vidcode

Vidcode Professional Development

Brooklyn Research, August 8th

TIME	ACTIVITY
10:00-10:15 15 mins	**Share: Introductions** • Share names + grade levels + experience teaching CS/coding • What are you bringing to this training? (creativity, openness, experience) • What do you hope to get out of it? (confidence, materials, community) • Namecards: Write your name and the animal that represents your teaching style
10:15-10:30 15 mins	**Explain: What is Vidcode?** • Learn about the motivation for STEM equality. • See some impressive example projects made by kids • Code as a form of self expression • Aesthetic and technical conversations in computational art
10:30-10:40 10 mins	**Explore: A guided tour of the Vidcode tool** • How to sign in to Vidcode • How to create new programs • The portfolio pages • The classroom gallery • The teacher dashboard • The code editor • Sign up for a teacher account if you haven't already
10:40-11:00 20 mins	**Practice: Code Your First Vidcode** • To a computer, a video is just a large set of data. A still image (one frame) is a list of pixel data and a video is a list of frame data. • Follow along • Choose a sample video to recreate, using the video filter blocks and paying close attention to the order of the rules. • In extra time explore the Famous Filters module
11:00-11:45 45 mins	**Collaborate: Make a Meme** • Plan and code a project using "Make a Meme" about how you feel about programming • Share projects

vidcode

11:45-12:15 30 mins	**Explain: What is CS, Coding, JavaScript?** ● Vocabulary ● Vidcode library vs. JavaScript ● Define: Coding, computer science, computational thinking, what do we give students when we teach them how to code?, what level is appropriate for my students? ● Go over: ○ Objects: containers for awesome data sets ○ Property: stuff about the object ○ Data types (strings, numbers) ○ Booleans: true or false ○ Arrays: list of things in order ○ Variable: A holder for data ○ Functions and Methods: actions (verbs) ○ Arguments ○ Loops: code that repeats
12:15-12:30 15 mins	**Explain: Coding Across Subject Areas** ● 3 levels of cross-curricular projects ○ 1. Content, for example replacing or augmenting a PowerPoint ○ 2. Using material, for example using a sine wave to create an animation ○ 3. Supporting research, for example creating simulations using data collected in a science lab
12:30-1:00 30 mins	**Lunch**
1:00-1:45 45 mins	**Collaborate: Example of a curriculum-integrated CS activity** ● Show Vidcode example: PSA: Black Lives Matter https://app.vidcode.io/share/BYDiy6QBbf ● Mini-lesson - HoC: Climate Facts ○ Hook: https://www.youtube.com/watch?v=0aiE_hq-SXE ● Discuss the live graph of Sea Ice Extent from NSIDC: https://nsidc.org/arcticseaicenews/ ● Interpret the graph and draw conclusions about it ● Produce and code a project (Stop Motion or PSA) about a conclusion you can draw from this graph, or how it impacts your life ● Apply what you learned about filters to make this video stand out on social media and get your message across

	• Evaluate how learning to code was different as a stand-alone subject, or integrated into subject area
1:45-2:15 30 mins	**Build: Curriculum Workshop** • **Brainstorm** a Vidcode Activity that supports your subject area • **Storyboard** a 30 second video with a partner on a class topic of your choice • Use own curricular materials as a reference • Use Vidcode lesson plans, scope & sequence • Talk through differentiation and accommodation in the classroom & level of JavaScript complexity for each grade level • If time permits, code an exemplar
2:15-2:30 15 mins	**Share: Curriculum Workshop** • Share curricular materials with each other • Jot challenges on notecards, share solutions • Discuss any anticipated challenges and create solutions together
2:30-2:45 15 mins	**Share: Parking Lot Walk** • Review the questions from today!
2:45-3:00 15 mins	**Debrief** • What did you learn today? • What are you excited about? • Homework: share one of your finished projects with a colleague • What materials can we develop to help you get started? • High fives for completing the day!
With extra time	**Explain: Assessment** • See examples of good code and bad code. • Learn to evaluate projects based on their output. • Learn to identify code that isn't working. • Discussion: Assessment – learn how to use the Vidcode assessment tool to fit the needs of your classroom. How does grading content come into play? How might technical assessments be reviewed in a quantitative and qualitative way?

vidcode

Debugging Guide

<u>Directions:</u> Try each step, in order, to debug your code.

1. **Reread the directions** on the current page of your tutorial
2. **Reread the directions** on previous pages
3. **Make sure** each line of code is on a new line
4. **Read your code line by line** to check your syntax. Are you missing something?
    - Does every **open parenthesis** have a **matching closing parenthesis?**
        i. For example, **text("hello!" ;** would not work because it only has an open parenthesis, and needs a matching closing one.
        ii. This is true for curly brackets { } square brackets [ ] and quotations " " as well.
    - Are you missing any **periods or semicolons?**
    - Is **text inside quotation marks**? Any colors, like "red", or messages that you want your text to display, like **"hello everyone!"**, should be inside quotation marks.
        i. For example text("**hello everyone!**"); or
        ii. tint("**green**", 50);
5. **Find out what code is working and what isn't** by isolating your code through commenting
    - Comments are lines of text that the computer doesn't read
        i. */* comment out a section of code */*
        ii. *// comment out a line of code*
6. **Ask your partner**
    - Partner reads code for errors
    - Partner points out errors
7. **Ask a neighbor**
    - Team member reads code for errors
    - Team member gives hints, but doesn't tell!
8. **Ask your librarian**
    - Librarian reads code for errors, or checks the answer in lesson plans
    - Librarian gives hints, but doesn't tell!
9. **Copy your code, refresh the page, and paste** it in the editor
10. **Save your code and start over**. You've got this. :)

vidcode

JavaScript Cheat Sheet

`var item = "anything!";`	Think of variables like food storage containers! You can check what's in these containers, replace them, or put something else in them entirely! When you set `my_text = text("hi!");` `my_text` is a **variable** holding `hi!`
`var colors = ["blue", "red", "green", "pink"];`	An **array** holds data in an ordered list. To get blue out of my array, I would type `color[0];` because **arrays** start counting from 0!
`colors.length;`	Get the **length** of my array. `colors.length;` would return **4.**

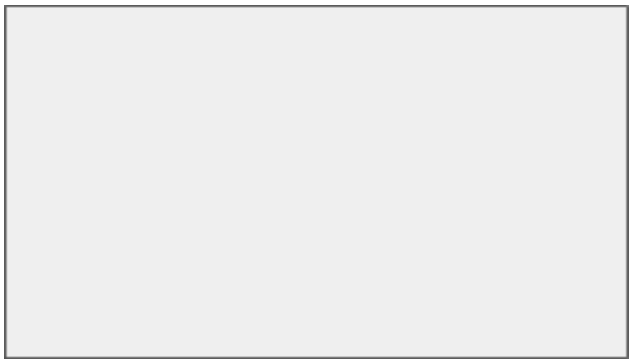
`repeat(function(){`
`  //code that repeats`
`}, 10);`
	Code that gets run over and over again. Using repeat, you can increase or decrease variables to create **animations**, **special effects**, and more!

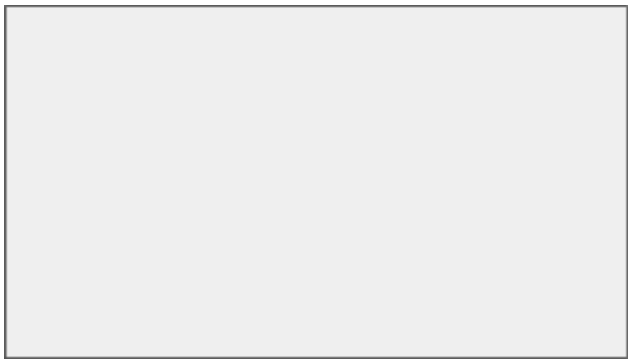
`var functionName = function(){`
`  //your code goes here`
`}`
	A **function** holds an **action** that your code can run.
	For example, the code below moves **any** graphic, text or drawing across the screen:
	`var moveAcrossStage = function(this_graphic){`
	`  this_graphic.x = this_graphic.x + 75;`
	`}`
	This way, I only ever have to write this code once!

`if(movie.currentTime > 1){`
`  //things happen`
`}else{`
`  //other things happen`
`}`
	If-else statements are a way to give instructions to your computer!
	If you see the building with the red door, **then** turn left, **else** keep going!
	If my video is halfway done, **then** make my graphic disappear.
	Put **if-else** statements **inside** repeat.

vidcode

Video Storyboard

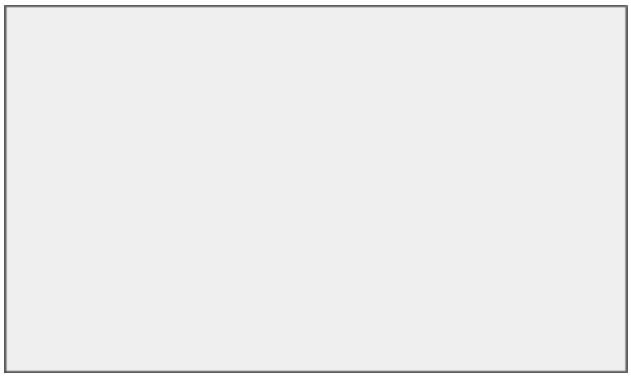


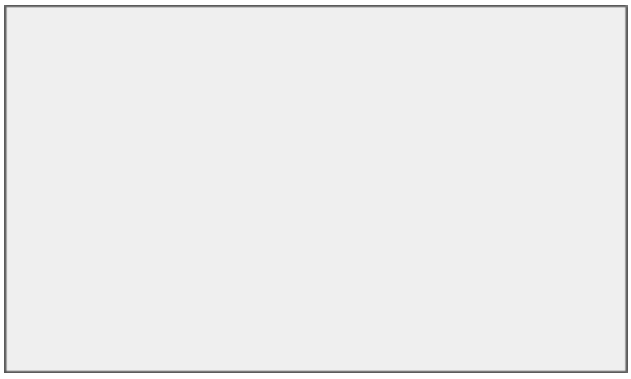


BEGINNING

1. Notes: _____

2. Notes: _____

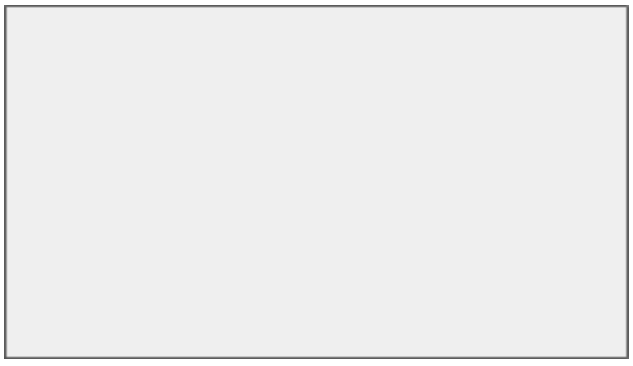


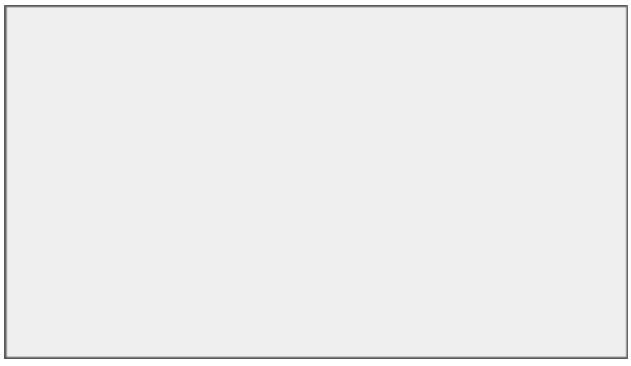


MIDDLE

3. Notes: _____

4. Notes: _____





END

5. Notes:_____

6. Notes:_____

vidcode

Project Assessment Rubric

	Unsatisfactory	Competent	Proficient	Distinguished
Project Content	Project does not convey the required information or understanding.	Project shows some understanding of the subject.	Project reflects understanding of the subject.	Project reflects understanding and synthesis of the subject.
Code Execution	Program does not work, or has major flaws that prevent its intended use.	Program mostly works, and has only minor flaws.	Program works in the way the student intended.	Program is functional and refined, with extra features that exceed the requirements.
Code Practice	Program is difficult to read. Code contains lines that do not work or are out of order.	Program can be read and is in a logical order.	Program is well-organized, easy to read and understand.	Program is well-organized, makes good use of whitespace and comments. Variables have helpful names.
Reflection	Student cannot describe how their code works.	Student can mostly describe how their code works.	Student can describe how their code works and can make changes that have desired effects.	Student can describe how their code works and how they wrote it, and help others debug their code.
Habits of mind	Student is not aware of the goal of the program, is frequently off- task, does not offer their own ideas, and gives up when it is difficult.	Student is aware of the goal of the program, returns to the task when asked, has some ideas when prompted, asks for help when stuck.	Student understands the goal of the program, has their own ideas, rarely goes off-task, and attempts to solve problems first before asking for help.	Student embraces the goal of the program and chooses to try out new ideas and multiple solutions, even when they are challenging.

6th Grade Ancient Civ Vocabulary

Big Idea: Planning and execution make the difference between a memorable meme and a forgettable one.

Module: Make a Meme https://app.vidcode.io/project/graphics

Time: 60 minutes

10 minutes background
20 minutes video production
20 minutes coding
10 minutes sharing

Standards

CCSS.ELA-LITERACY.RH.6-8.4
Determine the meaning of words and phrases as they are used in a text, including vocabulary specific to domains related to history/social studies.
CCSS.ELA-LITERACY.RH.6-8.7
Integrate visual information (e.g., in charts, graphs, photographs, videos, or maps) with other information in print and digital texts.
CCSS.MATH.PRACTICE.MP6 Attend to precision.

CCSS.MATH.PRACTICE.MP7 Look for and make use of structure. CCSS.MATH.PRACTICE.MP8 Look for and express regularity in repeated reasoning.

NGSS Engineering Practice 8 Obtaining, evaluating, and communicating information

Background (10 mins)

A meme is a tiny idea that people like and share because it is funny, weird, or gross.

Ever think about where the word “meme” comes from? It’s related to “memory.” Memes are things that are easy to remember! And what kinds of things do you work hard to remember? Vocab words! In this lesson, we’ll make our vocabulary words easy to remember by turning them into memes.

Watch this video as inspiration: https://www.flocabulary.com/ancient-egypt/

vidcode

Video Challenge (20 mins)

In teams, choose a vocabulary word and shoot a short video about it.

Irrigation *Polytheism* *Empire* *Ziggurat* *Levee* *City-State*

Use the Flocabulary video as inspiration. You don't have to be too literal. Be interesting!

Sample ideas:

Shoot a page in your social studies book.
Make some simple paper puppets and move them around with your hands.
Act out the word in character.
Film yourself writing or typing the definition.
Charades!

Code Challenge (20 mins)

Add text and effects to your video to make it really memorable. Animate a special effect that goes with the word. Add some cool music. Make sure you've conveyed the word and what it means!

Code reference: https://app.vidcode.io/reference

text('I love coding!', 60, 55); *Creates text on your video at position (x,y).*

text.color = "green"; *Changes the color of your text.*

text.size = "50px"; *Changes the size of your text.*

text.font = "Times"; *Changes the font of your text. Possible fonts: "Arial", "Comic Sans MS", "cursive", "serif", "monospace"*

audio("rock"); *Plays music over your video. You can change it to "dance", "electronic", "funk", "rock", or "retro"!*

Sharing (10 mins)

Publish the finished memes, and show them one-by-one on the class projector. Have a face-off: which team did each word better? Vote on the most memorable meme. Turn the videos off and have a pop quiz!

vidcode

Hour of Code: Climate Facts

The goal of this Hour of Code lesson is for students to research and understand a fact about the Earth's climate, engaging with the work of scientists and artists in response to climate change.

Students should take their research, and plan a 30-second video sharing a fact that they learned. They can use props, art they made, their environment and other actors in their videos.

Big Idea: Coding levels up your presentation skills.

Module: Climate Science & Code https://app.vidcode.io/project/hourofcode-science

Time: 60 minutes

25 minutes video production
25 minutes coding
10 minutes reflection

Video Challenge (25 mins)

Before your students start filming, they should plan out what they're going to be creating. Choose a topic: what fact about climate do they want to share?

Have students create a storyboard or write a short script, and use it as their guide. Or, if they want to adlib, have them write a short summary of their fact or topic.

Think about the story: what will your filming environment look like? How many people will you film? What props will you use?

How will you incorporate effects and graphics into your video during editing?

You can choose to record the actual video through the Vidcode interface, or from a program installed on your students' computers, such as PhotoBooth.

Go the the Project Page https://app.vidcode.io/project/hourofcode-science. Students can record their videos directly onto the interface or click the background to exit record, and upload their video with the button on the right.

vidcode

Code Challenge (25 mins)

Once they're happy with their videos, it's time to start coding! They should follow the steps on the left of the screen to go learn how to edit their videos with code.

This can be done individually, or in pairs.

Use the docs or Reference page for more information on how to edit videos with JavaScript.
https://app.vidcode.io/reference

Reflection (10 mins)

After publishing their videos, students can click 'View your Video here' and share the url of the video with their classmates.

Students should talk about what they learned about climate change, and how they used art and code to create their video.

What did they learn? What is JavaScript? What is creative coding? How can they use these things in the future?

vidcode

Programming for Teachers

A letter of encouragement from someone who's been there before.

Hello. It's nice to meet you. I'm really glad you're trying out Computer Science. The prospect may scare you, and that's ok. That's why Vidcode has dedicated years to making coding as engaging and accessible as possible to all learners. And it's also why I'm here with some advice for teachers.

Start with a growth mindset.

In the immortal words of April Murillo, aged 11, "Coding isn't hard, it's just a lot."

And if April can do it, so can you. You've already taken the most important step: you're trying. If your students can see you learning something new at your age, they will grow into lifelong learners themselves.

Read the directions.

Practice what you preach, Teach! Read the whole step before changing your code. Read it again. Make sure you understand it. The things you actually have to *do* are in purple. Make sure you've done it properly and checked your work before moving on to the next step!

Mistakes show you're trying.

Take your code one line at a time. Test it. Make sure it does exactly what you want it to do. As soon as something breaks, stop. Ask for help. Show your class your mistake. Get them to solve it. Celebrate when you've squashed that bug. Debugging is a legitimate use of learning time. Remember, we care more about the process than the product. Or to quote Miley Cyrus, "Ain't about how fast I get there. Ain't about what's waiting on the other side. It's the climb."

Anyone can be an expert.

Even kids. Kids have the luxury of time. They've probably messed around with stuff like this way more than you have, which means they made more mistakes, and therefore they have more experience. Learn from them. Let them help you. They will be proud of themselves. They may even work harder so they can show you up again in the future.

vidcode

Stay on-task.

At some point, all students will have to try out all the different videos, filters, and graphics. They will even legitimately have to search for images on the internet or choose emojis. But draw the line. “You have two minutes to choose, or else you have to use the dog.” A good rule of thumb (or heuristic) is that as you make your rounds, you shouldn’t see them off Vidcode twice in a row. You can always check the step number at the top of their screen to see if they’re falling behind the rest.

Go back.

Program not working? Don’t know why? Start over! There’s no shame in starting from the beginning. There’s even an old programmer adage that it’s good luck to accidentally delete your code. You won’t have to make the same mistakes again, and it will end up better.

Copy your code.

Let’s say you look at a problem and think, “I’ve solved this before.” Don’t waste your time solving it again! Go back to your old project and make a copy! There is sufficient value in identifying and adapting reusable code that it’s inefficient to keep typing everything from scratch.

We’re stronger together.

At some point, a student will call you over for help. You will stare at the screen. It’s not working, but you have no idea why. You try everything you can think of, but nothing helps. You start to panic.

What are you, some kind of superhero? Give yourself a break! I have a PhD in CS and this happens to me every day. Call another kid over and have them work on it together. Project it up on the screen and have the class shout out possible solutions. Worst case scenario, just delete stuff until it works. It’s probably something dumb and impossible to see, like they used double quotes on one side and two single quotes on the other. And if they have to start over, remind them of Miley Cyrus.

You’ve got this.

You really do. I have faith in you.

Happy coding!
Dr. Em