BACKGROUND

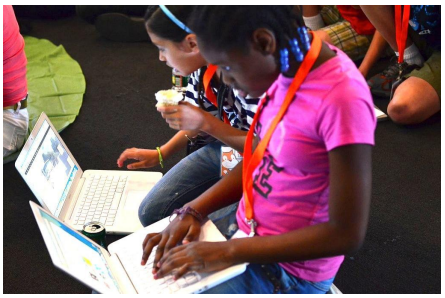
Welcome to the Creative Computing Curriculum Guide!

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

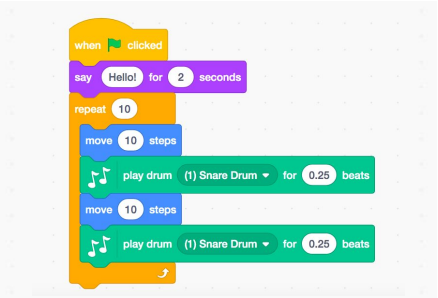
1. What is Creative Computing?
2. What is Scratch?
3. What is this guide?
4. Who is this guide for?
5. What do I need in order to use this guide?
6. What is included in this guide?
7. How should I use this guide?
8. Where did this guide come from?

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.

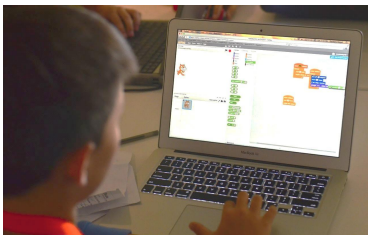


Creative computing is about agency. Many young people with access to computers participate as consumers, rather than designers or creators. Creative computing emphasizes the knowledge, practices, and fundamental literacies that young people need to create the types of dynamic and interactive computational 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:

Creative computing is for everybody!

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.

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.