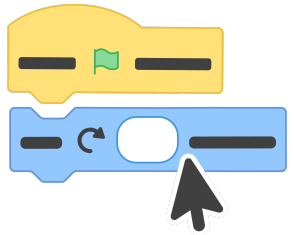
Creative Learning Guide for Families



Scratch pioneered block-based programming, enabling young people to learn to code creatively and interactively. In our blog post "A Parents' and Guardians' Guide to Scratch," we note, "While Scratch can be a great entry point into a lifelong love of Computer Science, **we designed Scratch with every kid in mind, not just those who are inherently curious about programming**. In our increasingly technological world, it's critically important that every child understands how technology works and feels empowered to understand and create, not just consume, the content and technology they interact with each day. **Scratch is a powerful tool for self-expression**, whether your child is interested in art, music, animation, science, mathematics, storytelling, or anything else they can imagine."

The Scratch Foundation creative learning philosophy is aligned with a **constructionist learning model**: We believe that learning is most impactful when making something, whether unplugged or using digital tools. Creating Scratch projects fosters the **development of computational and creative thinking skills** critical for future success:

- **identify problems**
- **break them into smaller parts**
- **debug them**
- **and iterate** on solutions

And, as learners explore with Scratch, they begin to develop skills like persistence, collaboration, and self-reliance, in addition to problem solving — essential skills for everyone in today's society.

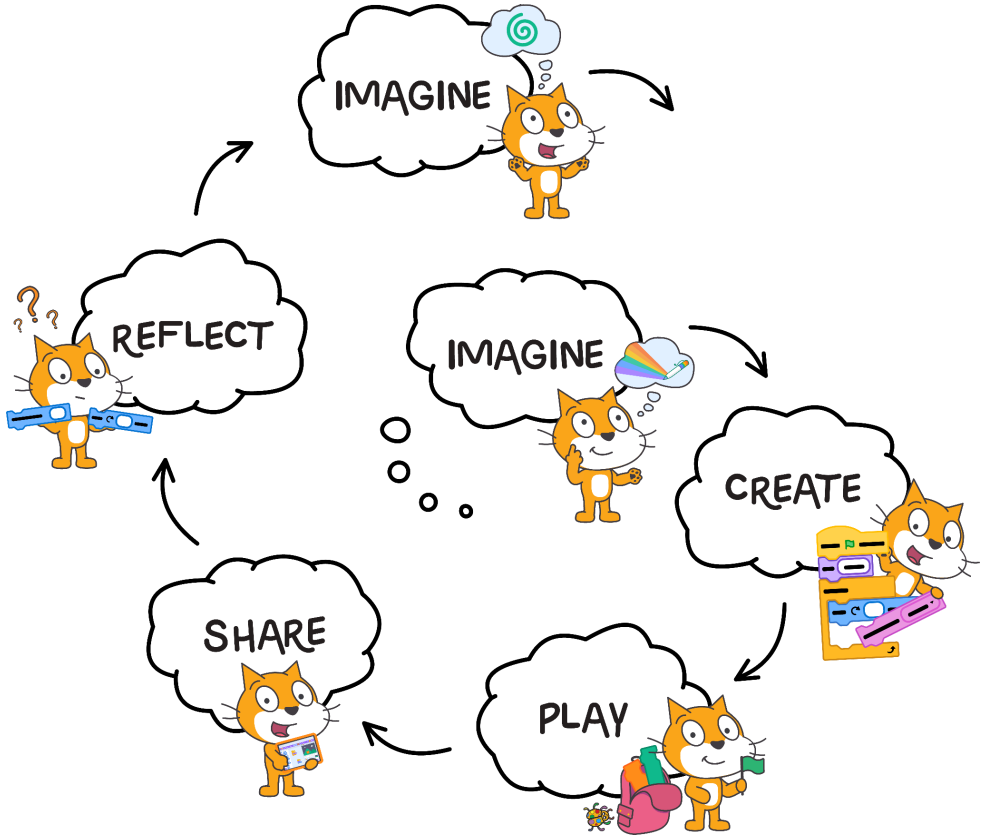
As a family member interested in engaging in creative learning and creative coding with your learner, we encourage you to support playful learning and tinkering mindset values by:

- letting them engage playfully in projects that are **meaningful** to them and **elicit joy**
- helping them develop a mindset that is comfortable with the discomfort of getting stuck (**making room for risk and iteration**)
- and helping them develop a mindset that thinks critically about strategies for getting unstuck (**saving space for the process to start again or help them imagine what's next**)

The Creative Learning Spiral

In his blog post "Ten Tips for Cultivating Creativity (excerpt from Lifelong Kindergarten)," Mitch Resnick (Professor of Learning Research at the MIT Media Lab, director of Lifelong Kindergarten research group, and founder of the Scratch project) shares, "Although it's certainly true that children are naturally curious and inquisitive, they **need support to develop their creative capacities and reach their full creative potential**....As children move through the [creative learning spiral], they get new ideas and continue to the next iteration of the spiral, with another cycle of imagining, creating, playing, sharing, and reflecting."

We encourage you to help **guide your learner through the creative thinking process**.



Tips for Creating Together

1. Find a dedicated time

Pick a time when you and your child can be together. You could either set aside an hour or two on a weekend, or take advantage of a rainy day or unexpected downtime. Whatever time you choose, try to minimize distractions to maximize being creative together. Consider picking a recurring time to cultivate routines that help you both get into a creative spirit. ***(See our Getting Started with Scratch resources and our Learning Library to learn about the platform and locate tutorials, starter projects, and coding cards for inspiration.)***

2. Brainstorm ideas together

People are most engaged when they pursue projects that are personally meaningful to them. What kinds of things are personally meaningful to you and your child? Spend some time thinking about or writing down ideas about what you both enjoy. You might want to create a story together, recreate a family trip, or create a project centered around a few items you love. One brainstorming method you could try is writing down as many ideas as you can on post-its within a set time period (3–5 minutes), then grouping the post-its and ranking them based on favorite choices. ***(Our Scratch Design Journal could be a helpful place to jot down ideas.)***

3. Let your child drive

Learning happens best when literally experienced first-hand. Empower your child to control the mouse or keyboard to encourage them to try things out on their own. If you have to grab the tools to troubleshoot or point something out, you can ask them to try the steps again on their own to practice. If your child is particularly nervous about controlling the tools, you might offer the option of having your child spend some time brainstorming ideas or creating objects in the physical world that accompany your Scratch project before diving into hands-on experience.

*From our blog post "Creating Together: How families can facilitate collaboration when coding with Scratch"

4. It's okay to not have all the answers

"Failures" are a natural and celebrated part of the tinkering process. Start your project with the mentality that you are both newcomers, learning and exploring together. With this mindset, "mistakes" and "failures" become opportunities for learning something new or unexpected, and you and your child will be able to practice fearless exploration. If you don't know the answer to a question, you can work together to try to find a solution. You might stumble upon something surprising! ***(Our Debugging resources provide tips and tricks for identifying problems and finding solutions.)***

Try these prompts to support deepening their critical thinking, self efficacy, and modeling collaborative problem solving to help them "get unstuck" when faced with program bugs or challenging issues.*

"I love it! What is it?"

"Let's test it out."

"Can you explain what your program does?"

"Walk me through your code. What does it say?"

"What do you want your program to do?"

"Can you tell me more about that?"

"What new things did you try out?"

"Which block category do you think would be helpful?"

"I don't know, but let's find out together."

"What are your next steps for this project?"

* Inspired by the Creative Computing Lab "Getting Unstuck Curriculum."

5. Learn from each other

There's a lot to discover about your child, especially when approaching working together with a curious mind. You may uncover new interests, develop new routines, or reveal undiscovered skills. Be open to switching roles as well — your child can spend time as the leader or teacher, guiding you through the activity. Or, one of you may serve as the "creative director" while the other builds. Either way, make sure to give each other room to try different roles throughout your time together.

Tips for Facilitating Collaboration Between Siblings or Peers

1. Help guide a brainstorming session

Any collaborative project can be difficult to get started. When overseeing a project between siblings or peers, help young programmers figure out what they want to create by guiding a brainstorming session and offering general ideas. Ask them what they are passionate about, or about a recent event that happened in their lives. Help them think of something they have in common, like a love for sports, or a favorite movie or book. Honing in on something personally meaningful to each programmer is a great place to start when planning out a Scratch or ScratchJr project! ***(Our Scratch Design Journal could be a helpful place to jot down ideas.)***

2. Suggest unique roles

Since the Scratch programming environments exist on-screen, collaborating can seem difficult at first. However, if each team member has a unique role, everyone can contribute in their own unique and creative way. As the adult, you can assist young collaborators by suggesting roles based on their strengths, or assigning different tasks. Roles can include art director, planner, code-writer, code-reviewer/debugger, presenter, etc. If working with siblings or friends of different ages, ask the older programmer to navigate the younger programmer through more complex tasks. Celebrate each contribution — everyone's work will be crucial in the final project!

3. Incorporate off-screen activities

We also recommend incorporating creative off-screen activities when working on a project. As the adult facilitator, you can help guide these activities by suggesting peers or siblings draw out their programs before creating them on-screen, or create a storyboard of what they want their project to be. Structure project creation by timing who gets to work on-screen, and switch the "head programmer" after an amount of time. You can also facilitate a "telephone" game, encouraging one child to start a project, then after 5 minutes, the next child adds their own ideas to it, and so on! ***(Our Scratch Design Journal could be a helpful place to storyboard.)***

4. Encourage collaborative problem solving

Encourage children to solve problems they encounter on their own before seeking out parent help. Teach them the meaning of "troubleshooting" and "debugging," and ask that they try at least 3 things together to solve the problem before asking an adult. This will help peers or siblings work collaboratively — they might be surprised what they can figure out together! ***(Our Debugging resources provide tips and tricks for identifying problems and finding solutions.)***

5. Foster older and younger sibling collaboration

If you have children of different ages, the older sibling might be programming in Scratch and the younger might be using ScratchJr — that's wonderful! Older and younger siblings can both work in ScratchJr together, and that's a great place to start collaborating. The older sibling can pass along programming tips, ask questions, and help the younger sibling decide what to create. Using the Paint Editor Tool, siblings can draw and take pictures of characters and backgrounds that are personally meaningful to them, and program a story they both care about. Suggest "story-starters" that they can both have fun with using ScratchJr, like re-creating a favorite family vacation, or incorporating family members into a fairy tale or movie they both love! ***(See our Learning Library to locate tutorials, starter projects, and coding cards for inspiration and projects to work on together.)***

Start with Exploration

The blog post "Start with Exploration, Not Explanation" by Natalie Rusk, a research scientist who was one of the creators of the Scratch programming language, offers this advice:

> *[We] developed Scratch for learning through exploring, experimenting, and tinkering. Scratch coding blocks are designed like LEGO bricks....children learn by putting blocks together and taking them apart..."playing" or "messing around" with it, trying things out and seeing what worked. This playful approach helped them build their confidence in their ability to learn and problem solve.*

Embrace Multiple Pathways, Multiple Solutions

The blog post "There's More Than One Way to Code a Cat" by Natalie Rusk, offers this advice:

> *"How do I make my character jump?" a student asks while coding a project in Scratch. Before responding, I find it's helpful to ask what they have in mind, so they can think aloud about the process. Talking out their idea is often enough to help them to figure out what to do next. It's interesting how often students will come up with a way to code that is different than one I might have suggested, but ends up working the way they want.*
>
> *If I do suggest example code to look at...I feel it's important to acknowledge that there are multiple possible approaches, saying something like, "Here's one way to do it, see if it does what you want."...Focusing on a single pathway not only limits the creative potential of coding, it also limits who becomes interested in coding and decides to learn more....Each day we see how opening up possibilities provides motivation and meaning for more students to continue learning, creating, and sharing their ideas and interests with others.*

Computational Fluency and Self-Expression

The blog post "Computational Fluency (excerpt from Lifelong Kindergarten)" by Mitch Resnick, offers this advice:

> *Our approach with Scratch is distinct from many introductions to coding in that we focus explicitly on helping children learn to express themselves creatively through coding. Most introductions to coding are based on puzzles. Children are asked to create a program to move a virtual character past some obstacles to reach a goal. As children create programs to solve these puzzles, they learn basic coding skills and computer science concepts.*
>
> *With Scratch, we focus on projects instead of puzzles. When we introduce children to Scratch, we encourage them to create interactive stories, games, and animations, based on their own interests....Why focus on projects? We take seriously the analogy between coding and writing. When you learn to write, it's not enough to learn spelling, grammar, and punctuation. It's important to learn to tell stories and communicate your ideas. The same is true for coding....Becoming fluent, whether with writing or coding, helps you to develop your thinking, develop your voice, and develop your identity.*

For More Inspiration...

- See our Getting Started with Scratch resources and our Learning Library to locate tutorials, starter projects, and coding cards for inspiration and projects to work on together. Great ways to encourage exploration might be to start with:
    - Sprite Creation
    - Sound and Music
    - Create a Story
    - Bring Yourself Into Scratch
    - Interactive Art
    - Catch Game
    - Making Faces, Stop Motion
    - Physical Computing with Scratch and Makey Makey
- Creative Learning at Home: Play, Learn, and Connect - Offers a variety of creative activities for children and families to do at home. Available in English, Portuguese (Brazil), and Spanish. Created by our friends at the Brazilian Creative Learning Network, with support from LEGO Foundation and Lemann Foundation.
- Family Creative Learning - Family Creative Learning is a workshop series that engages children and their families to learn together — as designers and inventors — through the use of creative technologies. Workshops were designed to build on families' relationships and cultural backgrounds and to strengthen their social support and expertise around computing.