer mentors for other classes, clubs, or events.
- ❑ Ask students to think back on the design process by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ Activity Design handout
- ❑ Scratch Cards
  http://scratch.mit.edu/help/cards
- ❑ Scratch Design Studio list
  http://scratch.mit.edu/users/ScratchDesignStudio/

REFLECTION PROMPTS

+ Who do you envision using your activity or resource?
+ What do you hope people will learn from using your activity or resource?
+ What challenges might learners experience in doing the activity or using the resource? How might you further support them in dealing with these challenges?

REVIEWING STUDENT WORK

+ Does the activity or resource facilitate an introduction or exploration into creative computing? What feedback can you offer the student?

NOTES

+ Students particularly interested in supporting others' learning can be great candidates for becoming peer mentors during class or at an afterschool or lunchtime Scratch Club.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

ACTIVITY DESIGN

NAME: _____

How can you help others learn more about Scratch and creative computing? Design an activity that helps other people learn Scratch. It can be an off-computer activity (like Creature Construction), project idea (like Build-a-Band), or challenge activity (like Debug It!). You could even develop a new type of activity or handout! Brainstorm using the questions below, and then use the activity and handout planners to give more detail.

WHO IS THIS FOR?

Who is your audience? Who do you want to help learn more about Scratch and creative computing?

WHAT WILL THEY LEARN?

What are the learning goals? What new things do you hope people will learn from using your activity?

WHAT DO THEY NEED?

What supplies will people need? What other types of support will help people successfully engage in your activity?

MY ACTIVITY

(TITLE)

SUGGESTED TIME
__-__ MINUTES

OBJECTIVES

(2 LEARNING GOALS)

By completing this activity, learners will:

+

+

ACTIVITY DESCRIPTION

(PROJECT INSTRUCTIONS)

❑ What will learners create? How will they do this?

❑ How will learners share their work with others?

❑ How will learners reflect on their designs?

RESOURCES

(2 PROJECT RESOURCES - studios, handouts, etc.)

❑

❑

REFLECTION PROMPTS

(3 REFLECTION QUESTIONS)

+

+

+

REVIEWING STUDENT WORK

(2 WAYS TO CHECK IF A LEARNER COMPLETED THE ACTIVITY)

+

+

NOTES

(TIPS AND TRICKS)

+

+

+

NOTES TO SELF

❑ .....

❑ .....

❑ .....

❑ .....

(TITLE)

(PROJECT OVERVIEW)

(PROJECT DESCRIPTION)

(PROJECT PICTURE)

START HERE

(PROJECT INSTRUCTIONS)

- ❑
- ❑
- ❑
- ❑

(ILLUSTRATED PROJECT INSTRUCTIONS)

THINGS TO TRY

(3 THINGS TO DO IF THEY GET STUCK)

- ❑
- ❑
- ❑

FINISHED?

(3 THINGS TO DO IF THEY HAVE EXTRA TIME)

- +
- +
- +

UNIT 5 ACTIVITY

MY DEBUG IT!

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ become more fluent with computational practices (experimenting and iterating, testing and debugging, reusing and remixing, abstracting and modularizing) by designing a debugging challenge

ACTIVITY DESCRIPTION

- ❑ Optionally, have the My Debug It! handout available to guide students during the activity.
- ❑ Give students the opportunity to create their own debugging challenge for others to solve. Bugs can focus on a specific computational concept, Scratch block, interaction, or some other programming challenge. Encourage students to take inspiration from their own experiences of getting stuck and unstuck while developing Scratch projects.
- ❑ Let students swap and try testing and debugging one another's buggy projects. Optionally, have students add their debug-it program to the My Debug It! studio or a class studio.
- ❑ Ask students to reflect back on their problem-creating approaches by responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ My Debug It! handout
- ❑ My Debug It! studio
  http://scratch.mit.edu/studios/475637

REFLECTION PROMPTS

+ What was the problem?
+ Where did your inspiration come from?
+ How did you imagine others investigating and solving the challenge?
+ Did others have alternative approaches to finding and fixing the problem than what you expected? What were their strategies?

REVIEWING STUDENT WORK

+ Do projects include a debugging challenge to solve?
+ What different testing and debugging strategies did students employ?

NOTES

+ Remind students to include a challenge description in the notes of the project page on the Scratch website.
+ Got extra time or need a warm-up activity? Let students exercise their problem-seeking and problem-solving skills on other contributed debug-it programs in the My Debug It! studio.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

MY DEBUG IT!

IT'S TIME TO DESIGN YOUR OWN DEBUG IT PROGRAM. WHAT WILL YOU CREATE?

In this activity, you will create your own Debug It! challenge for others to investigate, solve, and remix.

START HERE

- ❑ Reflect back on the different kinds of bugs you've encountered in creating and debugging your own projects.
- ❑ Generate a list of possible debugging challenges you could create. A Debug It! can focus on a specific concept, block, interaction, or some other programming challenge.
- ❑ Build your Debug It! program.

PLANS FOR MY DEBUG IT!

NOTES TO SELF

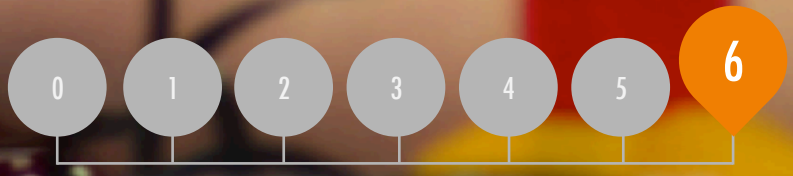
- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

FINISHED?

+ Add your debugging challenge to the My Debug It! studio: http://scratch.mit.edu/studios/475637
+ Swap Debug It! programs with a neighbor and try to solve each other's buggy programs.
+ Help a neighbor.
+ Try debugging other programs in the My Debug It! studio.

UNIT 6
HACKATHON

YOU ARE HERE



WHAT'S INCLUDED

UNIT 6 OVERVIEW

THE "BIG IDEA"

In this final unit, students will build on their creative computing experiences by engaging in the design of an open-ended project of their choosing. To help you and your students tackle this open-ended design experience, we were inspired to frame this unit as a hackathon. With its ethos of embracing just-in-time learning and problem solving, encouraging iterative planning-making-sharing, and celebrating a connected and collaborative environment, the hackathon is an ideal creative computing culminating experience.

School is done but some students do not seem to notice. Busy debugging their #scratch game. A team effort. @Sheena1010

LEARNING OBJECTIVES

Students will:

+ be introduced to the format of a hackathon event
+ demonstrate knowledge of computational concepts (sequence, loops, events, parallelism, conditionals, operators, data) and practices (experimenting and iterating, testing and debugging, reusing and remixing, abstracting and modularizing) by defining, developing, and presenting a personally meaningful, self-directed project
+ have multiple opportunities for collaboration by working in peer teams, sharing skills, and giving and receiving multiple rounds of feedback

KEY WORDS, CONCEPTS, & PRACTICES

+ hackathon
+ design sprint
+ project pitch
+ unfocus group
+ showcase

NOTES

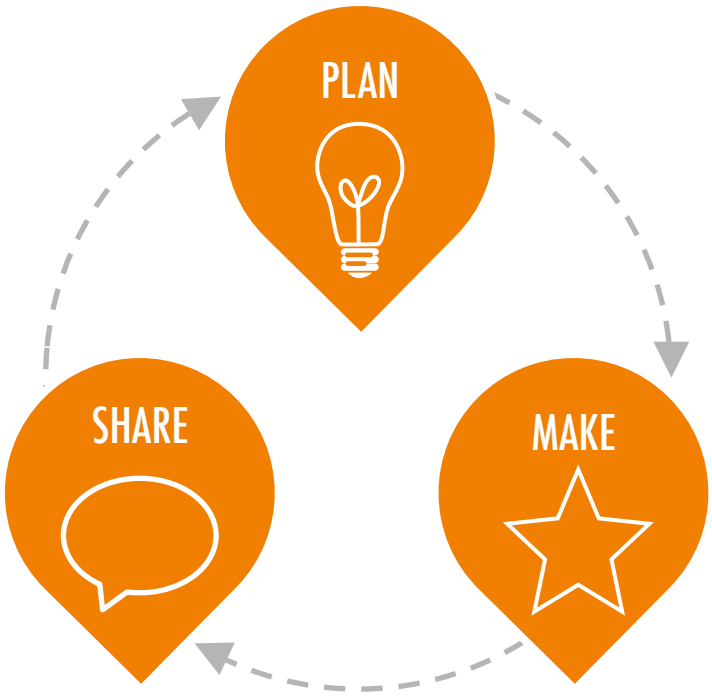
+ This unit can accommodate either independent or collaborative group projects. Pick one option or allow students to choose.

WHAT IS A HACKATHON?

"Hack" has a negative connotation to some – but it has a long history of standing for playfulness, curiosity, persistence, and creativity. One of our favorite definitions frames "hack" as "an appropriate application of ingenuity". With this definition, what better capacity for young learners than learning how to "hack"?

A hackathon takes the playful ingenuity of hacking – and situates it in an intensely focused and time-limited context. In this unit, learners will brainstorm an idea, develop a project, and showcase a final prototype using an iterative plan-make-share cycle.

Hackathons provide excellent opportunities for learners to invent their own personally meaningful and relevant projects to work on, which can be developed as independent final projects or in collaborative teams. It is a chance for students to demonstrate their knowledge in Scratch, expand upon current skills, and develop and test ideas within a collaborative, creative, flexible, and playful learning environment.



HOW DOES IT WORK?

Throughout the duration of the hackathon, students will engage in iterative cycles in which they PLAN, MAKE, and SHARE. This iterative cycle encourages students to engage in meaningful acts of ideation, creation, and reflection.

PLAN

What do you want to work on? Brainstorm ideas and prepare a plan of action!

MAKE

Design and develop project creations with resources and help from others.

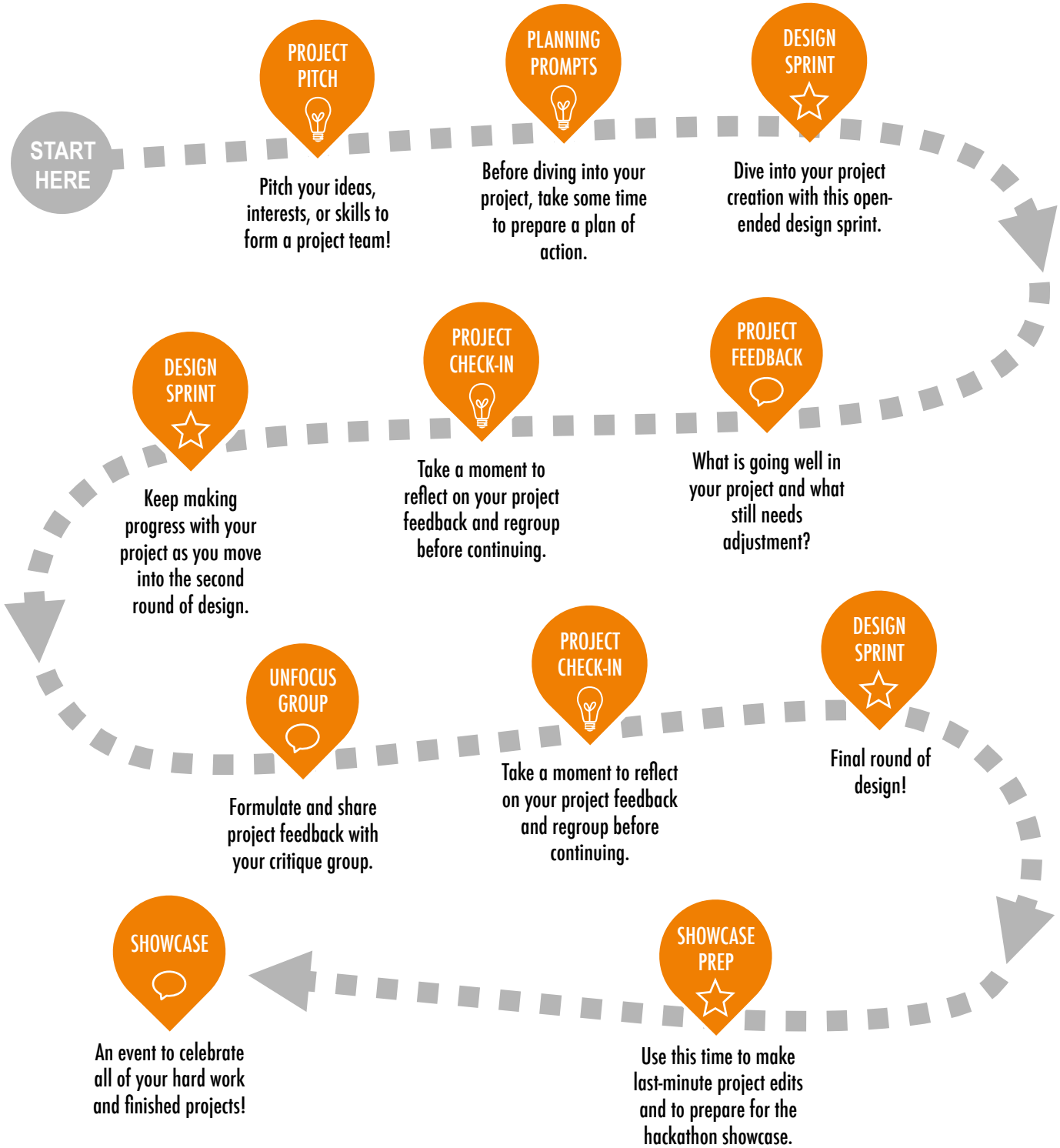
SHARE

Share your project with others and gather feedback to guide your next steps!

POSSIBLE PATH

The hackathon-inspired activities for this unit are designed to challenge students to build up a more complex project within an open-ended and collaborative learning environment. All of the important culture-building we've been doing – encouraging risk-taking and persistence, recognizing failures as learning opportunities, focusing on process over product, and cultivating a culture of cooperation and fun – culminates in this unit.

To help you get started, we have included a suggested sequence of activities that follow the plan-make-share design cycle.



UNIT 6 ACTIVITY

PROJECT PITCH

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ brainstorm project ideas based on personal interests
+ pitch ideas, interests, and skills to form project teams

ACTIVITY DESCRIPTION

- ❑ Introduce students to the concept of a pitch. With a pitch, students can either announce a project idea in order to recruit other team members, or they can promote their interests, skills, or talents, in order to be recruited by other teams.
- ❑ Distribute the Pitch handout, giving students time to brainstorm and to respond on the handout. Some students already may have a project idea or have identified a specific interest or skill they want to share or further explore. Let students know that if they don't have a specific project idea or interest, they will have an opportunity to join another team. Optionally, have students review inspiration projects identified during the Unit 1 My Studio activity.
- ❑ Give each student a chance to pitch to the rest of the group. Ask people to line up if they want to pitch, and give them thirty seconds each to describe their project, interest, or skill.
- ❑ Provide time for students to form project groups of 3 or 4 people. Optionally, have students write their names and project interests on sticky notes that can be arranged and sorted on a wall to facilitate team-building.

RESOURCES

- ❑ Pitch handout
- ❑ sticky notes (optional)

REFLECTION PROMPTS

+ What has been your favorite project to work on so far?
+ What kinds of projects are you interested in creating next?
+ What knowledge, skills, or talents could you contribute to a project?

REVIEWING STUDENT WORK

+ Did each student get a chance to pitch their idea or interests?
+ Did each student find a project team to join?

NOTES

+ Students can be enormously valuable in providing support and guidance to each other throughout all of the Scratch sessions, and particularly during the hackathon sessions. Encouraging young people to share their knowledge and skills with others makes things easier for the facilitator, but can also significantly deepen creators' learning and understanding.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

PROJECT PITCH

PROJECT PITCH BY: _____

Use the prompts below to brainstorm ideas for projects you're interested in working on during the hackathon. You will have 30 seconds to pitch your ideas, interests, and skills to the rest of the group!

MY FAVORITE PROJECT

What has been your favorite project to work on so far? What made this project stand out for you?

MY HACKATHON PROJECT IDEA

What kinds of projects are you interested in creating next?

MY SKILLS AND INTERESTS

What knowledge, skills, or talents would you like to contribute to a project?

UNIT 6 ACTIVITY

PROJECT PLANNING

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ identify an appropriately-scoped project to work on
+ develop an outline of activities or tasks required to complete the project
+ generate a preliminary list of resources required to complete the project

ACTIVITY DESCRIPTION

- ❑ Taking some time at the start of the final project to explore ideas, identify tasks involved in completing the project, and list what is (and isn't) already known can be very beneficial for successful project completion.
- ❑ Divide the group into project teams. Optionally, distribute the Project Planning and Project Sketches handouts to each team or individual.
- ❑ Review different elements for planning projects (project sketches, outline of tasks, list of resources, storyboards/ wireframes). Give the teams 15 minutes to brainstorm ideas, plans, and resources for their projects. Students who already have a clear concept and plan are welcome to start working on their project design.
- ❑ Optionally, collect the completed Project Planning and Project Sketches handouts at the end of this activity to return to students at the beginning of Design Sprint sessions.

RESOURCES

- ❑ Project Planning handout
- ❑ Project Sketches handout

REFLECTION PROMPTS

+ What project do I want to create?
+ What steps will I take to develop my project?
+ What resources (e.g., people, sample projects) do I already have to develop my project?
+ What resources (e.g., people, sample projects) might I need to develop my project?

REVIEWING STUDENT WORK

+ Is the project appropriately scoped for the amount of time and resources available for this hackathon?
+ How can you make resources accessible to students who need them?

NOTES

+ Although planning is helpful, it shouldn't be all-consuming or the only way of doing things. Different students will want and need to plan and tinker to different extents – and different phases of the project will require different approaches. Multiple design and development styles should be encouraged and accommodated.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....


Music
Garden
Magic
Two player games
Sports
Art

PROJECT PLANNING

PROJECT PLANS BY: _____

Use the prompts below to start thinking about the elements needed to develop your project.

MY PROJECT

Describe the project you want to create.

List the steps needed in order to create your project.

MY RESOURCES

What resources (e.g., people, sample projects) do you already have?

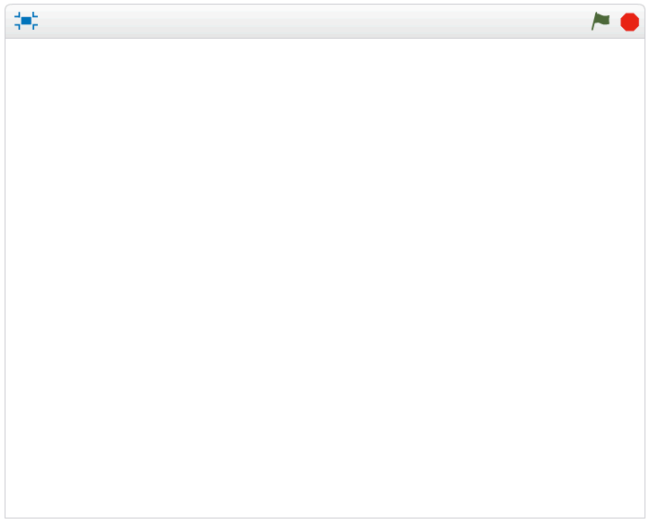
What resources (e.g., people, sample projects) might you need to develop your project?

PROJECT SKETCHES

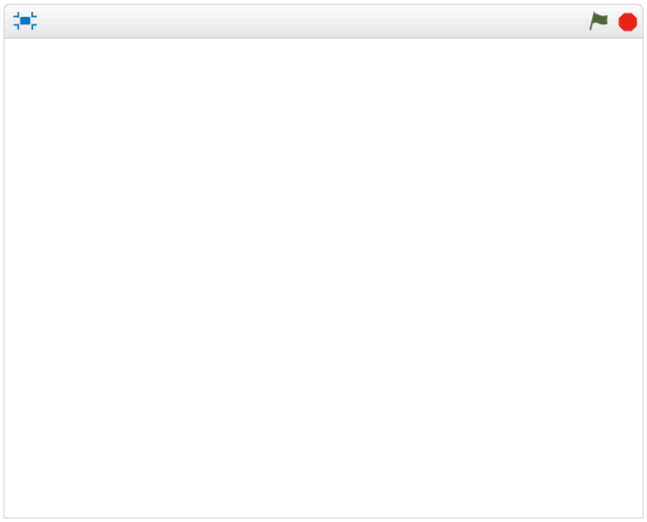
PROJECT SKETCHES BY: _____

Use the space below to draw sketches of what your project will look like!

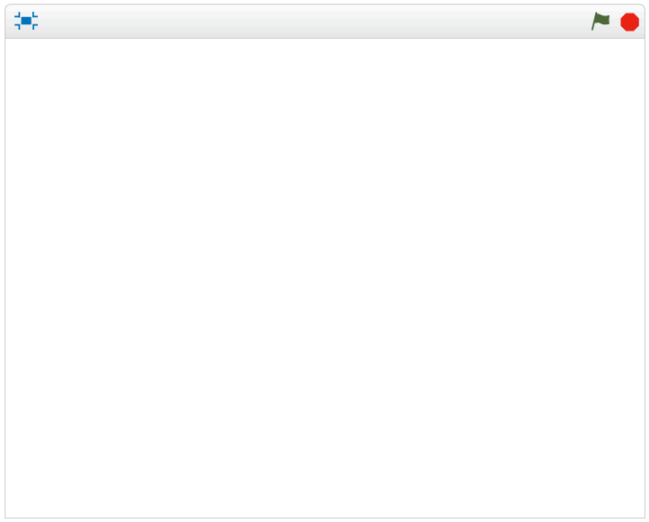
MY PROJECT SKETCHES



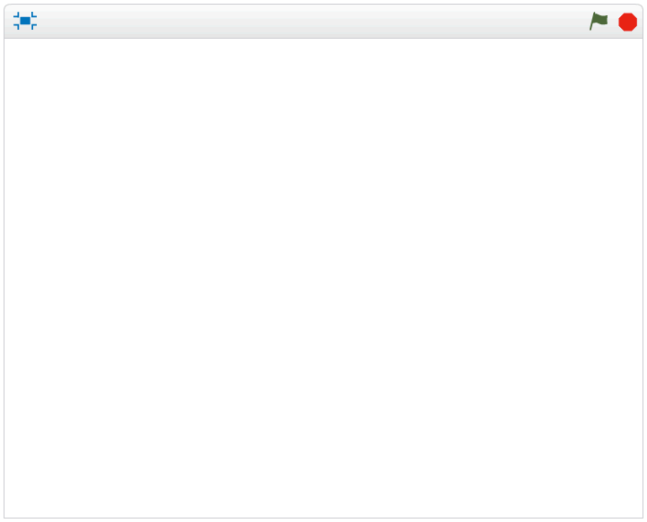
What's happening? What are the important elements?



What's happening? What are the important elements?



What's happening? What are the important elements?



What's happening? What are the important elements?

UNIT 6 ACTIVITY

DESIGN SPRINT

SUGGESTED TIME
45-60 MINUTES

OBJECTIVES

By completing this activity, students will:

+ use computational concepts and practices to further develop a Scratch project of their choosing

ACTIVITY DESCRIPTION

- ❑ Introduce students to the concept of a design sprint, which is a specified amount of time dedicated to working intensely on developing projects.
- ❑ Ask students to write down goals for this session using the Project Check-In activity or by responding to the reflection prompts in their design teams or in their design journals. Give students their completed Project Planning, Project Feedback, and Unfocus Group handouts to guide them in reflecting on original project goals and to encourage them to make plans for refinement based on feedback.
- ❑ Give students self-directed time to work on their projects. Introduce and distribute additional support resources as needed. In addition to peer support, having a collection of readily-available support resources can help students continue to make progress. Sample projects on the Scratch website (http://scratch.mit.edu) can provide ideas, and additional resources can be found on the ScratchEd website (http://scratched.gse.harvard.edu).
- ❑ Optionally, ask students to post their project drafts in a class studio.

RESOURCES

- ❑ additional resources (e.g., sample projects, handouts, Scratch Cards, craft material)

REFLECTION PROMPTS

+ What part of your project will you be working on today?
+ What might you need help with in order to make progress?

REVIEWING STUDENT WORK

+ Are individuals or groups making reasonable progress?
+ What feedback or suggestions do you have for the projects?

NOTES

+ All design activities are constrained – by time, by resources, by our own abilities at a given moment – and compromises may need to be made. The open-ended designing sessions are a great opportunity to have conversations with students about the essential elements of their projects. What are the most important aspects of the projects? What can reasonably be accomplished in the remaining time?

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....


MacBook Air

UNIT 6 ACTIVITY

PROJECT FEEDBACK

SUGGESTED TIME
30-45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ will work together in small critique groups to give each other preliminary feedback on their projects
+ test projects-in-progress
+ formulate and share feedback for others

ACTIVITY DESCRIPTION

- ❑ Divide the group into feedback teams of 3-4 people, so that their feedback team members are not also members of their project team. Optionally, have students gather in their critique groups from the Unit 0 Critique Group activity.
- ❑ Distribute the Project Feedback handout to each person, and review the different handout elements. Ask students to fill out the top portion of the handout with their name and project title.
- ❑ Ask students to spend 10 minutes reviewing each project in their feedback team and critiquing the project draft using the Red, Yellow, Green feedback questions. When the review is complete, each student will have received feedback on their project from the other members of their feedback group.
- ❑ After all the feedback rounds have ended, give students time to meet with their project team members to review the feedback and reflect on which suggestions they want to incorporate into their project during the next Design Sprint session. Optionally, collect the completed Project Feedback handouts at the end of this activity to return to students at the beginning of the Project Check-In activity or Design Sprint sessions.

RESOURCES

- ❑ Project Feedback handout

REFLECTION PROMPTS

+ What aspects of your project could someone give you feedback about?
+ What feedback, if any, do you plan to incorporate into your project next?

REVIEWING STUDENT WORK

+ Did each student have opportunities to give and receive feedback from various sources?
+ Did each student complete the Project Feedback handout?

NOTES

+ Different people will provide different perspectives on the project-in-progress. Create opportunities for learners to get feedback from a variety of sources, including themselves!

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

PROJECT FEEDBACK

FEEDBACK FOR: _____

PROJECT TITLE: _____

RED, YELLOW, GREEN

FEEDBACK BY	[RED] What is something that doesn't work or could be improved?	[YELLOW] What is something that is confusing or could be done differently?	[GREEN] What is something that works well or you really like about the project?

PARTS OF THE PROJECT THAT MIGHT BE HELPFUL TO THINK ABOUT:

+ Clarity: Did you understand what the project is supposed to do?
+ Features: What features does the project have? Does the project work as expected?
+ Appeal: How engaging is the project? Is it interactive, original, sophisticated, funny, or interesting? How did you feel as you interacted with it?

UNIT 6 ACTIVITY

PROJECT CHECK-IN

SUGGESTED TIME
15-30 MINUTES

OBJECTIVES

By completing this activity, students will:

+ review project progress and feedback
+ develop an outline of activities or tasks required to complete the project
+ generate a list of resources required to complete the project

ACTIVITY DESCRIPTION

- ❑ In this activity, students will perform a project check-in, where they will update fellow team members about their design progress so far and outline a plan for an upcoming design sprint based on feedback received. Optionally, give students or groups a Project Check-In handout to guide them during this activity.
- ❑ Divide the group into project teams. Optionally, redistribute to students their completed Project Planning, Project Feedback, and Unfocus Group handouts.
- ❑ Give teams time to reflect back on original project goals and acquired feedback. Invite students to outline next steps and plans for project refinement for an upcoming design sprint.

RESOURCES

- ❑ Project Check-In handout

REFLECTION PROMPTS

+ What has been your favorite part of the process so far?
+ What parts of your project still need to be worked on?
+ What parts of your project will you be working on next?
+ What might you need help with in order to make progress?

REVIEWING STUDENT WORK

+ Are teams making reasonable progress and plans?
+ Are group members working cooperatively and collaboratively while discussing and sharing project responsibilities?

NOTES

+ The Project Check-In is a short planning activity. We recommend using it as a warm-up activity at the beginning of each Design Sprint session.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

PROJECT CHECK-IN

CHECK-IN BY: _____

Discuss your design progress with your team and outline a plan for next steps based on feedback.

PROJECT PROGRESS

What has been your favorite part of the process so far?

What parts of your project still need to be worked on?

NEXT STEPS

What parts of your project will each group member be working on next?

What might you need help with in order to make progress?

UNIT 6 ACTIVITY

UNFOCUS GROUP

SUGGESTED TIME
30–45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ interview, observe, and ask others for feedback on projects-in-progress

ACTIVITY DESCRIPTION

- ❑ Hosting an unfocus group is an idea we borrow from IDEO. Introduce the unfocus group concept, where students will share their projects-in-progress and request feedback from a diverse collection of people.
- ❑ Optionally, distribute the Unfocus Group handout to each person.
- ❑ Help students brainstorm possible unfocus group participants. Encourage them to consider their target audience as well as unusual users or unexpected cases who can offer a unique perspective or interesting feedback (e.g., parents, teachers, siblings, other students, community members).
- ❑ Give students time to identify, interview, observe, and record feedback from two unfocus group members.
- ❑ Allow students time to meet with their project team members to share feedback collected from different unfocus group sources. Optionally, collect the completed Unfocus Group handouts at the end of this activity to return to students at the beginning of the Project Check-In or Design Sprint sessions.

RESOURCES

- ❑ Unfocus Group handout

REFLECTION PROMPTS

+ Describe your unfocus group participants and why you chose them.
+ How might their ideas influence your project?

REVIEWING STUDENT WORK

+ Did students identify and interview two unfocus group participants?

NOTES

+ Help students get creative in researching and discovering feedback sources. Is there a local game design company that might be interested in helping? Could projects be shared with students from another school?
+ If unfocus group members are not available to be interviewed during the session (e.g., teachers, parents, siblings, community members), you can organize this activity for outside of class time or assign it as homework.

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

UNFOCUS GROUP

PROJECT TITLE: _____

INTERVIEW BY: _____

In this activity, you will interview and observe others to get feedback on your project-in-progress.

IDENTIFY

+ What kinds of people might be able to offer you a unique perspective on your project?
+ Who are two unfocus group members you plan to share your project draft with?

OBSERVE

Share your project with your unfocus group and observe their reactions.

+ What are they getting stuck on?
+ Are they interacting with your project the way you imagined?
+ Are they doing anything surprising?

INTERVIEW

After you observe, interview your group about their experience.

+ What feedback did you receive from your interview?
+ What suggestions, if any, do you plan to incorporate into your project next?

UNIT 6 ACTIVITY

SHOWCASE PREP

SUGGESTED TIME
30–45 MINUTES

OBJECTIVES

By completing this activity, students will:

+ work on their final project drafts and prepare for the final project showcase

ACTIVITY DESCRIPTION

- ❑ Remind students that they will be sharing their projects with each other (and possibly guests) as a way of acknowledging the hard work that has taken place and of reflecting on their experiences. Explain that this session is an opportunity for finalizing their works-in-progress and coming up with a strategy for sharing their projects with others.
- ❑ Give students time to work on their projects and prepare for presenting final drafts at the project showcase. Optionally, collect final works-in-progress into a class studio for ease in presenting. Optionally, invite students to add their projects to the Hackathon studio.
- ❑ Distribute the Project Reflections handout to students and discuss the What?, So what?, Now what? framework as a way for them to present their experiences to others.

RESOURCES

- ❑ Project Reflections handout
- ❑ Hackathon studio
  http://scratch.mit.edu/studios/488267

REFLECTION PROMPTS

+ What is your project?
+ What was your process for developing the project?
+ What do you want to create next?

REVIEWING STUDENT WORK

+ Did each group or individual complete a Project Reflections handout?

NOTES

+ Students may be feeling anxious or stressed about completing their projects. This is an opportunity to remind them that: (1) this experience is just a waypoint on their paths as computational creators, and (2) some types of stress can be good, helping us to focus on our goals and get things done!

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....

PROJECT REFLECTIONS

PROJECT REFLECTIONS BY: _____

Use the prompts below to reflect on your design process.

WHAT?

What is your project?
How does it work? How did you come up with the idea?

SO WHAT?

What was your process for developing the project?
What was interesting, challenging, and surprising? Why?
What did you learn?

NOW WHAT?

What are you most proud of about your project?
What would you change?

UNIT 6 ACTIVITY

SHOWCASE

SUGGESTED TIME
45–60 MINUTES

OBJECTIVES

By completing this activity, students will:

+ share their final projects with others and reflect on their overall design process and computational creation experiences

ACTIVITY DESCRIPTION

- ❑ Create a celebratory mood in the space by inviting guests, playing music, hanging decorations, and/or providing snacks.
- ❑ Optionally, use a projector and screen to display projects.
- ❑ Invite students to share their final projects and discuss their design processes with others. Optionally, make student progress visible by having design notebooks and prior projects available.
- ❑ Give students time to reflect on all of their creative computing experiences by reviewing their design journals and responding to the reflection prompts in their design journals or in a group discussion.

RESOURCES

- ❑ projector and screen for presentations (optional)

REFLECTION PROMPTS

+ Look through your design notebook. What types of notes did you take?
+ Which notes were most helpful?
+ What has been your favorite Scratch project to work on so far? Why is it your favorite?
+ What do you want to create next?

REVIEWING STUDENT WORK

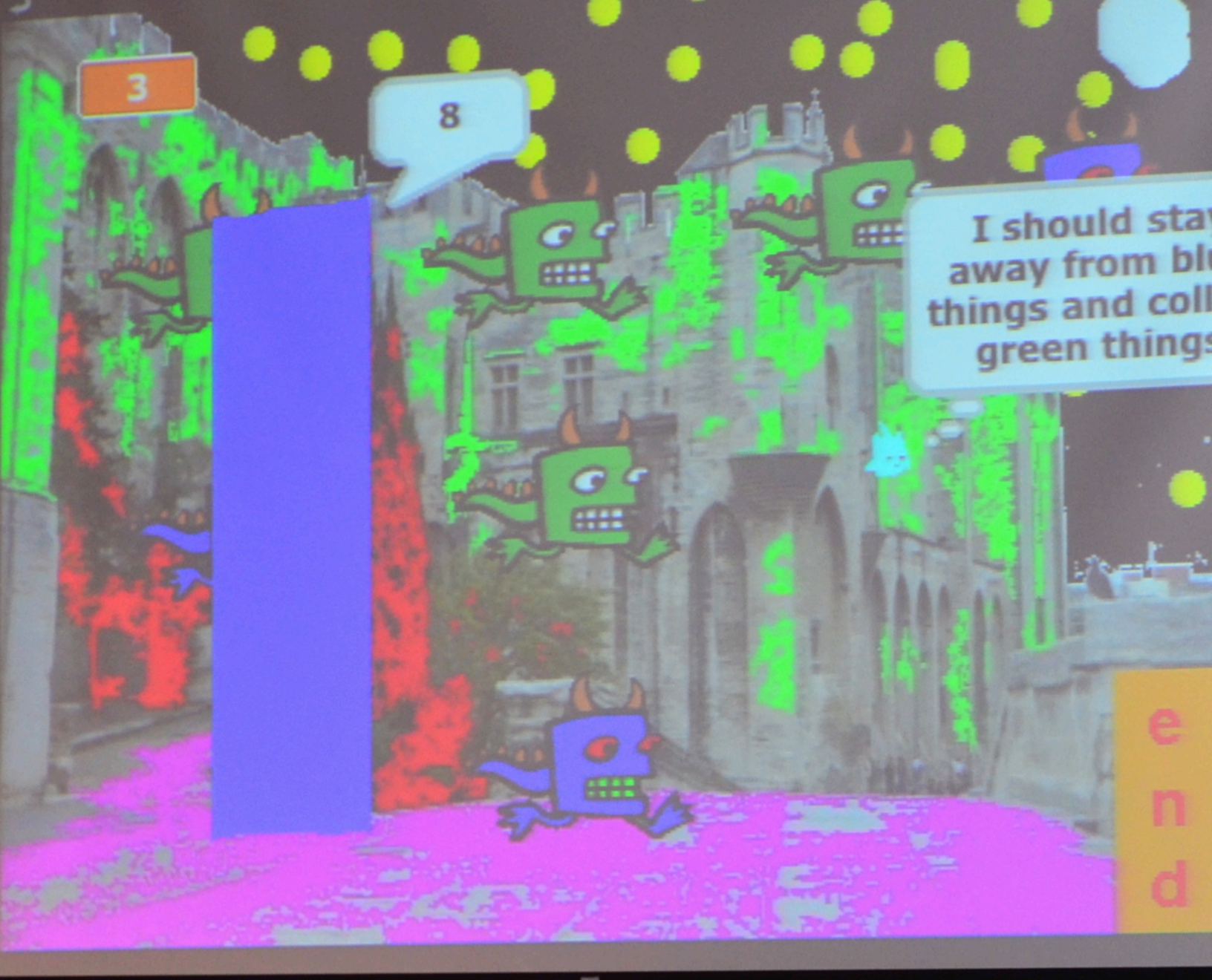
+ Did each team or individual have the opportunity to share their work and be celebrated?

NOTES

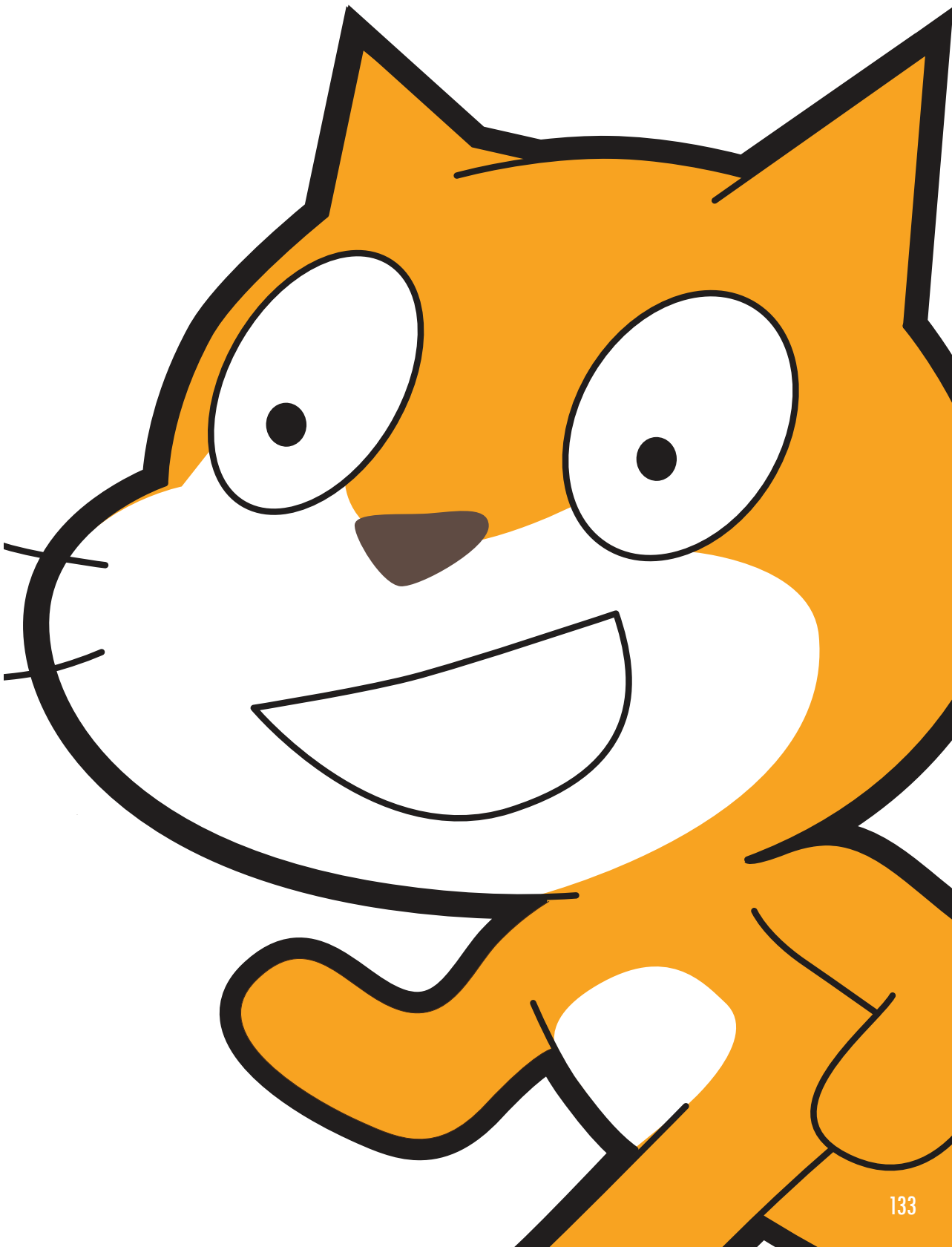
+ Sharing can take place in a variety of ways: individuals presenting to the entire group, concurrent subsets of students presenting, live demos, accessing projects from the web, etc.
+ Project portfolios, design journals, final project feedback handouts, and final project reflection handouts are a few (of many different possible) types of artifacts that may be collected for assessment purposes. (See Appendix.)

NOTES TO SELF

- ❑ .....
- ❑ .....
- ❑ .....
- ❑ .....


3
8
I should stay
away from blu
things and coll
green things
end

APPENDIX



GLOSSARY

A guide to the key words, concepts, and practices in the curriculum guide:

Visit the Scratch help pages at http://scratch.mit.edu/help or the community-generated Scratch Wiki at http://wiki.scratch.mit.edu for additional, Scratch-specific terminology.

abstracting and modularizing: The computational practice of exploring connections between the whole and the parts.

animation: An illusion of continuous motion created by the rapid display of a sequence of still images with incremental differences.

arcade day: A strategy for sharing student work and whole group activity. Students place their finished projects in Presentation Mode and then walk around and engage with each other's work.

backdrop: One out of possibly many frames, or backgrounds, of the Stage.

backpack: A Scratch feature that can be used to conveniently transfer media and/or scripts between projects.

bitmap: An image that is defined by a two-dimensional array (grid) of discrete color values (a.k.a. "pixels"). Contrast with vector graphics.

broadcast: A message that is sent through the Scratch program, activating receiving scripts.

cloning: A Scratch feature that allows a sprite to create duplicates of itself while the project is running.

computational concepts: The concepts designers engage with as they program, such as sequence, loops, conditionals, events, parallelism, operators, and data.

computational perspectives: The broader perspectives that designers may form about world around them through computing – such as expressing themselves, connecting with others, and posing questions about technology's role in the world.

computational practices: The distinctive habits of mind that programmers develop as they work, such as experimenting and iterating, testing and debugging, remixing and reusing work, and abstracting and modularizing.

conditionals: The computational concept of making decisions based on conditions (e.g., current variable values).

control: One of the ten categories of Scratch blocks. They are color-coded gold, and are used to control scripts.

costume: One out of possibly many "frames" or alternate appearances of a sprite. A sprite can change its look to any of its costumes.

critique group: A group of designers who share ideas and test projects-in-progress with one another in order to get feedback on how to further develop their projects.

data: The computational concept of storing, retrieving, and updating values.

design demo: An activity in which students are invited to present their work to the class and demonstrate how they implemented a particular block, skill, or design strategy within their project.

design sprint: A specified amount of time dedicated to working intensely on developing projects.

events: The computational concept of one thing causing another thing to happen.

experimenting and iterating: The computational practice of developing a little bit, then trying it out, then developing some more.

feedback fair: A sharing activity in which half of your students stay in their seats with their projects open while the other half walks around exploring projects, asking questions, and giving feedback. Once complete, the students then switch sides and start the process over.

gallery walk: A sharing activity in which students put their projects in presentation mode and then walk around and explore each other's projects.

hardware and extensions: Supplemental materials that connect the digital world of Scratch with the physical world. Examples of hardware extensions include: LEGO WeDo, PicoBoard, and MaKey MaKey.

interactive collage: A Scratch project that incorporates a variety of clickable sprites.

looks: One of the ten categories of Scratch blocks. They are color-coded purple, and are used to control a sprite's appearance.

loops: The computational concept of running the same sequence multiple times.

make a block: A feature found within the More Blocks category that allows students to create and define their own custom block or procedure.

motion: One of the ten categories of Scratch blocks. They are color-coded medium-blue, and are used to control a sprite's movement.

operators: The computational concept of supporting mathematical and logical expressions.

paint editor: Scratch's built-in image editor. Many Scratchers create their own sprites, costumes, and backdrops using it.

pair programming: A programming methodology in which developers pair up and work side-by-side on a project.

parallelism: The computational concept of making things happen at the same time.

pass-it-on story: A Scratch project that is started by a pair of people, and then passed on to two other pairs to extend and reimagine.

peer interviews: A sharing activity in which students take turns interviewing one another about their processes of reflection, self-assessment, and research.

pitch: An activity in which students either announce a project idea in order to recruit other team members, or promote their interests, skills, and talents in order to be recruited by other teams.

presentation mode: A display mode in Scratch that allows projects to be viewed at an enlarged size. It is accessed by pressing the button on the top left of the Scratch program. This mode is also called full screen mode or enlarged screen.

profile page: A page on the Scratch online community dedicated to displaying information about a Scratch user, such as projects they have created or bookmarked (a.k.a. "favorited").

project editor: A feature of the Scratch online community that allows projects to be modified. This includes the script area (where scripts are assembled), the sprite area (where sprites can be manipulated), and the stage area (where sprites are positioned and where backgrounds can be accessed).

red, yellow, green: A reflection and sharing activity in which individuals identify aspects of their projects as not going well or still needing work ("red"), confusing or contentious ("yellow"), or working well ("green").

remix: A creative work that is derived from an original work (or from another remix). A remix typically introduces new content or stylistic elements, while retaining a degree of similarity to the original work.

reusing and remixing: The computational practice of making something by building on existing projects or ideas.

Scratch screening: A sharing activity in which students gather around to observe each other's Scratch projects.

scripts: One or more Scratch blocks connected together to form a sequence. Scripts begin with an event block that responds to input (e.g., mouse click, broadcast). When triggered, additional blocks connected to the event block are executed one at a time.

sensing: One of the ten categories of Scratch blocks. They are color-coded light-blue, and are used to detect different forms of input (e.g., mouse position) or program state (e.g., sprite position).

sequence: The computational concept of identifying a series of steps for a task.

showcase: A strategy for sharing in which students present their final projects to others and reflect on their design processes and computational creation experiences.

sound: An audio file that can be played in a Scratch project, available by importing from Scratch's built-in sound library, or creating a new recording. Sounds are played by using sound blocks, which control a sound's volume, tempo, and more.

sprite: A media object that performs actions on the stage in a Scratch project.

stage: The background of a Scratch project. The stage can have scripts, backdrops (costumes), and sounds, similar to a sprite.

studio: A user-created gallery in the Scratch online community that can be used to highlight projects contributed by one or many users.

testing and debugging: The computational practice of making sure things work – and finding and solving problems when they arise.

theatre metaphor: A way of describing the design of Scratch that emphasizes its intentional similarity to theatre, with actors (sprites), costumes, backdrops, scripts, and a stage.

tips window: Built directly into the Project Editor, the Tips Window is a form of getting help in Scratch.

unfocus group: An activity in which students share their projects-in-progress and request feedback from a diverse collection of people.

variables and lists: A changeable value or collection of values recorded in Scratch's memory. Variables can store one value at a time, while lists can store multiple values.

vector graphic: An image that is defined by a collection of geometric shapes (e.g., circles, rectangles) and colors. Contrast with bitmap.

video sensing: A Scratch feature that makes use of video from a webcam to detect motion or display video input on the stage.

STANDARDS

The activities in this guide make connections to several different K-12 curriculum standards, including the Common Core State Standards, the CSTA K-12 Computer Science Standards, and ISTE NETS. We have included connections to Common Core Standards as an example.

For more connections, please visit the guide site at http://scratched.gse.harvard.edu/guide

Common Core State Standards for Mathematics 2010

http://www.corestandards.org/wp-content/uploads/Math_Standards.pdf

+ Make sense of problems and persevere in solving them – Many guide activities engage students in solving debugging challenges, which encourage students to discover different ways of finding and solving problems. *Example activity: Unit 1 - 4 Debug It!*
+ Reason Abstractly and Quantitatively – Students can express abstract concepts and demonstrate their understandings of quantitative relationships such as variables through visual representations designed in Scratch. *Example activity: Unit 4 Score*
+ Model with Mathematics – Certain activities in the guide challenge students to represent previously learned equations, data comparisons, or other mathematical relationships as Scratch programs. *Example activity: Unit 4 Interactions*
+ Attend to precision – On- and off-screen activities help students recognize the importance of attending to detail when specifying instructions or a sequence of code intended to elicit a particular outcome. *Example activity: Unit 1 Programmed to Dance*
+ Look for and Make Use of Structure – Looking through scripts during a debugging challenge, reading through someone else's project code while remixing a project, or reviewing work to build up more complex programs can engage students in looking closely to discern repeated patterns or structure within their own or others' Scratch programs. *Example activity: Unit 3 Conversations*

Common Core State Standards for English Language Arts/Literacy 2010

http://www.corestandards.org/wp-content/uploads/ELA_Standards.pdf

+ They demonstrate independence. – Most activities and projects in the guide are designed to be self-directed or can be easily adjusted to accommodate independent work, although collaborative projects and group work are encouraged. *Example activity: Unit 1 About Me*
+ They respond to the varying demands of audience, task, purpose, and discipline. – Students are made aware of varying types of audience, task, purpose, and discipline when sharing projects to the worldwide Scratch online community or designing projects and activities for others. *Example activity: Unit 5 Activity Design*
+ They comprehend as well as critique. – A variety of feedback exercises and collaborative projects engage students in sharing works-in-progress, asking questions, and exchanging constructive critique. *Example activity: Unit 0 Critique Group*
+ They use technology and digital media strategically and capably. – During self-directed activities, students learn to navigate to different parts of the Scratch website to develop projects, search for inspiration, connect with others, and pursue personal learning goals. *Example activity: Unit 5 Know Want Learn*
+ They come to understand other perspectives and cultures. – In remixing others' projects, students need to read, understand, and interpret the code and intention of work that is not their own. When building up collaborative projects, students learn to cooperate, compromise, and share work with others. *Example Activity: Unit 3 Pass It On*

COMPUTATIONAL THINKING

Over the past several years, we have been captivated by "computational thinking" as a way to describe the learning and development that take place with Scratch. In this section, we share: (1) our definition of computational thinking as a set of concepts, practices, and perspectives, (2) an instrument for assessing student proficiency with computational practices, and (3) a self-reflection instrument to help teachers assess how they support computational practices in the classroom.

These definitions and instruments were developed in collaboration with Wendy Martin, Francisco Cervantes, and Bill Tally from Education Development Center's Center for Children & Technology, and Mitch Resnick from MIT Media Lab. Additional computational thinking resources are available at http://scratched.gse.harvard.edu/ct

COMPUTATIONAL CONCEPTS

CONCEPT	DESCRIPTION
sequence	identifying a series of steps for a task
loops	running the same sequence multiple times
parallelism	making things happen at the same time
events	one thing causing another thing to happen
conditionals	making decisions based on conditions
operators	support for mathematical and logical expressions
data	storing, retrieving, and updating values

COMPUTATIONAL PRACTICES

PRACTICE	DESCRIPTION
experimenting and iterating	developing a little bit, then trying it out, then developing some more
testing and debugging	making sure things work – and finding and solving problems when they arise
reusing and remixing	making something by building on existing projects or ideas
abstracting and modularizing	exploring connections between the whole and the parts

COMPUTATIONAL PERSPECTIVES

PERSPECTIVE	DESCRIPTION
expressing	realizing that computation is a medium of creation "I can create."
connecting	recognizing the power of creating with and for others "I can do different things when I have access to others."
questioning	feeling empowered to ask questions about the world "I can (use computation to) ask questions to make sense of (computational things in) the world."

ASSESSING DEVELOPMENT OF COMPUTATIONAL PRACTICES

The following instrument can be used to assess students' development of fluency with computational thinking practices (experimenting and iterating, testing and debugging, reusing and remixing, abstracting and modularizing). The first column indicates a question for the student (as part of a design journal prompt or interview, for example). The second, third, and fourth columns indicate how low, medium, and high levels of proficiency might be manifested.

EXPERIMENTING AND ITERATING	LOW	MEDIUM	HIGH
Describe how you built your project step by step.	Student provides a basic description of building a project, but no details about a specific project.	Student gives a general example of building a specific project in a certain order.	Student provides details about the different components of a specific project and how they were developed in a certain order.
What different things did you try out as you went along with your project?	Student does not provide specific examples of what s/he tried.	Student gives a general example of trying something in the project.	Student provides specific examples of different things s/he tries in a project.
What revisions did you make and why did you make them?	Student says s/he made no revisions, or only states s/he made revisions but gives no examples.	Student describes one specific revision s/he made to the project.	Student describes the specific things s/he added to the project and why.
Describe different ways you tried to do things in your project, or when you tried to do something new.	Student provides no examples of trying something new.	Student provides an example of trying something new in the project.	Student describes specific new things s/he tried in a project.
TESTING AND DEBUGGING	**LOW**	**MEDIUM**	**HIGH**
Describe what happened when you ran your project that was different from what you wanted.	Student does not describe what was different when s/he ran the project from what s/he wanted.	Student describes what went wrong in the project, but not what s/he wanted it to do.	Student gives a specific example of what happened and what s/he wanted to have happen when s/he ran the project.
Describe how you read through the scripts to investigate the cause of the problem.	Student does not describe a problem.	Student describes reading through the scripts but does not provide a specific example of finding a problem in the code.	Student describes reading through the scripts and provides a specific example of finding a problem in the code.
Describe how you made changes and tested to see what happened.	Student does not describe what problems s/he had or the solution.	Student provides a general example of making a change and testing it out to see if it worked.	This student provides a specific example of making a change and testing it out to see if it worked.
Describe how you considered other ways to solve a problem.	Student does not provide an example of a solution to a problem.	Student provides a general example of a solution to the problem.	This student provides a specific example of a solution to the problem.

REUSING AND REMIXING	LOW	MEDIUM	HIGH
Describe if/how you found inspiration by trying other projects and reading their scripts.	Student does not describe how s/he found ideas or inspiration from other projects.	Student provides a general description of a project that inspired him/her.	Student provides a specific example of project that inspired him/her and how.
How did you select a piece of another project, and adapt it for your project?	Student does not describe how s/he adapted scripts, ideas or resources from other projects.	Student identifies scripts, ideas or resources s/he adapted from other projects.	Student provides specific examples of scripts, ideas or resources s/he adapted from other projects and how.
How did you modify an existing project to improve it, or enhance it?	Student does not describe modifying another project.	Student provides a general description of modifications s/he made to another project.	Student provides specific examples of modifications s/he made to other projects and why.
How did you give credit to people whose work you built on or are inspired by?	Student does not give credit to others.	Student names people whose work inspired him/her.	Student documents in project and/or on the Scratch website the people whose work inspired him/her.
ABSTRACTING AND MODULARIZING	**LOW**	**MEDIUM**	**HIGH**
How did you decide what sprites are needed for your project, and where they should go?	Student provides no description of how s/he selected sprites.	Student provides a general description of deciding to choose certain sprites.	Student provides a specific description of how s/he made decisions about sprites based on goals for the project.
How did you decide what scripts are needed for your project, and what they should do?	Student provides no description of how s/he created scripts.	Student provides a general description of deciding to create certain scripts.	Student provides a specific description of how s/he made decisions about scripts based on goals for the project.
How did you organize the scripts in ways that make sense to you and others?	Student does not describe how s/he organized scripts.	Student provides a general description of how s/he organized the script.	Student provides specific examples of how s/he organized the script and why.

SUPPORTING COMPUTATIONAL PRACTICES IN THE CLASSROOM

The following instrument can be used to help you reflect on how you are supporting computational practices in your learning environment – which may be a classroom, a library, or another learning environment. The purpose of the instrument is to help you notice the types of opportunities to learn that you are designing and supporting.

EXPERIMENTING AND ITERATING: developing a little bit, then trying it out, then developing some more

The activity provided opportunities for students to…	**NONE**	**SOME**	**LOTS**
build a project step by step			
try things out as you go			
make revisions based on what happens			
try different ways to do things, or try new things			

NOTES FOR NEXT TIME:
If **none**, how can I make room, or build time, for more?
If **some**, how can I deepen, or strengthen, those activities?
If **lots**, what have I noticed, or learned?

TESTING AND DEBUGGING: making sure things work – and finding and solving problems when they arise

The activity provided opportunities for students to…	**NONE**	**SOME**	**LOTS**
observe what happens when you run your project			
describe what is different from what you want			
read through the scripts to investigate the cause of the problem			
make changes and test to see what happens			
consider other ways to solve the problem			

NOTES FOR NEXT TIME:
If **none**, how can I make room, or build time, for more?
If **some**, how can I deepen, or strengthen, those activities?
If **lots**, what have I noticed, or learned?

REUSING AND REMIXING: making something by building on existing projects or ideas

The activity provided opportunities for students to…	**NONE**	**SOME**	**LOTS**
find ideas and inspiration by trying other projects and reading the scripts			
select a piece of another project, and adapt it for your project			
modify an existing project to improve or enhance it			
give credit to people whose work you build on or are inspired by			

NOTES FOR NEXT TIME:
If **none**, how can I make room, or build time, for more?
If **some**, how can I deepen, or strengthen, those activities?
If **lots**, what have I noticed, or learned?

ABSTRACTING AND MODULARIZING: exploring connections between the whole and the parts

The activity provided opportunities for students to…	**NONE**	**SOME**	**LOTS**
decide what sprites are needed for your project, and where they should go			
decide what scripts are needed for your project, and what they should do			
organize the scripts in ways that make sense to you and others			

NOTES FOR NEXT TIME:
If **none**, how can I make room, or build time, for more?
If **some**, how can I deepen, or strengthen, those activities?
If **lots**, what have I noticed, or learned?

FOR FURTHER READING

A selection of readings to further support your explorations of creative computing:

Books

+ Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. New York, NY: Basic Books.

+ Papert, S. (1993). The children's machine: Rethinking school in the age of the computer. New York, NY: Basic Books.

+ Kafai, Y. B. (1995). Minds in play: Computer game design as a context for children's learning. Mahwah, NJ: Lawrence Erlbaum. Available at http://www.yasminkafai.com/minds-in-play/

+ Margolis, J., & Fisher, A. (2002). Unlocking the clubhouse: Women in computing. Cambridge, MA: MIT Press.

+ Margolis, J., Estrella, R., Goode, J., Holme, J.J., & Nao, K. (2008). Stuck in the shallow end: Education, race, and computing. Cambridge, MA: MIT Press.

+ Kafai, Y. B., Peppler, K. A., & Chapman, R. N. (2009). The computer clubhouse: Constructionism and creativity in youth communities. New York: Teachers College Press.

+ Rushkoff, D. (2010). Program or be programmed: Ten commands for a digital age. New York, NY: OR Books.

+ Kafai, Y. B., & Burke, Q. (2014). Connected code: Why children need to learn programming. Cambridge, MA: MIT Press.

Dissertations

+ Monroy-Hernandez, A. (2012). Designing for remixing: Supporting an online community of amateur creators. Doctoral dissertation, Massachusetts Institute of Technology.

+ Brennan, K. (2013). Best of both worlds: Issues of structure and agency in computational creation, in and out of schools. Doctoral dissertation, Massachusetts Institute of Technology.

Papers

+ Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. American Educational Research Association meeting, Vancouver, BC, Canada.

+ Brennan, K. (2013). Learning computing through creating and connecting. IEEE Computer, Special Issue: Computing in Education. doi:10.1109/MC.2013.229

LINKS

Links to helpful creative computing resources:

TYPE	DESCRIPTION	LINK
Website	Scratch	http://scratch.mit.edu
Website	ScratchEd	http://scratched.gse.harvard.edu
Website	Flash	http://helpx.adobe.com/flash-player.html
Resource	Offline Version of Scratch	http://scratch.mit.edu/scratch2download
Resource	Scratch Cards	http://scratch.mit.edu/help/cards
Resource	Scratch Community Guidelines	http://scratch.mit.edu/community_guidelines
Resource	Scratch Remix FAQ	http://scratch.mit.edu/help/faq/#remix
Resource	Scratch Wiki	http://wiki.scratch.mit.edu
Resource	Scratch Discussion Forums	http://scratch.mit.edu/discuss
Resource	Scratch FAQ	http://scratch.mit.edu/help/faq
Resource	LEGO WeDo Construction Set	http://bit.ly/LEGOWeDo
Resource	MaKey MaKey	http://makeymakey.com
Resource	PicoBoard	https://www.sparkfun.com/products/10311
Resource	Scratch Design Studio List	http://scratch.mit.edu/users/ScratchDesignStudio
Video	Scratch Overview Video	http://vimeo.com/65583694 http://youtu.be/-SjuiawRMU4
Video	Unit 1 Programmed to Dance Videos	http://vimeo.com/28612347 http://vimeo.com/28612585 http://vimeo.com/28612800 http://vimeo.com/28612970
Video	Backpack Video Tutorial	http://bit.ly/scratchbackpack
Video	Make a Block Video Tutorial	http://bit.ly/makeablock
Video	Variables Video Tutorial	http://bit.ly/scratchvariables
Video	How can I connect Scratch with other technologies? Video Playlist	http://bit.ly/hardwareandextensions
Video	Scratch Chain Reaction Video	http://bit.ly/ScratchChainReaction