

A decorative graphic on the left side of the slide features several overlapping, colorful, rounded rectangular bars and circles. The colors include orange, red, green, purple, blue, and pink. The bars are oriented in various directions, creating a dynamic, abstract composition.

Facilitating multiple programming languages in one space

[JaredOLEary.com/present](https://jaredoleary.com/present)

July 13th, 2023

How to reach the resources

- Direct link is in the chat
- JaredOLEary.com
 - Presentations
 - Facilitating Multiple Programming Languages in One Space

Speaker



Jared O'Leary

JaredOLEary.com

How do we differentiate for students who...

- ...are required to attend and aren't interested in CS?
- ...are already pursuing a career path outside of CS?
- ...have a wide range of CS experience?
- ...have varying access to devices/internet at home?
- ...have various accessibility needs/accommodations?
- ...have variegated identities and interests?
- ...want to impact the world in ways we haven't considered?



Sequential design

Step 1

Step 2

Step 3



Rhizomatic design






Fixed



Open





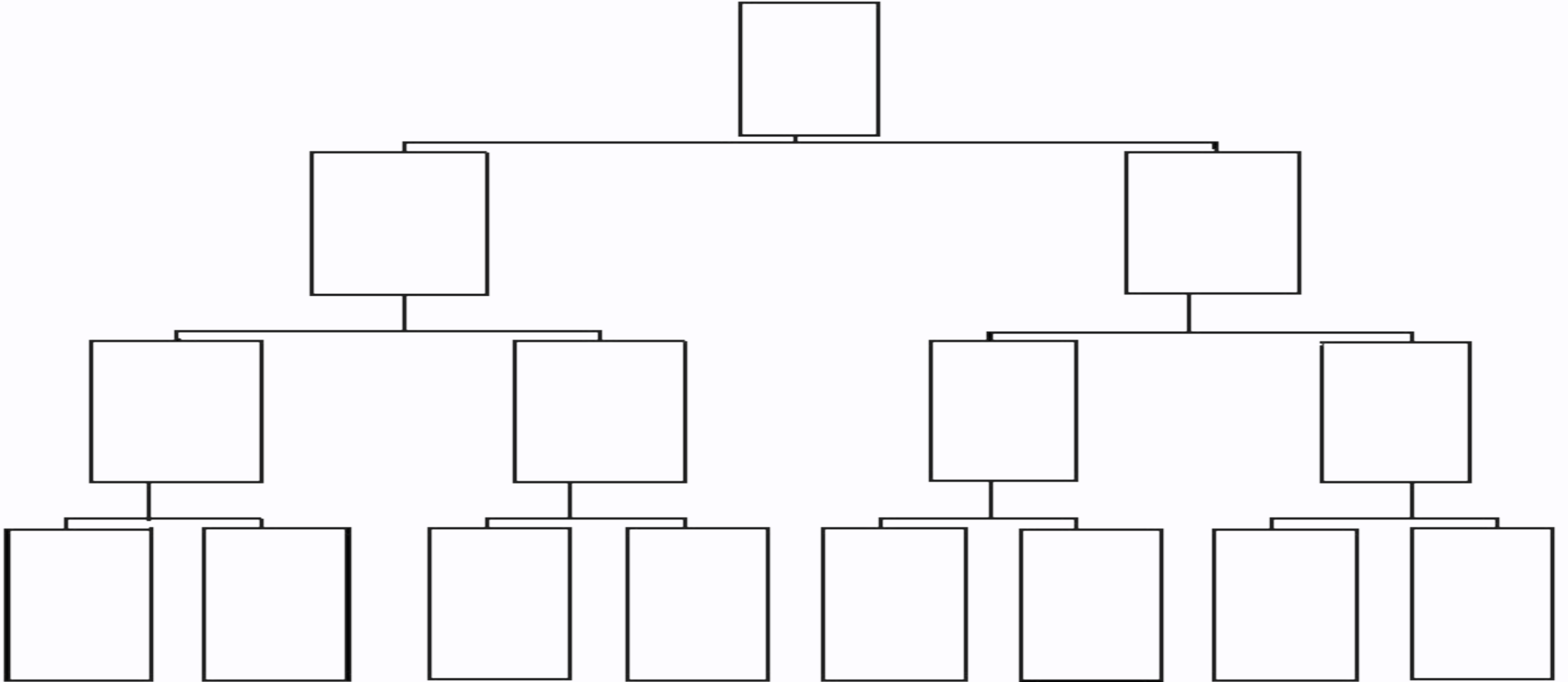
Technology Classes at Desert Thunder

Jared O'Leary
Arizona State University
Avondale Elementary School District

[Playlist with more videos of the classes I facilitated](#)



Start small and gradually expand



Selecting a language/platform

- Does it afford new creative opportunities?
 - Are kids interested exploring their interests in that way?
- How well do I currently know the language/platform?
 - How quickly can I learn it?
- How abundant are student-facing resources?
 - Are they self-guided or will students require more support?
- Are there multiple entry points?
 - Does it have low floors, high ceilings, and wide walls?



Implementation suggestions

- Start with a trial period
- Flexibility with switching languages/platforms
- Flexibility with deadlines
- Supporting just-in-time learning



If you need help

1. Check the built-in help or resources
2. Ask a friend for help
3. Ask another friend for help
4. If I'm not working with someone, ask me
 - a. If I'm working with someone, repeat steps 1-3



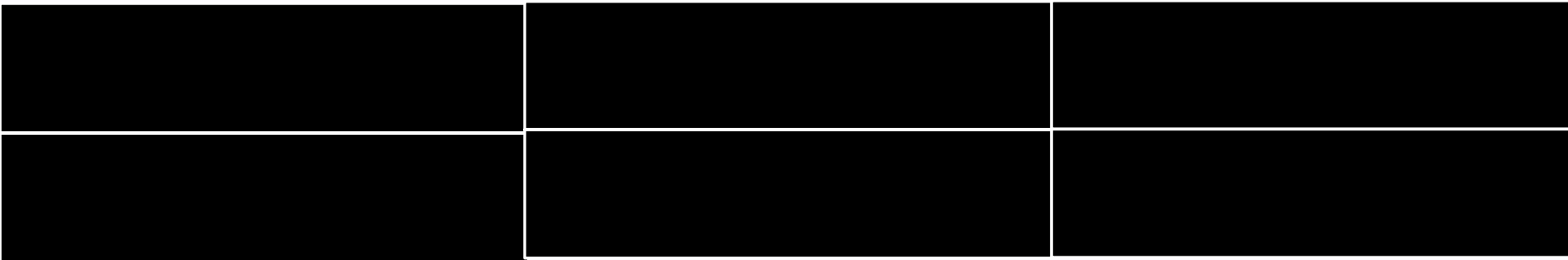
193: Critical Response Process



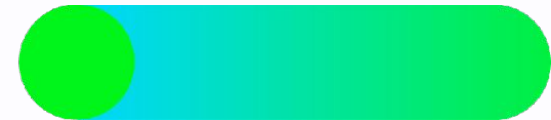
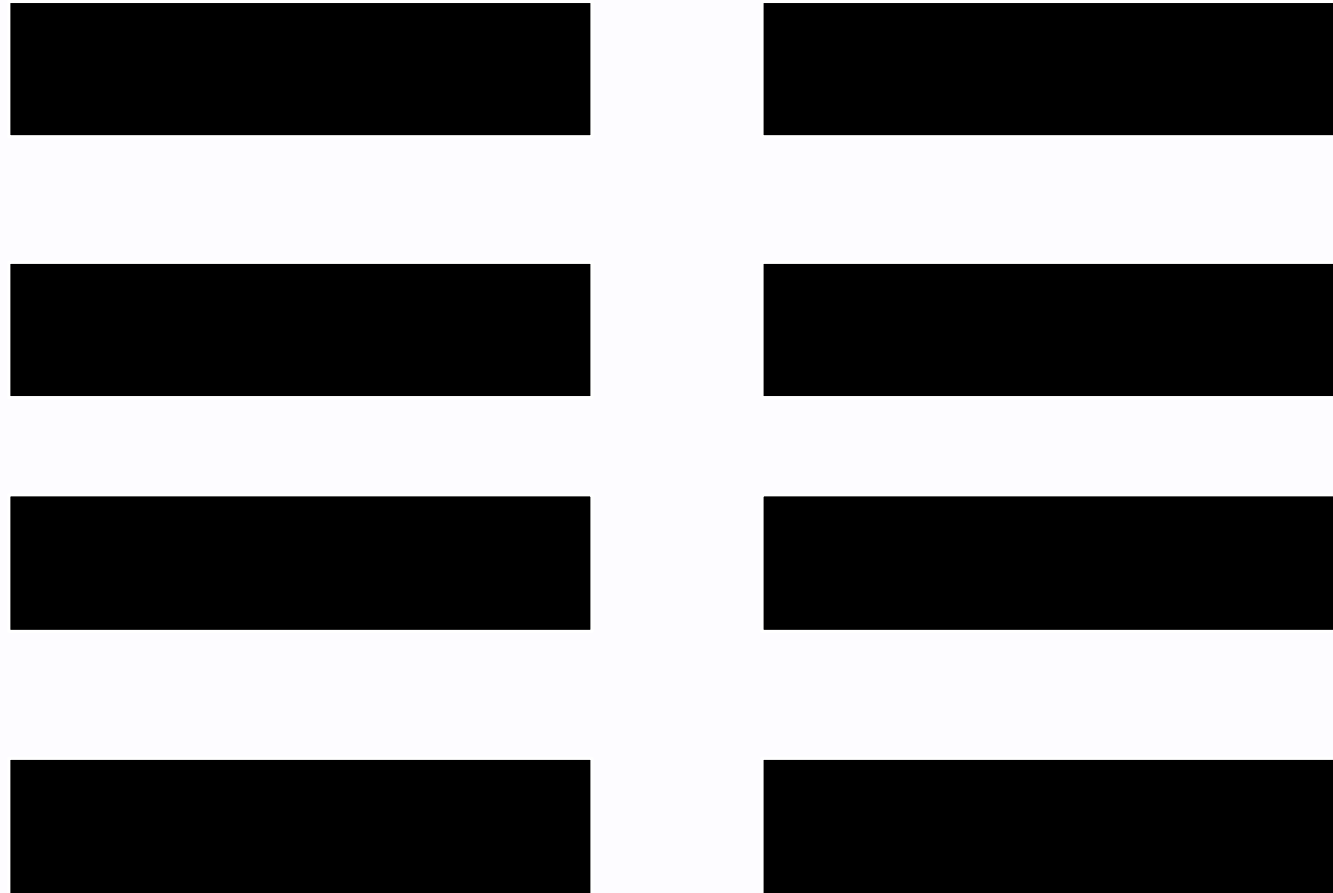
[All podcast episodes](#)



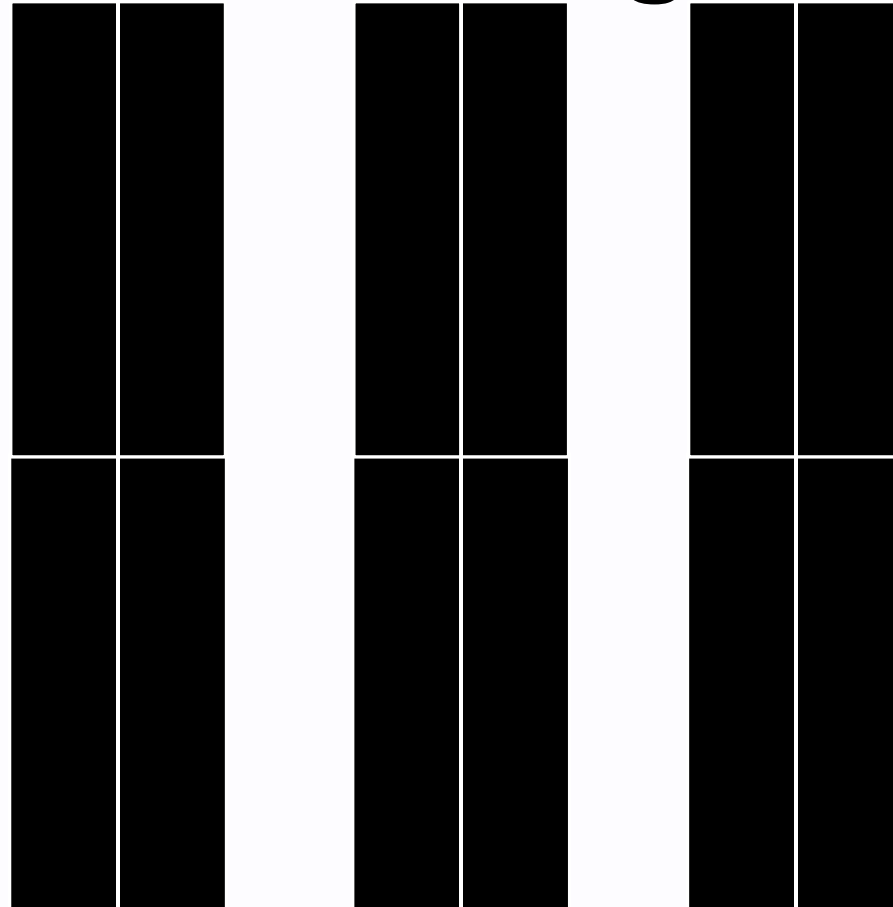
Room setups: Racetrack



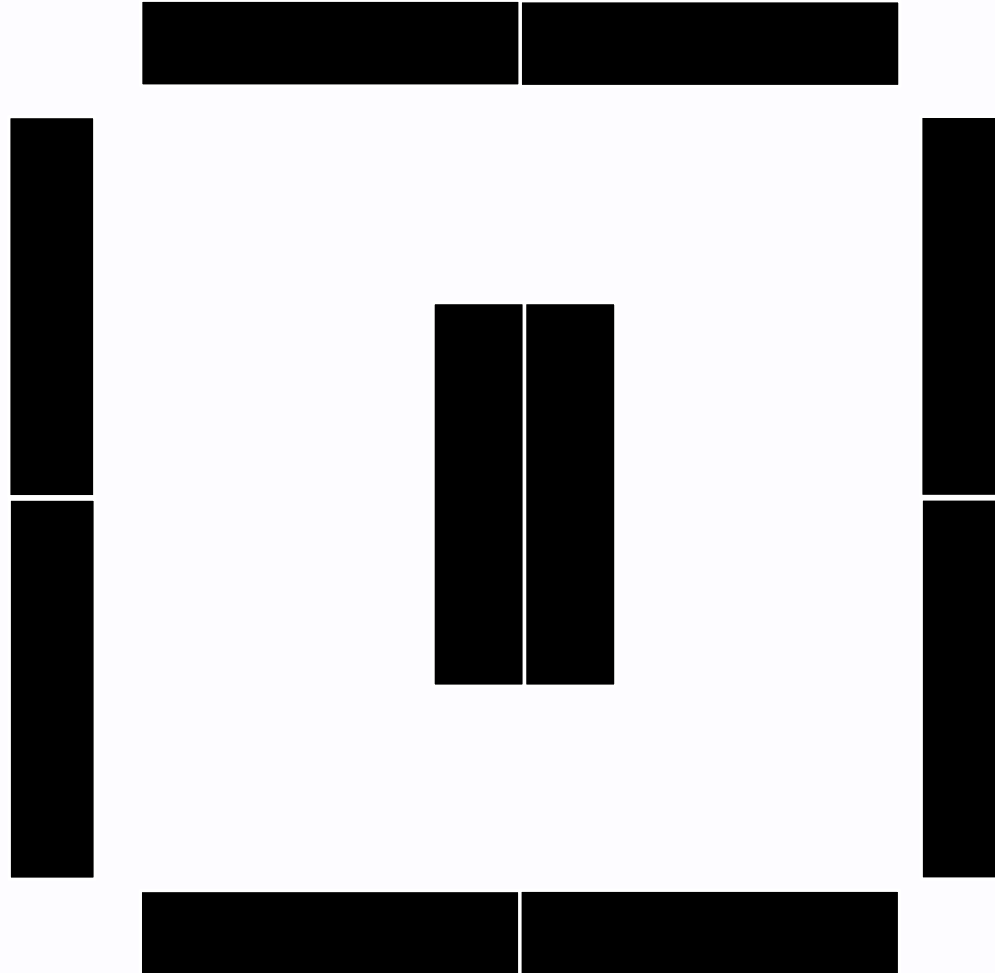
Room setups: Rows facing front



Room setups: Columns facing inward



Room setups: Donut



Example questions for different phases of learning:

Starter questions	Questions to stimulate computational thinking	Assessment questions	Final discussion questions
<ul style="list-style-type: none">• How many ways can we solve this bug or problem?• What happens if we reversed the order of this algorithm?• What kind of project can we create with conditionals?	<ul style="list-style-type: none">• When looking at two different projects or chunks of code: What's the same? What's different?• What patterns do you notice?• If you were to break this down into different pieces, how would you group or label each part of the code?• What do you think comes next in the code? Why?• What do you think comes before this code? Why?	<ul style="list-style-type: none">• What have you discovered?• How did you find that out?• Why do you think that?• What made you decide to do it that way?	<ul style="list-style-type: none">• Who solved the bug or problem in a similar way?<ul style="list-style-type: none">○ Who has a different solution or algorithm?• Do we all have the same code?<ul style="list-style-type: none">○ Why/why not?• Are there other ways to solve this bug or problem?<ul style="list-style-type: none">○ How do you know there are or aren't?• How is or isn't your solution the best solution?<ul style="list-style-type: none">○ What is it the best at?○ What do other algorithms do better than the one you chose?

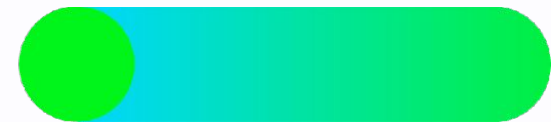
[Using Questions That Guide Mathematical Thinking to Think Computationally](#)



Example questions around levels of thinking:

Memory	Interpretation	Analysis	Evaluation
<ul style="list-style-type: none">• What projects or code have you previously worked on that might assist with this bug, problem, or project? <p style="text-align: center;">Translation</p> <ul style="list-style-type: none">• Without showing your code to someone, how would you explain how this works?<ul style="list-style-type: none">○ How would you explain it if the other person didn't know how to code but wanted to understand how your code worked?	<ul style="list-style-type: none">• When looking at another person's algorithm, can you explain what's similar and different between your algorithm and theirs?<ul style="list-style-type: none">○ What patterns do you notice?• How could you group your code or functions differently? <p style="text-align: center;">Application</p> <ul style="list-style-type: none">• How does your code solve the bug or problem?• What code should come next? Why?	<ul style="list-style-type: none">• What did you discover or learn in this algorithm?<ul style="list-style-type: none">○ How did you figure that out?○ Why do you think that?• What made you decide to order your algorithm that way and not another way? <p style="text-align: center;">Synthesis</p> <ul style="list-style-type: none">• Who has a different solution to the bug or problem?<ul style="list-style-type: none">○ Are their answers the same as yours? Why or why not?• How did your understandings from prior projects or bugs inform your code for this bug, problem, or project?	<ul style="list-style-type: none">• Have we found all of the possible solutions to the bug or problem?<ul style="list-style-type: none">○ How do we know if we have?○ Are there other ways to solve the bug or problem?• Is this the best solution?<ul style="list-style-type: none">○ What is this solution the best at and what is it not the best at?

[Using Questions That Guide Mathematical Thinking to Think Computationally](#)



Three types of questions:

Open

- After a student runs the program, ask what do you think?
- What does this program or project do?
- How would you describe your project to a friend?
- If you were to create a new project based on this one, what would you do?

Guided

- If making a game, you might ask what makes this game fun to play?
- If making an animation with code, you might ask how did your code work together to animate a sprite?
- Or even asking questions about specific concepts
 - How might you include variables in your projects? What about conditionals?

Closed

- Nobody mentioned the use of conditionals in this project, what are the conditionals used for?
- Why did we need a variable in this particular function?
- How many functions are running in parallel to animate this sprite?

[Talking About \[Computer Science\]: Better Questions? Better Discussions!](#)



Three more types of questions:

Analytical

- Nobody mentioned the use of conditionals in this project, what are the conditionals used for?
- Why did we need a variable in this particular function?
- How many functions are running in parallel to animate this sprite?

Judicial

- What was your favorite part of the story made in Scratch?
- What part of this program is interesting to you?
- Why did this program work?

Creative

- What would you do differently if you were to create a similar project?
- What code would you keep and what would you change?
- Now that you've completed this project, what can we do with what you learned?

[Talking About \[Computer Science\]: Better Questions? Better Discussions!](#)



Podcast episodes on questioning techniques

[Talking About \[Computer Science\]: Better Questions? Better Discussions!](#)

- In this episode I unpack Allsup and Baxter's (2004) publication titled "Talking about music: Better questions? Better discussions!" which is a short article that discusses open, guided, and closed questions, as well as a framework for encouraging critical thinking through questions. Although this article is published in a music education journal, I discuss potential implications for computer science educators.

[Thinking through a Lesson: Successfully Implementing High-level Tasks](#)

- In this episode I unpack Smith, Bill, and Hughes' (2008) publication titled "Thinking through a lesson: Successfully implementing high-level tasks," which provides a heuristic that can be used to prepare for a lesson.

[Using Questions That Guide Mathematical Thinking to Think Computationally](#)

- In this episode I discuss some example questions we can ask to encourage kids to think deeper about computer science and computational thinking by unpacking two papers on using guiding questions in mathematics education. The first paper by Way (2014) is titled "Using questioning to stimulate mathematical thinking" and the second paper by Pennant (2018) is titled "Developing a classroom culture that supports a problem-solving approach to mathematics."

[More questioning techniques episodes](#)

A tool to help administrators

<p><i>Academic Feedback Original</i></p>	<ol style="list-style-type: none"> 1. Oral and written feedback is consistently academically focused, frequent, and high quality. 2. Feedback is frequently given during guided practice and homework review. 3. The teacher circulates to prompt student thinking, assess each student's progress, and provide individual feedback. 4. Feedback from students is regularly used to monitor and adjust instruction. 5. <u>Teacher engages</u> students in giving specific and high-quality feedback to one another. 	<ol style="list-style-type: none"> 1. Oral and written feedback is mostly academically focused, frequent, and mostly high quality. 2. Feedback is sometimes given during guided practice and homework review. 3. The teacher circulates during instructional activities to support engagement and monitor student work. 4. Feedback from students is sometimes used to monitor and adjust instruction. 	<ol style="list-style-type: none"> 1. The quality and timeliness of feedback is inconsistent. 2. Feedback is rarely given during guided practice and homework review. 3. The teacher circulates during instructional activities, but monitors mostly behavior. 4. Feedback from students is rarely used to monitor or adjust instruction.
<p>Academic Feedback Crosswalk</p>	<ol style="list-style-type: none"> 1. Oral and written feedback is consistently academically focused, frequent, and high quality. 2. Feedback is frequently given during guided practice and review. 3. The teacher circulates to prompt student thinking, assess each student's progress, and provide individual feedback. 4. Feedback from students is regularly used to monitor and adjust classroom and project design. 5. <u>Teacher encourages</u> students in giving specific and high-quality feedback to one another in offline or online formats. 	<ol style="list-style-type: none"> 1. Oral and written feedback is mostly academically focused, frequent, and mostly high quality. 2. Feedback is sometimes given during guided practice and review. 3. The teacher circulates during class to support engagement and monitor student work. 4. Feedback from students is sometimes used to monitor and adjust classroom and project design. 	<ol style="list-style-type: none"> 1. The quality and timeliness of feedback is inconsistent. 2. Feedback is rarely given during guided practice and review. 3. The teacher circulates during class, but monitors mostly behavior. 4. Feedback from students is rarely used to monitor or adjust instruction.
<p>Crosswalk explanation</p>	<p><i>How might academic feedback differ?</i> Academic feedback is largely instantaneous in our coding classes because of the platforms we use. For instance, Code.org, Scratch, and Khan all allow you to check algorithms at any given moment to see if they are working as intended. Because coding teachers know this, our role is generally to encourage more student-to-student feedback than teacher directed feedback. Additionally, both teacher to student and student to student academic feedback can take place online and offline. For instance, students can provide online feedback to each other by asking questions about puzzles/projects or providing suggestions through commenting sections in Scratch. This allows students to provide asynchronous academic feedback across grade levels and across schools; however, face-to-face feedback should also be encouraged. While circulating the room and informally assessing for learning (by asking questions and looking at algorithms), teachers should make decisions about when to provide feedback, when to encourage a student to continue debugging on their own, and when to pair students up to engage in academic feedback in order to solve a particular bug or challenge. Observers should not sit in one location, but should instead follow the teacher during a class to see how one-to-one instruction and feedback occurs. Whether a student is doing well, struggling, or meeting expectations, teachers should briefly explain to observers the choices they are making with instruction and feedback (or intentional lack thereof) as each choice relates to where a student is at in relation to their prior learning and the class/project objectives.</p>		



Podcast episodes on rhizomatic learning

[Rhizomatic Learning with Catherine Bornhorst, Jon Stapleton, and Katie Henry](#)

- In this panel discussion with Catherine Bornhorst, Jon Stapleton, and Katie Henry, we discuss what rhizomatic learning is and looks like in formalized educational spaces, affordances and constraints of rhizomatic learning, how to support individual students within a group setting, standards and rhizomatic learning, why few people know and use rhizomatic learning approaches, how to advocate for and learn more about rhizomatic learning, and much more.

[Fostering Intersectional Identities through Rhizomatic Learning](#)

- In this episode, Jon Stapleton and I read our (2022) publication titled “Fostering intersectional identities through rhizomatic learning,” which uses mapping as a metaphor for individualized learning.

[Applications of Affinity Space Characteristics in \[Computer Science\] Education](#)

- In this episode I unpack my (2020) publication titled “Applications of affinity space characteristics in music education,” which has twelve characteristics of informal learning spaces that I will discuss in relation to computer science education.

[More rhizomatic learning episodes](#)

192: How to Get Started with CS Education



[All podcast episodes](#)



A decorative graphic consisting of several thick, rounded rectangular bars and circles in various colors (magenta, cyan, orange, red, green, blue, purple) arranged in a stylized, abstract pattern. The bars and circles are positioned around the central text and extend towards the edges of the slide.

Exploration + Q&A

JaredOLEary.com