MAKER ED

CYBER ARCADE

PROGRAMMING AND MAKING WITH MICRO:BIT

Educator Curriculum Guide

Middle School Version (Grades 6–8)

ACKNOWLEDGEMENTS

This curriculum was developed in collaboration with Sarah H. Chung (curriculum development and photography), Dora Medrano Ramos (Maker Ed project lead), Goli Mohammadi (editorial), and Kim Dow (design).

OAKLAND UNIFIED SCHOOL DISTRICT
Community Schools, Thriving Students

Maker Ed is also grateful for the collaboration with the Oakland Unified School District in the creation and implementation of this curriculum.

MakerEd.Org @makeredorg @makeredorg makereducationinitiative

The Maker Education Initiative is a 501(c)(3) nonprofit organization. EIN#: 83-4594261

OVERVIEW

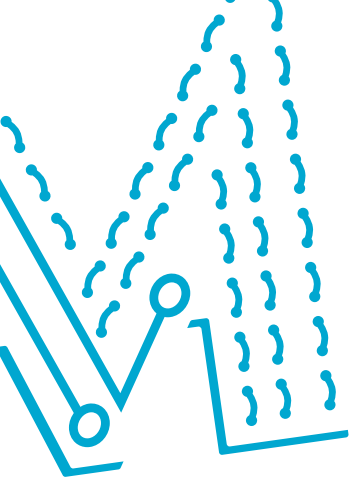
This curriculum is a fun and creative introduction to computer science and hands-on making for makers in middle school grade levels (ages 12–14) with little to no experience in programming or 3D design. In each session, makers have opportunities to develop knowledge in computer science, engineering, art, and game design, as well as exposure to real-world industry concepts and vocabulary.

Using a Micro:bit (a pocket-sized computer) and MakeCode (free online software), young makers practice problem-solving and teamwork to create interactive arcade games. Project-based making and coding is all about learning by making mistakes and working through challenges. This curriculum guide offers facilitation and troubleshooting tips to encourage makers to use available resources to work through challenges with their peers (as opposed to viewing the teacher as the expert).

This 10-week unit provides instruction and activities for 10 hour-long sessions. Each lesson includes an introduction, essential questions, learning outcomes, key vocabulary, a list of materials, teacher prep work, facilitation tips, additional resources, step-by-step lessons, and common troubleshooting tips.

Sessions intentionally synthesize skills learned in previous sessions. However, as each community and learning space is different, lessons are also designed to be modular. Adaptations and pacing can be changed as needed to best suit the needs of you and your learners.

Note: This Cyber Arcade curriculum guide for middle school is an abbreviated and faster-paced version of the Cyber Arcade curriculum guide for elementary school, which is 20 hour-long sessions.



What is a Micro:bit and physical computing?

In this curriculum, we use the **BBC Micro:bit**, a pocket-sized computer that has a variety of features, including 25 LED lights, two programmable buttons, an accelerometer, Bluetooth capability, a built-in compass, and can support external features such as servos, buttons, and speakers. A Micro:bit can be programmed using **Microsoft MakeCode**, a free web-based code editor that uses a block-based programming language.

Physical computing encourages interdisciplinary learning, problem-solving, and creativity through the use of physical materials, technology, and tools. In this 10-week unit, makers design, build, and code an interactive arcade game in creative ways, engaging in computer science, computational thinking, and problem-solving.

What is maker education?

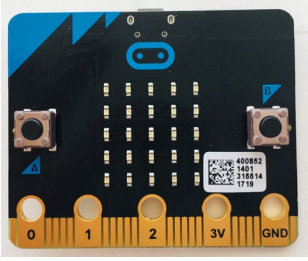
Maker education offers a transformational approach to teaching and learning that attends to the real and relevant needs of humans as learners.

MAKER EDUCATION IS, FIRST AND FOREMOST, HANDS-ON AND LEARNER-DRIVEN.

This empowering approach positions student agency and interest at the center, asking learners to become more aware of the design of the world around them and to see themselves as having the agency to tinker, hack, and improve that design.

Throughout this curriculum, educators are encouraged to match the needs of their learners by varying the pacing of the lessons, encouraging creativity and multiple pathways to completing projects, and allowing learners the freedom to explore, tinker, and make.

WEEK 1 INTRODUCTION TO MICRO:BIT



INTRODUCTION

This week makers learn how to start using a microcontroller (Micro:bit) with coding software (MakeCode) to begin designing cool physical computing projects.

ESSENTIAL QUESTIONS

- What is a microcontroller?
- How can I use one to make cool stuff that connects to the physical world?
- How do artists, engineers, and makers solve problems when they're working/inventing?

LEARNING OUTCOMES

1. Learn about a microcontroller and how to program one with coding software.
2. Write lines of code and learn how to save, download, and upload code from software to the microcontroller.
3. Engage in pair programming as a way of collaborating with a partner.

VOCABULARY

Physical computing: Creating or using devices that interact with the world around us

Microcontroller: Small device (similar to a mini computer) used to control other devices and machines

Pair programming: Method of working in pairs where one person "drives" the laptop while the other "navigates" or instructs the driver

Software: Term used to describe computer programs

Code: Set of instructions that a computer understands

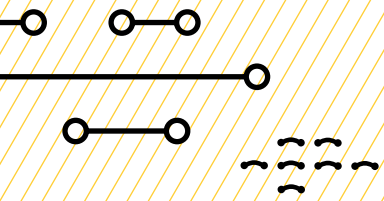
Upload: Sending information or data to a device or computer

Download: Taking information or data from a device or computer

USB to micro-USB cable: Cord that transfers data and information from the computer to a device

USB flash drive: Portable storage device that can store and move files

Troubleshooting: Using resources to solve issues as they arise



MATERIALS LIST

EACH PAIR OF MAKERS NEEDS:

- Micro:bit microcontroller
- Laptop with internet connection
- USB to micro-USB cord
- USB flash drive
- External battery pack
- AAA batteries (2)
- Notebook

TEACHER PREP WORK

1. Ensure the internet connection is working.
2. Connect your laptop to a projector or screen.
3. Preload videos and slideshow.
4. Print the **Troubleshooting Tips** at the end of the lesson and post in the classroom.

FACILITATION TIPS

Make sure that you're comfortable with the save/download/upload process so you can help makers going through this process.

Makers will likely be curious and explore various other blocks on the software page. It's up to you as the facilitator to decide how much you want to let your makers explore. It can be helpful to verbally honor their curiosity but remind them to follow your specific instructions to ensure everyone learns the basics first. Remind them that there will be more time to explore and that this is just the first day.

For makers who finish quickly or need more of a challenge, inform them that there are many tutorials on the **MakeCode website** (such as "Flashing Heart," "Name Tag," or "Smiley Buttons") that they can try on their own to explore the Micro:bit further.

Saving files: This curriculum doesn't require that students create MakeCode accounts and instead uses USB flash drives to save files. Students can create accounts if they have an email address they can login and verify their account with. This enables them to easily save their files, and they can then login and use MakeCode from any other computer with an internet connection.

Collaboration: Let smaller issues work themselves out. Record specific positive examples that you can share with makers in the moment or at the end of the project. These examples provide models for all learners.

Frustration: When frustration levels are not high, let learners figure it out on their own or keep facilitation at a minimum by asking a question and walking away. When frustration levels are high, intervene more directly to help makers find some success.

Circulate among the makers and monitor for both collaboration and frustration.

ADDITIONAL RESOURCES

Micro:bit Project Tutorials

Wonderful Idea Co: Micro:bit Mutants

Micro:bit Tutorial on YouTube

INTRODUCTION TO MICRO:BIT

STEP 1 (5 MINUTES)

Get to know the Micro:bit microcontroller board.

Group makers in pairs, and give each pair a Micro:bit microcontroller board to study.

EXPLAIN

The Micro:bit is one of many types of microcontrollers—small devices, similar to a mini computer, used to control other devices and machines.

Ask makers to study the Micro:bit board closely and:

- Write down and draw their observations in their notebook.

After 3 minutes, ask makers to share:

- What do you notice?
- What do you wonder?

Parts to emphasize that will be used today:

- LED grid
- Micro-USB port
- Battery port

STEP 2 (10 MINUTES)

Get to know the coding software and pair programming.

Assign each pair of makers a laptop and explain that next they'll engage in **pair programming**, which is one way engineers work together. Ask them to figure out who has the next birthday coming up, and that person will start as the "driver."

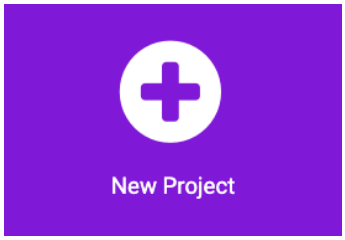
EXPLAIN

The two roles are:

Driver	Operates the laptop, typing, clicking around, etc.
Navigator	Instructs the driver where to go and what to do next

Remind makers to stay in the roles of navigator and driver. The driver shouldn't make any decisions without the navigator.

Instruct the navigators to help the driver to get to the MakeCode website: makecode.microbit.org



Next, makers will:

- Create a **[+new project]** in MakeCode. This will take them to the coding **software**.
- Explore the coding software for 3 minutes (timed).

After 3 minutes, ask makers to switch roles. Set the timer for another 3 minutes.

After makers have each had a chance to explore in the driver and navigator roles, ask them to share:

- What did you discover?
- What do you wonder?

STEP 3 15 MINUTES

Write code for the LED display grid.

Ask makers to switch roles again, and instruct drivers to open a new blank project. Share your screen so that everyone can see how to write their first lines of **code** to program the Micro:bit.

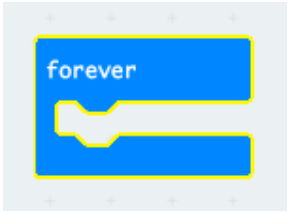
Note: This first project is also available as a step-by-step tutorial called "Flashing Heart" on the MakeCode website.

EXPLAIN

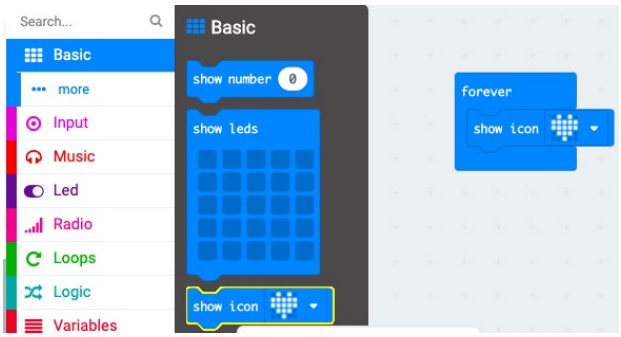
We'll program an animated display for the LED grid using the software and simulator. Navigators, watch closely and instruct your driver to follow these instructions.

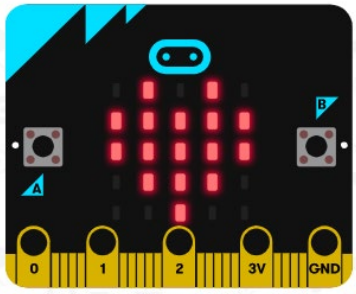
DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

1. Start by using the **forever** block that is in the coding space.

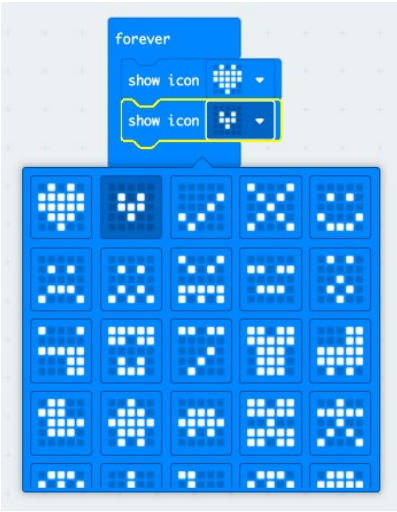


2. Go to the **Basic** menu and drag over a **show icon** block into the **forever** block. You should see a heart display pop up in the simulator on the left side of the screen.



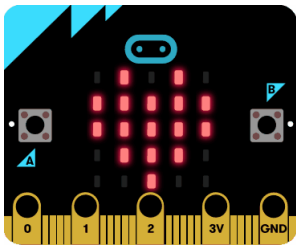


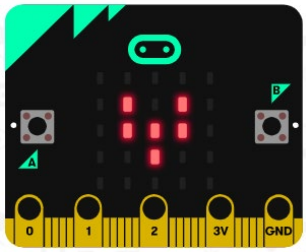
3. Next, click, drag, and drop a second **show icon** block into the code. This time, click the **show icon** dropdown menu and choose the second small heart image to be displayed on the Micro:bit simulator.



4. Repeat Steps 3 and 4 two more times so you have four **show icon** blocks in a row with large and small hearts.

5. Check to see if the simulator shows a flashing heart.





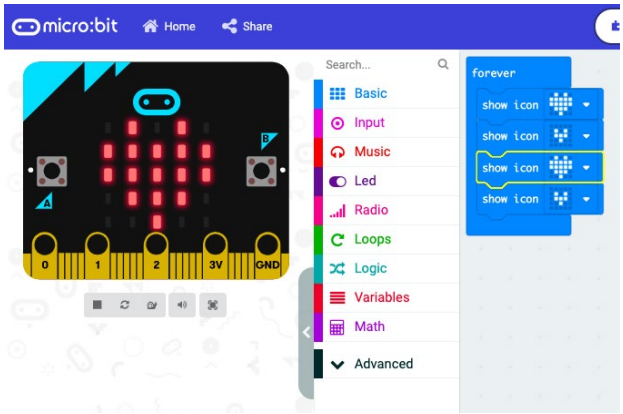
Once makers have successfully coded the flashing heart animation, they will switch roles again to create a new animation using different **show icon** choices from the **Basic** menu. Remind them to stay within the **Basic** menu and **show icon** blocks. Set a timer for 5 minutes.

STEP 4

5 MINUTES

Save the code.

After makers have programmed their animated display using the **show icon** blocks, they should see their animation in the simulator on the left side of the screen. It's now time to save their code to the **USB flash drive**.



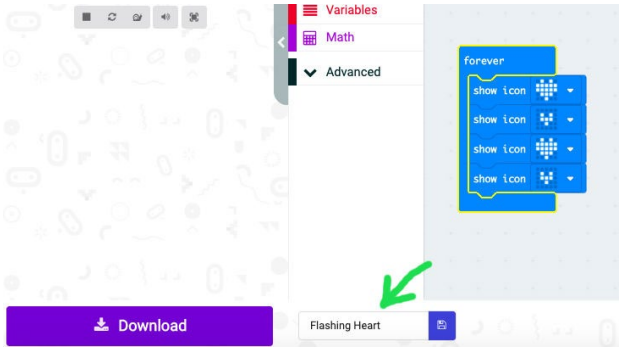
Makers will switch roles again before continuing to the next step.

EXPLAIN

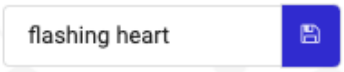
Give your partner a high five for writing your first lines of code together! Next, we'll save our code, then **upload** the code to the Micro:bit so we can see our animation on the Micro:bit. Navigators, watch closely and instruct your driver to do the following steps.

DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

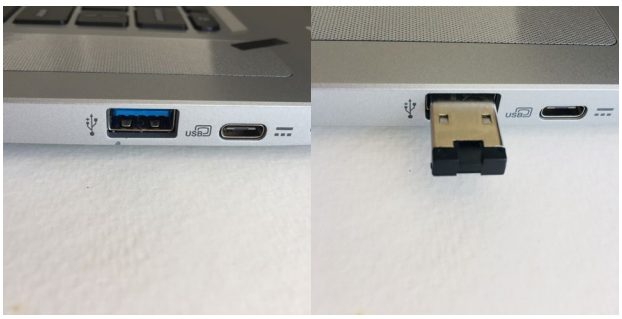
1. Give your project a unique name, like "Flashing Heart".



2. Save your file by clicking on the blue disk icon to the right of the file name.

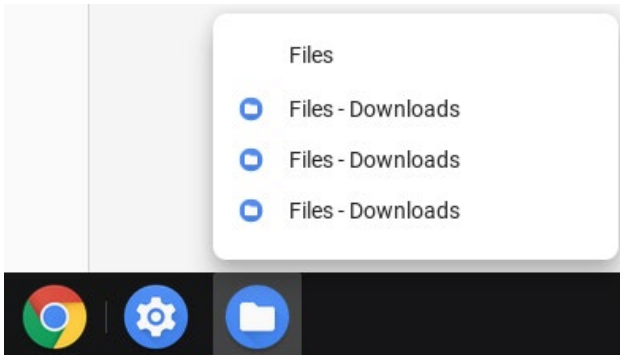


3. Connect your USB flash drive into a USB port on your computer.

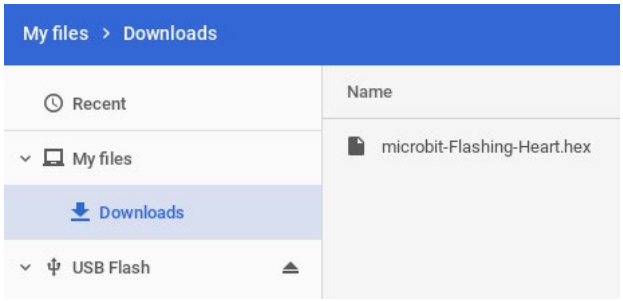


4. Go to the **Files-Downloads** folder on your computer. This is a digital folder that holds information. You can think of it like a backpack full of different folders. You should see both the file

you just named and your USB flash drive listed on the left.



5. Drag the file to the USB flash drive. You need to do this EVERY TIME to save your files so you can keep track and use them again in the future.



In the next step, we'll upload the code to our Micro:bit microcontroller!

STEP 5 10 MINUTES

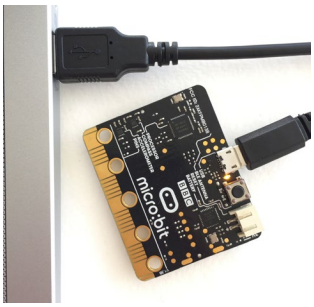
Upload the code onto the Micro:bit microcontroller.

Makers will now connect the Micro:bit board to the laptop in another USB port to upload the code onto the Micro:bit. Remind them to be gentle, look carefully at the shape of the ports, and not force the connection.

DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

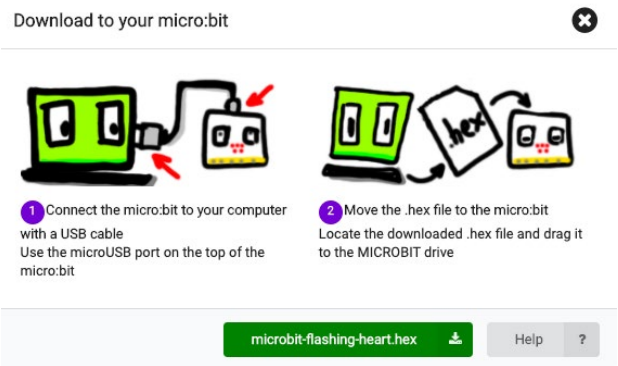
1. First, carefully connect the micro-USB cord to the board.

2. Next, connect the USB to the laptop. You should see the LED grid on the Micro:bit light up. It might have some previously saved code already displayed on it.

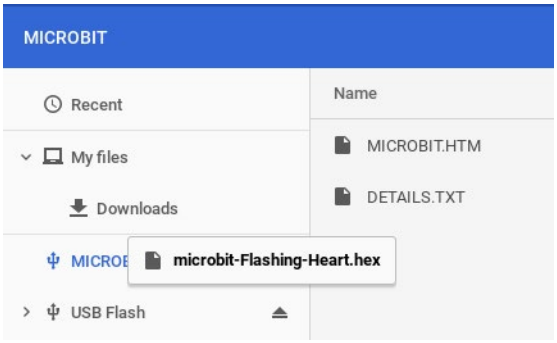


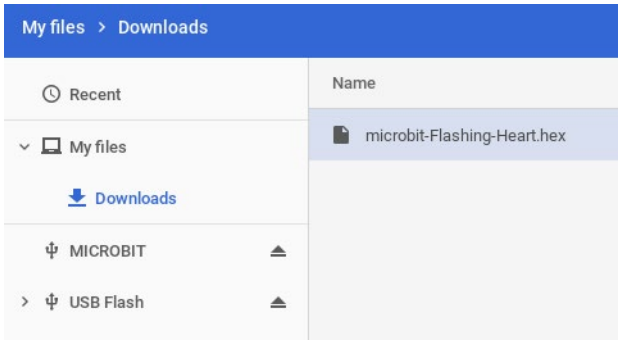
3. Once the board is connected, press the purple **Download** button in the MakeCode software.

You should see this window pop up:



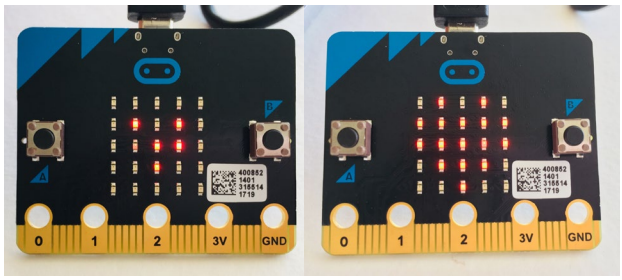
4. Next, go to the **Files-Downloads** folder again in the computer's hard drive and find your named file. Click and drag it onto the Micro:bit icon.





5. You should see a yellow LED on the Micro:bit flashing quickly. This means your code is being uploaded to the Micro:bit. It will stop when the code is finished uploading.

6. Watch to see the display you created load up onto the Micro:bit display grid!



STEP 6

10 MINUTES

Connect the battery pack to the Micro:bit.

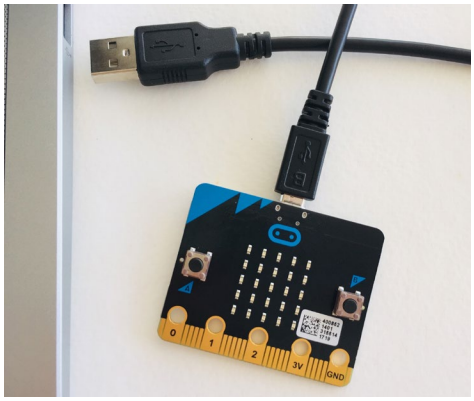
DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

Next, it's time to eject the Micro:bit board from the laptop and connect it to an external battery pack!

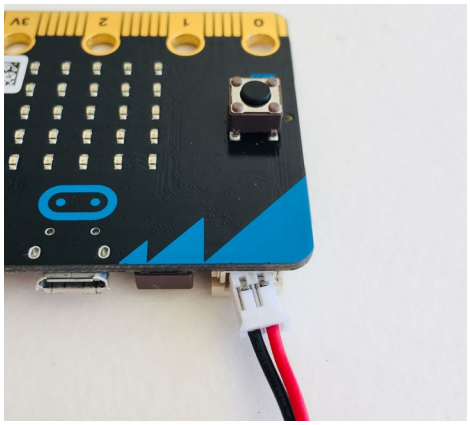
1. First, we need to safely eject the Micro:bit from the computer by

clicking the eject button next to the Micro:bit listed in the finder.

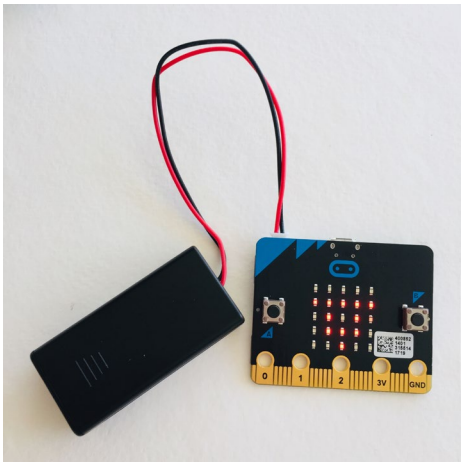
2. Next, safely remove the board from the USB cord. The board will no longer display the code because it isn't being powered.



3. Connect the external battery pack to the Micro:bit



4. You should now see your code displayed on the LED grid!



Once everyone is comfortable with the saving, downloading, and uploading process, they can reconnect the Micro:bit to the computer and continue exploring and coding displays for the Micro:bit.

STEP 7 5 MINUTES

Clean up.

Makers will:

- Disconnect the battery pack.
- Put supplies and technology in assigned bins.
- Return laptops to cart and plug in for charging.
- Clear tables of garbage and recycling.

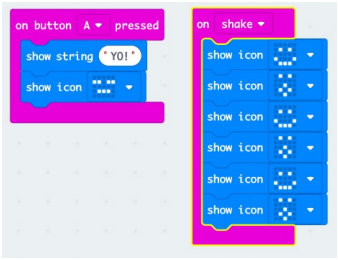
TROUBLESHOOTING TIPS

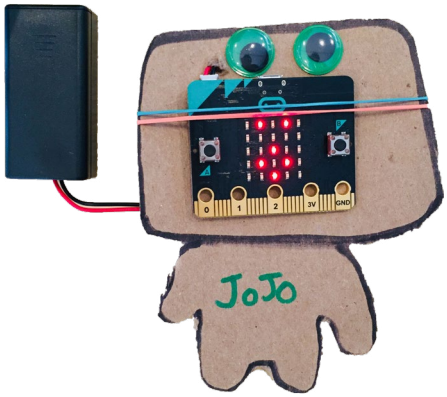
Display on board is not showing what we expected.	**Version check** • Check to see if the latest saved copy of the code has been uploaded. • Re-save the latest version and drag it onto the Micro:bit. **Bugs in code** • Check your code to look for mistakes. • Check to see if there are extra blocks hiding in the coding space.
The LED on the Micro:bit isn't flashing when we press upload.	**Bad cable or USB port** • If the Micro:bit doesn't show up in the computer's menu, try a different cable. • Try a different USB port.
Code isn't uploading correctly to the board, and the board might feel hot.	**Burnt board** • Press the reset button on the board, and then try uploading again. • If the Micro:bit board feels hot, it could be overheated and stop working. Try uploading to a new Micro:bit board.
When we connect the battery pack, the Micro:bit doesn't power up.	**Battery problems** • Check the batteries to see if they're in the right position (+/–). • The batteries could be low or dead. Try replacing them. • Battery pack could overheat or have stopped working. Try a new battery pack.

TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

WEEK 2 CREATE A MICRO:BIT CHARACTER





INTRODUCTION

Now that makers have been introduced to the basics of the Micro:bit microcontroller, it's time to get creative with making a Micro:bit character. Makers work in pairs to co-design a character using maker supplies. Then, they will experiment with writing code to give the character a personality and make it interactive.

ESSENTIAL QUESTIONS

- What are inputs and outputs?
- How can we program them to tell stories or create personality characteristics?
- How do artists, engineers, and makers solve problems when they're working/inventing?

LEARNING OUTCOMES

1. Learn how to code both inputs and outputs in the MakeCode software.
2. Engage in project-based learning through problem-solving and troubleshooting by creating a character with a microcontroller and digital code.

VOCABULARY

Microcontroller: Small device (similar to a mini computer) used to control other devices and machines

Pair programming: Method of working in pairs where one person drives the laptop while the other navigates or instructs the driver

Software: Term used to describe computer programs

Code: Set of instructions that a computer understands

Input: Place where information enters a system

Output: Place where power or information leaves a system

Upload: Sending information or data to a device or computer

USB to micro-USB cable: Cord that transfers data and information from the computer to a device

USB flash drive: Portable storage device that can store and move files

Troubleshooting: Using resources to solve issues as they arise

MATERIALS LIST

EACH PAIR OF MAKERS NEEDS:

- Laptop with internet connection
- USB to micro-USB cord
- USB flash drive
- External battery pack
- AAA batteries (2)
- Notebook

ALL MAKERS NEED ACCESS TO:

- Cardboard
- Glue stick
- Markers
- Colored pencils
- Scissors
- Rubber bands
- Assorted paper
- Assorted maker supplies (pipe cleaners, pom-poms, googly eyes, etc.)

Items can be portioned out per table or set up in an area where students can access them freely.

TEACHER PREP WORK

1. Ensure the internet connection is working.
2. Connect your laptop to a projector or screen.
3. Preload videos and slideshow.
4. Organize maker supplies (pipe cleaners, pom-poms, googly eyes, etc.) into paper trays.
5. Go through the lesson and create your own example of a Micro:bit character to share as a helpful reference for makers troubleshooting their own projects.
6. Print the Troubleshooting Tips at the end of the lesson and post in the classroom.

FACILITATION TIPS

Make sure that you're comfortable with the save/download/upload process so you can help makers going through this process.

Materials management: It's up to you as the educator to decide what works best for your class. You can portion out maker materials into paper trays for each table, or have a dedicated area where makers can access materials freely as needed.

Collaboration: Let smaller issues work themselves out. Record specific positive examples that you can share with makers in the moment or at the end of the project. These examples provide models for all learners.

Frustration: When frustration levels are not high, let learners figure it out on their own or keep facilitation at a minimum by asking a question and walking away. When frustration levels are high, intervene more directly to help makers find some success.

Circulate among the makers and monitor for both collaboration and frustration.

ADDITIONAL RESOURCES

Micro:bit Project Tutorials

Wonderful Idea Co: Micro:bit Mutants

Micro:bit Tutorial on YouTube

CREATE A MICRO:BIT CHARACTER

STEP 1

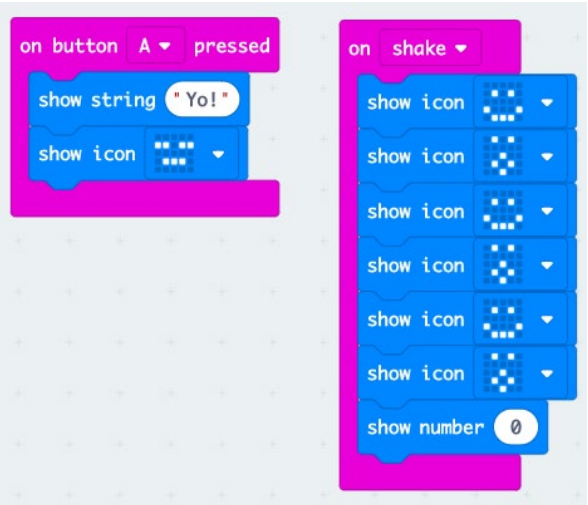
10 MINUTES

Code a character demonstration.

Gather makers around to view the projector as you demonstrate one possibility of coding a character in the **software**. Show them your example character that has the **code** below uploaded to the board.

EXPLAIN

This code could be an example for a character named Jojo. When you meet Jojo, instead of shaking hands or a fist bump, you press **Button A** and Jojo says "Yo!" and the LED grid displays a neutral face.



Jojo is a pretty cool and normal bot, but Jojo is also ticklish. When you **shake** Jojo, it "giggles" and shows a laughing animated facial expression on the LED grid.

Button A and **shake** are both examples of **Input** blocks from the **Input** blocks menu. You can program them to cause various reactions, or outputs. These can be used in creative ways to code a character with personality.

STEP 2

35 MINUTES

Tinker with project-based coding and making.

Next, makers work on designing their own characters. While tinkering and experimenting with code, makers can use the maker supplies (cardboard, rubber bands, markers, googly eyes, etc.) to turn the Micro:bit **microcontroller** itself into a fun physical character with personality.

Note: Advise makers not to apply glue directly to the Micro:bit board.

Their design should include:

- 3 **inputs** that cause at least 3 different **outputs**
- A short story about their character—including name, personality, etc.—and how it relates to their programming

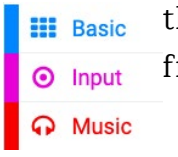
Makers will:

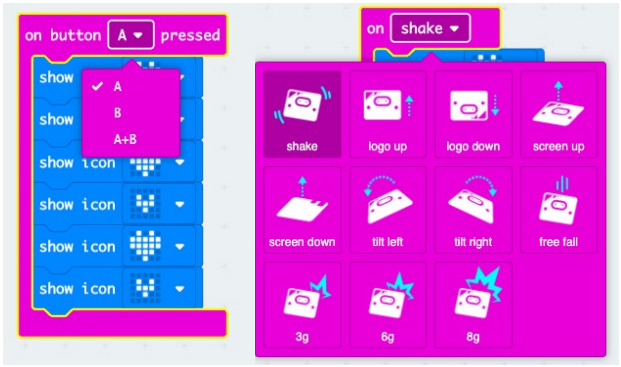
- Get their laptop and bins.
- Choose their **pair programming** navigator and driver roles (time for 15 min each).
- Tinker and explore programming various inputs and outputs in the software and simulator before connecting the Micro:bit to the computer.

Encourage makers to explore and experiment with other blocks within the **Basic** and **Input** blocks.

Remind makers to save changes to the code they want to keep onto their **USB flash drive**.

When they're ready to upload to the Micro:bit, support makers in connecting the **USB to micro-USB cable** to the computer to **upload** their code and connecting the external battery pack to the board to see their code working on the Micro:bit independently from the laptop.



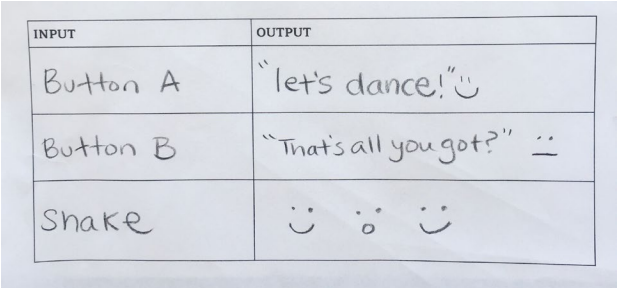


STEP 3 — 5 MINUTES

Document.

Makers will now spend a few minutes writing down their character features in their notebook. Here are examples of a character plan filled out with multiple inputs and outputs.

INPUT	OUTPUT
Button A	"I'm Hungry" + Neutral face
Shake	"Noooo!" + Animated sad face
Button B	"Yum" + Animated smile



STEP 4 — 5 MINUTES

Share and reflect.

Ask for volunteers who would like to present their character to the rest of the group, and have the rest of the characters shared as an interactive gallery walk.

Ask makers to reflect on:

- What challenges did you encounter?
- How did you work through these challenges?

STEP 5

5 MINUTES

Clean up.

Makers will:

- Disconnect the battery pack.
- Put supplies and technology in their assigned bins.
- Return laptops to cart and plug in for charging.
- Return tools and materials that can be used again to the right place.
- Clear tables of garbage and recycling.

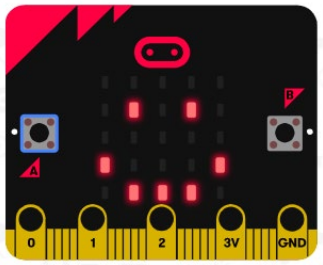
TROUBLESHOOTING TIPS

Display on board is not showing what we expected.	**Version check** • Check to see if the latest saved copy of the code has been uploaded. • Re-save the latest version and drag it onto the Micro:bit. **Bugs in code** • Check your code to look for mistakes. • Check to see if there are extra blocks hiding in the coding space.
The LED on the Micro:bit isn't flashing when we press upload.	**Bad cable or USB port** • If the Micro:bit doesn't show up in the computer's menu, try a different cable. • Try a different USB port.
Code isn't uploading correctly to the board, and the board might feel hot.	**Burnt board** • Press the reset button on the board, and then try uploading again. • If the Micro:bit board feels hot, it could be overheated and stop working. Try uploading to a new Micro:bit board.
When we connect the battery pack, the Micro:bit doesn't power up.	**Battery problems** • Check the batteries to see if they're in the right position (+/–). • The batteries could be low or dead. Try replacing them. • Battery pack could overheat or have stopped working. Try a new battery pack.

TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

WEEK 3 DESIGNING GAMES WITH CONDITIONALS & NUMBER RANGES



INTRODUCTION

This week makers continue to use the Micro:bit microcontroller and begin coding and designing interactive games and art.

ESSENTIAL QUESTIONS

- How can we use code and math to create a fun game?
- What is a conditional statement?
- How do artists, engineers, and makers solve problems when they're working?

LEARNING OUTCOMES

1. Learn how to use code, conditionals, and number ranges to create a game.
2. Engage in project-based learning through problem-solving and troubleshooting by creating a game using a Micro:bit microcontroller and code.

VOCABULARY

Conditional: Set of rules performed if a certain condition is met

Game mechanics: Basic actions, processes, visuals, and control mechanisms that are used to "gamify" an activity

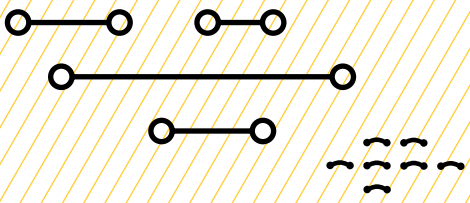
Game designer: Person responsible for designing game storylines, plots, objectives, scenarios, the degree of difficulty, and character development

Game engineer: Person who works with teams of developers on the entire process of creating a video game

Accelerometer: Device used to measure moving forces

Pseudocode: Detailed, informal description of what a computer program must do

Troubleshooting: Using resources to solve issues as they arise



MATERIALS LIST

EACH PAIR OF MAKERS NEEDS:

- Micro:bit microcontroller
- Laptop with internet connection
- USB to micro-USB cord
- USB flash drive
- External battery pack
- AAA batteries (2)
- Notebook

TEACHER PREP WORK

1. Ensure the internet connection is working, and connect your laptop to a projector or screen.
2. Preload videos and slideshow to save time.
3. Prepare the MakeCode file for *if_then_else* in Step 3.
4. Prepare the MakeCode file for digital dice in Step 4, and upload it to a Micro:bit board.
5. Print the **Troubleshooting Tips** at the end of the lesson and post in the classroom.

FACILITATION TIPS

"I Love My Neighbor" is a fun warm-up and icebreaker that uses conditionals. This is a fun and active way to transition makers from the school day to after school. It's also a great game to use as a break when makers need to be physical and move around.

Inspiring creativity: As projects become more unique and individualized, make plenty of room for makers to try new and complex things. Encourage makers to look around the software, try things, and share their learnings with others. They can also use the Troubleshooting Tips, search the internet for help, or use the tutorial page on the **MakeCode website.**

Collaboration: Let smaller issues work themselves out. Record specific positive examples that you can share with makers in the moment or at the end of the project. These examples provide models for all learners.

Frustration: When frustration levels aren't high, let learners figure it out or keep facilitation at a minimum by asking a question and walking away. When frustration levels are high, intervene more directly to help makers find some success.

Circulate among the makers and monitor for both collaboration and frustration.

ADDITIONAL RESOURCES

Micro:bit Game Design with Conditionals

I Love My Neighbor: Icebreaker Game with Conditionals

Examples of Dice Games

How to Play Dice Games

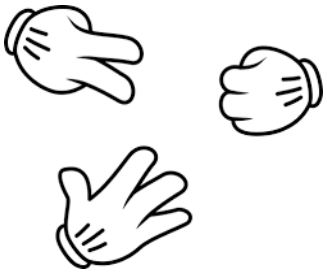
What is an Accelerometer?

How an Accelerometer Works

DESIGNING GAMES WITH CONDITIONALS

STEP 1 5 MINUTES

Learn about game mechanics.



Ask makers to raise their hand if they've ever played Rock Paper Scissors.

Ask a volunteer to explain how to play Rock Paper Scissors.

EXPLAIN

The rules and conditions they just described are called **game mechanics**, the rules and rewards that make up game play and create a fun and engaging experience.

On the board or using the slide deck, review the game mechanics for Rock Paper Scissors:

- There are two players.
- Each player chooses to throw either Rock, Paper, or Scissors.
- If Rock and Paper, Paper wins.
- If Rock and Scissors, Rock wins.
- If Paper and Scissors, Scissors wins.
- Players can choose to do two out of three rounds to win.

Note: It's possible to program the game Rock Paper Scissors in Micro:bit as well, but the computer needs very specific information. If makers are interested, for the future, there's a **Rock Paper Scissors Micro:bit tutorial.**

STEP 2 5 MINUTES

Learn about conditional statements.

EXPLAIN

Game mechanics can also be programmed into digital and video games using code. To do this, **game designers** and **game engineers** use what they call **conditionals**. An example of a common conditional statement is: **if_then_else.**

Ask if anyone has heard of a Tamagotchi or Giga Pet. Show **this video** on a projector or large screen, if available.

"Tamagotchi is back" on YouTube, uploaded by CNN Business, 10/10/2018

EXPLAIN

This popular interactive toy/game is also based on conditionals. When the digital pet is "hungry" or wants to "play," then the user would have to press buttons to "feed" it or "play a game" with it. If they didn't do those things, the pet would die. The digital display of this popular game is similar to the Micro:bit. Next, you'll work together to program your own digital Micro:bit pet.

STEP 3 15 MINUTES

Learn about conditional statements in MakeCode.

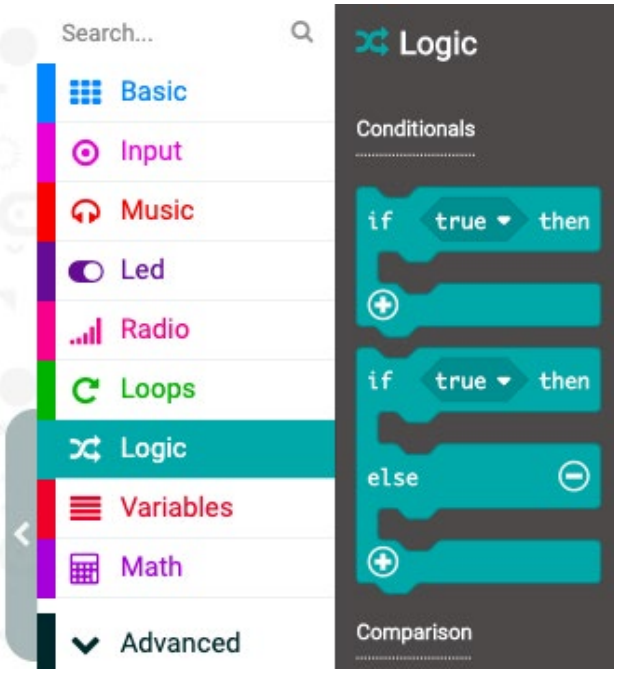
Show an example of a conditional statement in MakeCode with the Micro:bit simulator and talk through the logic.

EXPLAIN

We've already used **Basic** and **Input** blocks. Now we'll explore using **Logic** blocks. In the **Logic** menu, there's an **if_then** block we can use to code rules or game mechanics, similar to when we played Rock Paper Scissors. You can use these blocks to code different game mechanics.

DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

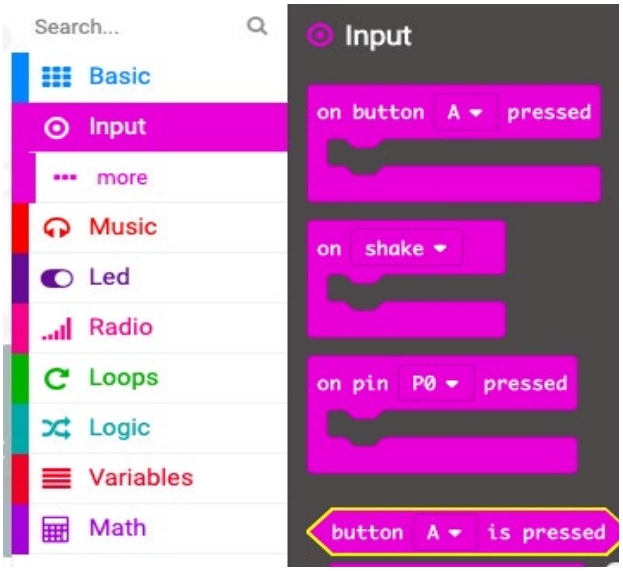
1. Click on the **Logic** menu of blocks. Click and drag over an **if_true_then** block to the coding space.



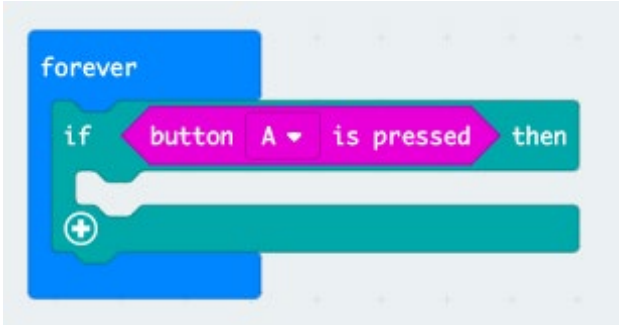
2. Notice the rectangular shape with the pointed ends that reads **true** inside of it. In the MakeCode software, you can drop any other block that

matches this shape into blocks with the same shape.

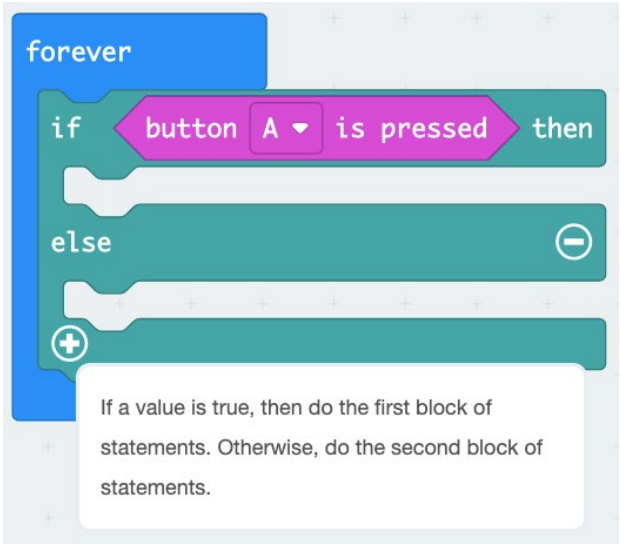
For example, click on the **Input** menu. Notice there's a block that matches the rectangle with the pointed ends (shown outlined in yellow).



3. Drag the **button A is pressed** block and drop it into the **if_then** statement so the code would read, **if button A is pressed then**.



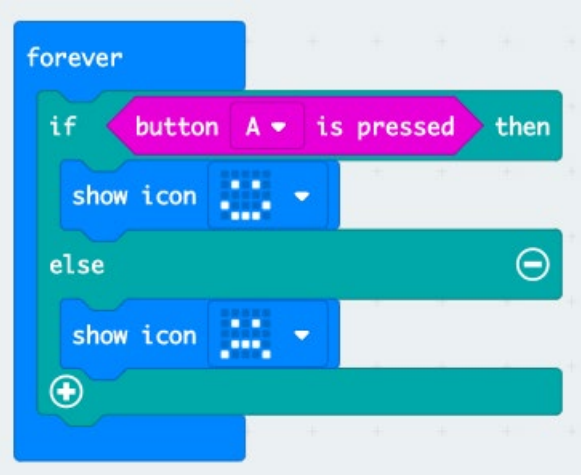
4. Next, click on the **+** in the **if_then** block. You'll see the green block grow with a space for an **else** statement.

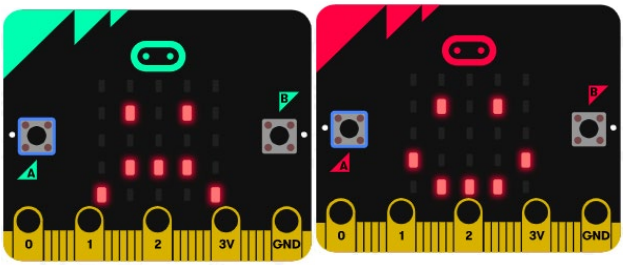


5. The example code shown could be the start of programming a digital pet game, similar to the Tamagotchi. For example, **button A** could be "feeding" the character, which then shows a smile on the LED display.

Demonstrate on the simulator, pressing **button A** to show the smile and frown.

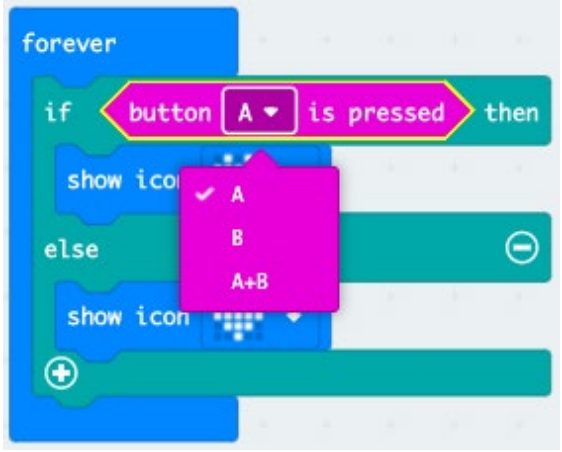
IF button A is pressed
THEN show smile
ELSE show frown

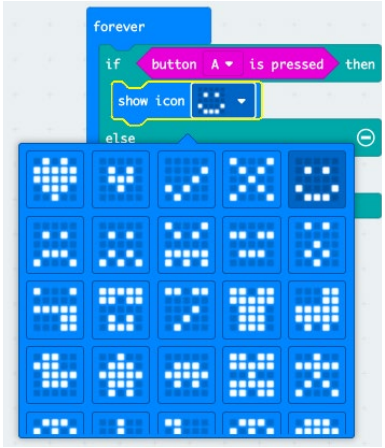




6. Remember, in order to save the code, we need to name and save the file, then connect the USB flash drive to move the file onto the USB flash drive.

7. Next, you and your partner can tinker with this code by changing the inputs and displays.





STEP 4 10 MINUTES

Program digital dice.

Now that we've programmed **conditionals**, next we'll use the **Math**

blocks to explore using number ranges for designing games. We'll start by programming "digital dice."

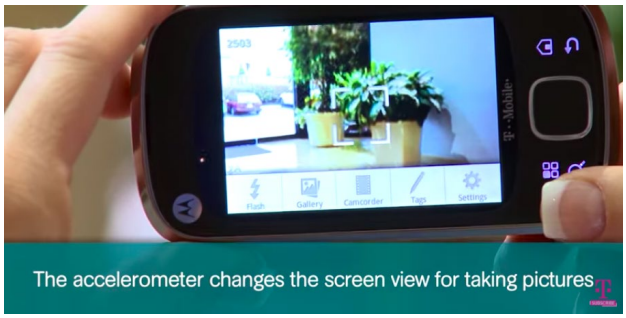
- Ask makers: "What are some games you've played that use dice?"

EXPLAIN

We can program the Micro:bit to act like a dice. There are different types of dice, both six-sided and ones that have more than six sides. We'll start with coding a regular six-sided dice and program the Micro:bit to select a random number in the range 1–6 when we "shake" the board.

- Ask makers: "How do you think the Micro:bit recognizes when it's being shaken?"

The Micro:bit has an **accelerometer** that is able to sense when it's being moved in a direction. Most smartphones also contain an accelerometer, which is how they know to change the orientation of the screen when the phone is tilted. (This optional short **video** explains the role of accelerometers in phones.)



"**What is an Accelerometer?**" on YouTube, uploaded by T-Mobile, 7/29/2010

STEP 5 (20 MINUTES)

Makers become game designers and game engineers.

EXPLAIN

Now that you know how to program conditionals and **Math** blocks, you can spend the rest of the session as **game designers** and **game engineers**. You will work together to design a simple game using either conditionals or by programming number ranges and deciding what the **game mechanics** are for your game. Your game mechanics will be documented in your notebooks. One example is "Micro:bit Truth or Dare".

DEMONSTRATE GAME EXAMPLE: MICRO:BIT TRUTH OR DARE

In the code below, **button A** (truth) and **button B** (dare) are programmed as inputs to pick a random number between 1–6.

Assigning the rules or game mechanics might look like:

Micro:bit Truth or Dare

	Truth	Dare
1	What superpower would you like to have and why?	Try to lick your elbow.
2	What is the strangest dream you've ever had?	Balance a spoon on your nose for 10 seconds.
3	If you were on a deserted island, which 2 people would you bring with you and why?	Sing your favorite song in a funny voice.
4	Have you ever fallen asleep in class?	Draw a face on your hand, go up to someone you don't know and make it say, "Have a nice day."
5	Would you like to go to school in a banana costume for $25?	Play the staring game with someone until one of you laughs.
6	What is your least favorite food?	Go sing "happy birthday" to someone you don't know.

Other examples of game mechanics using number ranges or conditionals are:

- Moving pieces on a board game — the Micro:bit tells you how far to move.
- Participating in a scavenger hunt — the Micro:bit chooses the task.
- Playing a card game — the Micro:bit tells you how many cards to take.
- Playing hot potato — the Micro:bit reacts to being passed and shows when the game is over.
- Any game using the Micro:bit as a die roll or a coin flip.

Note: Makers can choose to work with another group to use two Micro:bits for their game design.

STEP 6

Clean up.

Makers will:

- Disconnect the battery pack.
- Put supplies and technology in their assigned bins.
- Return laptops to cart and plug in for charging.
- Clear tables of garbage and recycling.

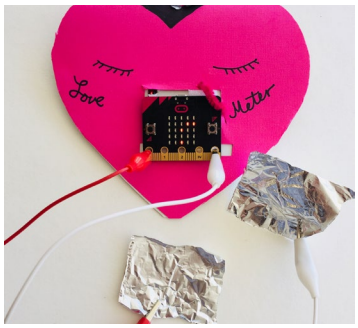
TROUBLESHOOTING TIPS

The board isn't showing what we coded.	**File version check** • Check to see that you've uploaded the most recent copy of the code. • Resave the latest version and drag and drop onto the Micro:bit.
The code isn't doing what we expected.	**Check for bug** • Read through the code. • Read it out to a friend. • Check to see if there are extra blocks that aren't supposed to be there.
The LED on the Micro:bit isn't flashing when we click Upload.	**Bad cable or port** • If the Micro:bit isn't showing up in the computer menu, try a different cable. • Try a different USB port on the laptop.
Our code isn't uploading correctly to the board. The board feels hotter than usual.	**Burnt board** • Try pressing the reset button on the board. • Try uploading to a new Micro:bit board.
The board isn't turning on when connected to the battery pack.	**Battery** • Check the batteries to see if they're charged. • Check to see if the batteries are flipped.

TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

WEEK 4 UI/UX OF MICRO:BIT GAMES



INTRODUCTION

In this session, makers learn how to design and code new ways of interacting with the Micro:bit microcontroller.

ESSENTIAL QUESTIONS

- How can we design a fun, interactive game experience?
- How do artists, engineers, and makers solve problems when they're working?

LEARNING OUTCOMES

1. Learn how to code the external pins on the Micro:bit and work with conductive materials.
2. Explore how design and user interface influence how successful or fun a game is.
3. Engage in project-based learning through problem-solving and troubleshooting by creating a game with a Micro:bit and code.

VOCABULARY

User: People playing/interacting with the game

User interface (UI): Physical and digital "interface" of a game or device

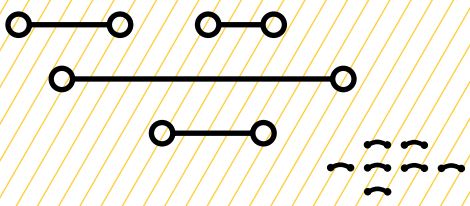
User experience (UX): How natural and enjoyable a game or device is to use

Switch: Device for making and breaking the connection in an electric circuit

Buttons: Device for making and breaking the connection in a circuit, similar to a switch but can be either momentary or can lock into place

Remix: To reuse but make changes to

Conductive materials: Materials that allow electricity to flow through them



MATERIALS LIST

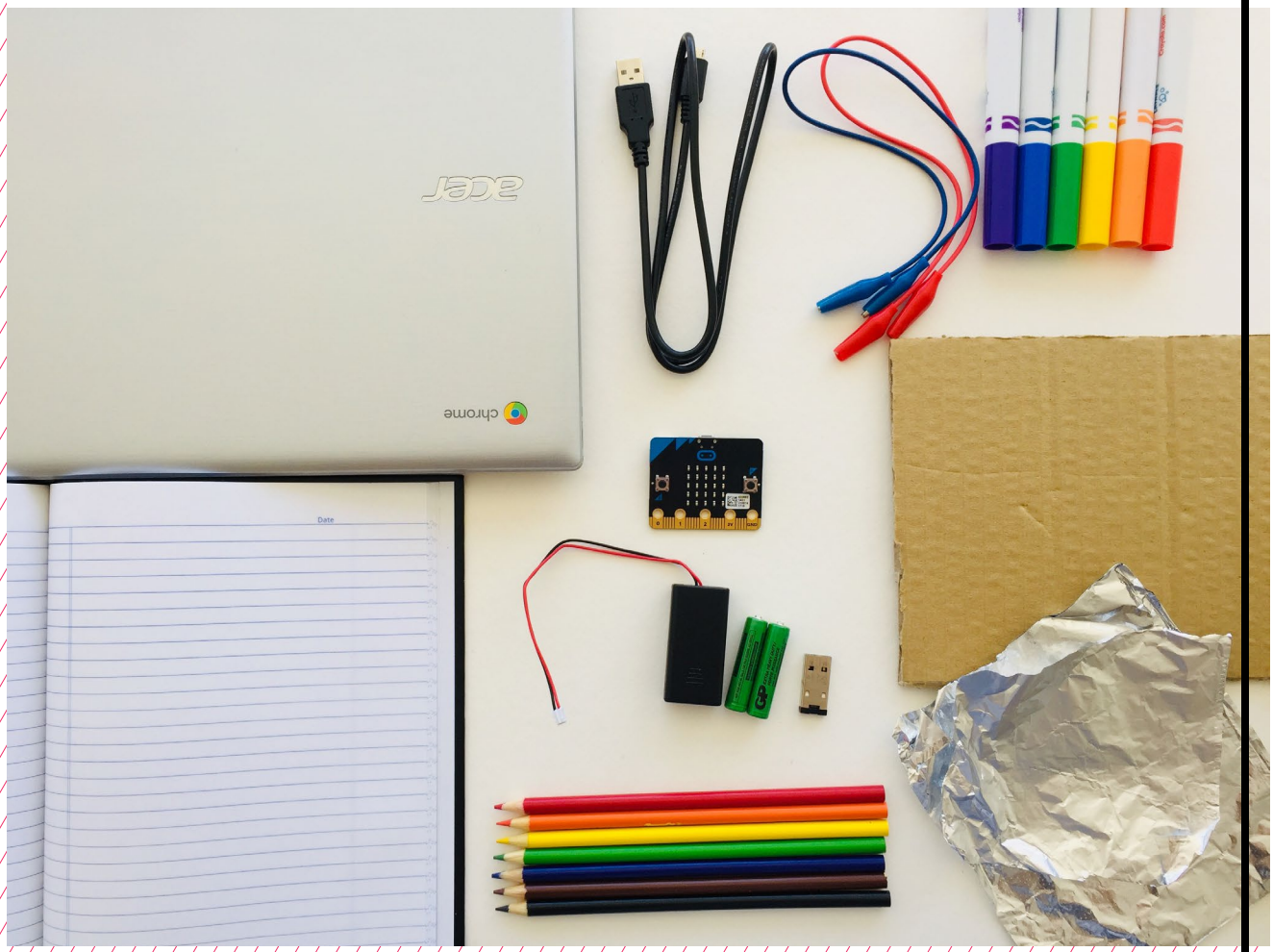
EACH PAIR OF MAKERS NEEDS:

- Micro:bit microcontroller
- Laptop with internet connection
- USB to micro-USB cord
- USB flash drive
- Alligator clips (2)
- Notebook

ALL MAKERS NEED ACCESS TO:

- Cardboard
- Aluminum foil
- Markers
- Colored pencils

Items can be portioned out per table or set up in an area where students can access them freely.



TEACHER PREP WORK

1. Ensure the internet connection is working and connect your laptop to a projector or screen.
2. Preload videos and slideshow to save time.
3. Code the Love Meter game and upload it to a Micro:bit as an example.
4. Arrange maker materials: aluminum foil, alligator clips, cardboard, and markers.
5. Print the **Troubleshooting Tips** at the end of the lesson and post in the classroom.

FACILITATION TIPS

Electronics: Tinkering with electrical connections can be a bit tricky and may not work when things aren't securely connected. If students get frustrated, encourage them to check their connections, check the troubleshooting guide, ask a classmate for help, and take breaks.

Inspiring creativity: Take note and celebrate surprising discoveries. Give makers opportunities to share the cool things they figure out with the rest of the class.

Collaboration: Let smaller issues work themselves out. Record specific positive examples that you can share with makers in the moment or at the end of the project. These examples provide models for all learners.

Frustration: When frustration levels aren't high, let learners figure it out or keep facilitation low touch by asking a question and walking away. When frustration levels are high, intervene more directly to help makers find some success.

Circulate among the makers and monitor for both collaboration and frustration.

ADDITIONAL RESOURCES

How Do Game Controllers Work?

BBC Micro:bit Love Meter

About Buttons

UI/UX OF MICRO:BIT GAMES

STEP 1 10 MINUTES

Introduce user interface/ user experience (UI/UX).

Ask makers:

- How many of you have used a game controller to play a video game?
- What are some of the game controllers/games you've played?

EXPLAIN

Did you know that designing a game experience (both the game itself and the controller) is a job? **User interface (UI)/user experience (UX)** designers and engineers are people who get paid to design and code ways in which **users** interact with games.

One of the coolest features of the Micro:bit is that you can connect alligator clips or other **conductive materials** to change the UI and UX.

In this next activity, we'll experiment with various ways to change the UI/UX of some simple games.

Either on a projector or shared slideshow, show this **video**, which explains how game controllers use circuits to control how the player interacts with the game software.

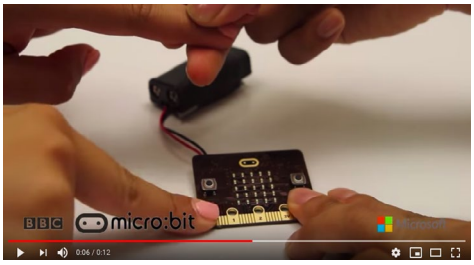
"**How Do Game Controllers Work?**" on YouTube, uploaded by Fun Kids Learn, 1/3/2017

After watching, ask makers to answer the following in their journals and then share out loud:

- What do all **buttons** have in common? *(A: They're* **switches***.)*
- How does the computer know what to do when a button is pressed? *(A: The code is activated by the button.)*

STEP 2 5 MINUTES

Introduce how to program the Love Meter game.



"**BBC: Micro:bit Love Meter"** on YouTube, uploaded by Touch Develop, 11/11/2015

EXPLAIN

Now that we know game controllers use switches, we'll code a simple game called Love Meter and add a switch to it.

Note: There's a guided tutorial for the Love Meter game on the MakeCode website.

Ask for two volunteers to play the game that you prepared as an example, or show this video if unable to prepare the example:

- Instruct one person to touch Pin 0 (zero) and the other to touch GND.
- Next, instruct them to touch hands.
- The LED display should read "Love Meter" and display a number.

EXPLAIN

This game works because the contacts on the Micro:bit and our skin are both conductive. When one person touches the metallic area under Pin 0 and another person touches GND, they become part of the circuit. When they touch each other's skin, this acts like a switch—just like the switch mentioned in the video. Now it's your turn to code the Love Meter game.

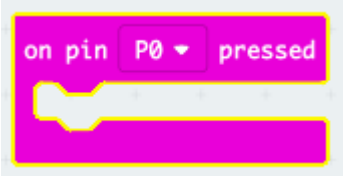
STEP 3 20 MINUTES

Program and remix the Love Meter game.

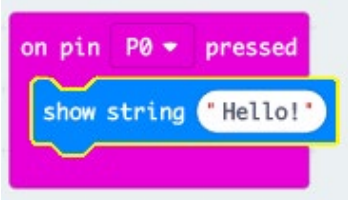
Next, makers will code and **remix** the Love Meter game to create their own version of the game.

DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

1. Drag over an **on pin P0 pressed** block from the **Input** menu.

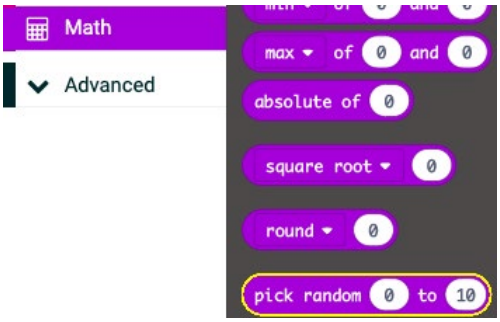


2. Drag over a **show string** block from the **Basic** menu. Type in "Love Meter" where it says "Hello".



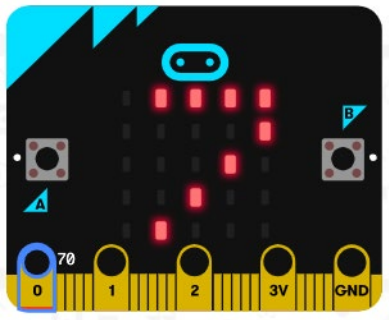
3. Drag over a **show number** block from the **Basic** menu.

4. Drag over a **pick random** block from the **Math** menu and put it into the oval in the **show number** block.



5. Change the values to be **pick random 0 to 100**.

6. Test the code in the simulator by clicking on **Pin 0.** The words "Love Meter" should scroll across the screen followed by a number between 0–100.



7. Once the code works, makers will save and upload the code to their Micro:bit and connect to their battery pack.

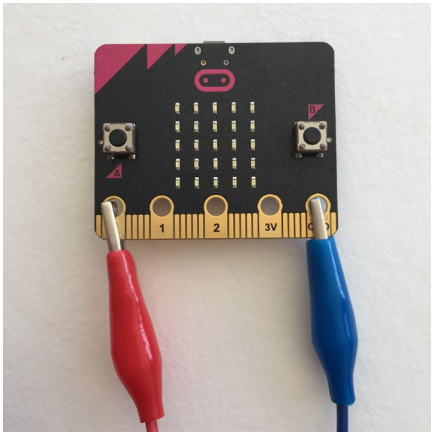
Note: If makers have extra time, they can edit the code to display different text and animations. Everyone should include **Pin 0**, **display**, **show number**, and **pick random**.

STEP 4 (15 MINUTES)

Add alligator clips and foil.

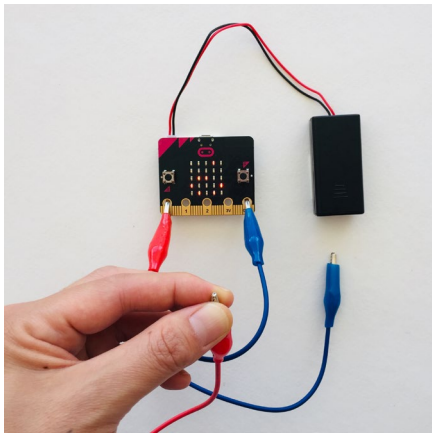
Using the same code from the Love Meter in Step 3, instruct makers to:

1. Connect alligator clips to **Pin 0** and **GND**.



2. Try playing the Love Meter again, but this time makers each hold a metal end of one alligator clip instead of directly touching the board. If working correctly, makers will be able to activate the Love Meter game code by touching one end of the metal alligator clip and then making contact

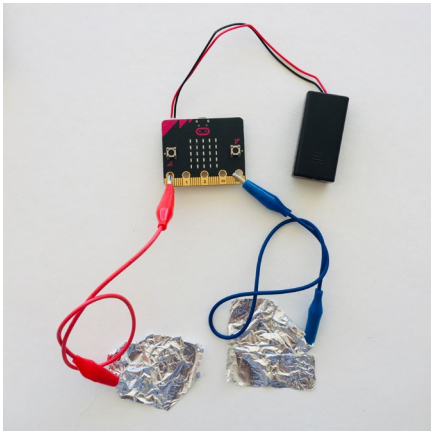
with the other person holding the other alligator clip.

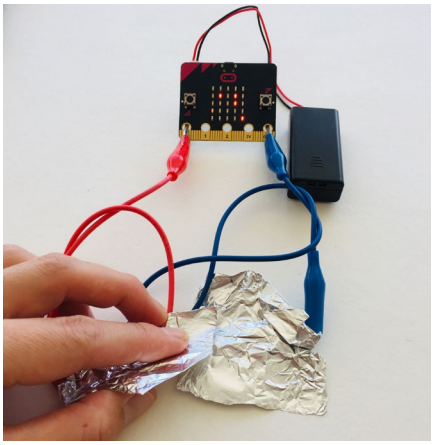


EXPLAIN

Aluminum foil is conductive. By touching two pieces of foil together, it creates a switch. If we attach foil to the alligator clips, we can touch the pieces of foil together to activate the Love Meter.

3. Demonstrate using aluminum foil connected to the alligator clips. Let makers experiment with UI/UX design by exploring different ways of positioning and interacting with the foil (two pieces, on hands, etc.).





STEP 5

5 MINUTES

Share and reflect.

Ask if any groups want to volunteer to demonstrate their UI/UX.

Ask them to share:

- What changes did you make to the original code?
- How does your UI/UX make the interaction more fun?

STEP 6

Clean up.

Makers will:

- Disconnect the battery pack.
- Put any materials they want to keep to use in their bin.
- Put away technology and make sure laptops are charging.
- Return tools and materials that can be used again to the right place.
- Clear tables of garbage and recycling.

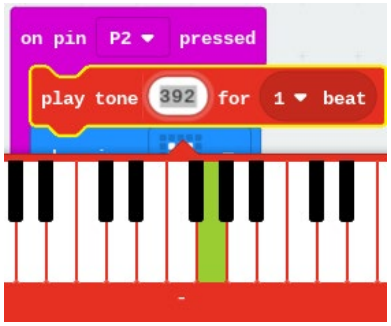
TROUBLESHOOTING TIPS

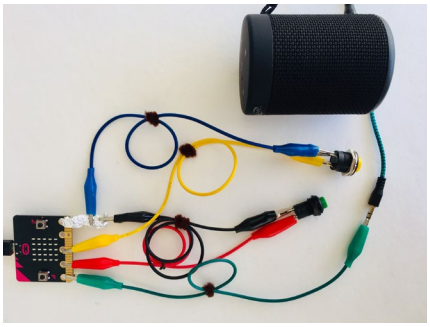
The board isn't showing what we coded.	**File version check** • Check to see that you've uploaded the most recent copy of the code. • Resave the latest version and drag and drop onto the Micro:bit.
The code isn't doing what we expected.	**Check for bugs** • Read through the code. • Read it out to a friend. • Check to see if there are extra blocks that aren't supposed to be there.
The LED on the Micro:bit isn't flashing when we click Upload.	**Bad cable or port** • If the Micro:bit isn't showing up in the computer menu, try a different cable. • Try a different USB port on the laptop.
Our code isn't uploading correctly to the board. The board feels hotter than usual.	**Burnt board** • Try pressing the reset button on the board. • Try uploading to a new Micro:bit board.
The board isn't turning on when connected to the battery pack.	**Battery** • Check the batteries to see if they're charged. • Check to see if the batteries are flipped.
We have alligator clips connected to the board, but the code isn't running.	**Alligator clips** • Make sure alligator clips are secure on the correct pins and are touching the metallic parts. • Try switching alligator clips.

TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

WEEK 5 CODING SOUND & SWITCHES





INTRODUCTION

In this session, makers continue to explore user interface and user experience (UI/UX) by connecting momentary and locking buttons to the Micro:bit microcontroller.

ESSENTIAL QUESTIONS

- How can we connect buttons to the Micro:bit and code music for a fun, interactive experience?
- How do artists, engineers, and makers solve problems when they're working?

LEARNING OUTCOMES

1. Learn how to connect buttons to the Micro:bit.
2. Learn how to code music blocks.
3. Engage in project-based learning through problem-solving and troubleshooting by creating a game with a Micro:bit and code.

VOCABULARY

User: People playing/interacting with the game

User interface (UI): Physical and digital method or "interface" of how the user interacts with the game

User experience (UX): How natural, intuitive, and enjoyable those interactions are

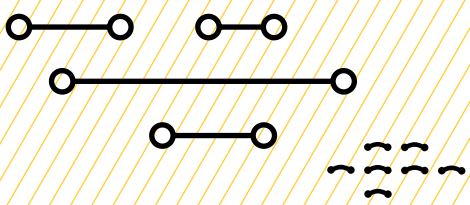
Button (or switch): Devices for making and breaking the connection in an electric circuit

Momentary button: Switch that doesn't make contact unless it's held down

Locking button: Switch that latches in a set position

Speaker: Device that receives and amplifies an audio signal so it can be heard

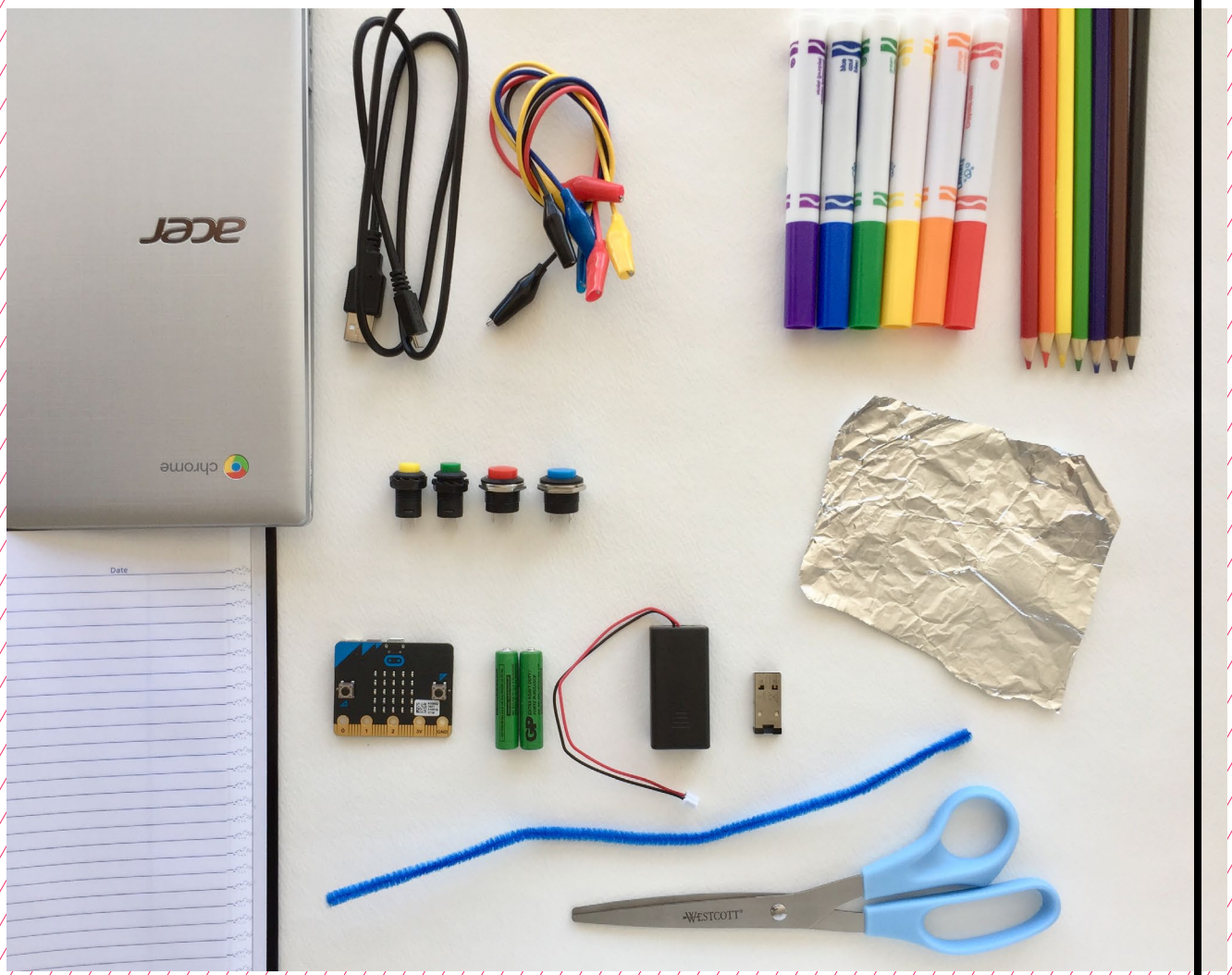
Troubleshooting: Using resources to solve issues as they arise



MATERIALS LIST

EACH PAIR OF MAKERS NEEDS:

- Micro:bit microcontroller
- Laptop with internet connection
- USB to micro-USB cord
- USB flash drive
- Battery pack
- AAA batteries (2)
- Alligator clips (4)
- Momentary buttons (2)
- Locking buttons (2)
- Notebook
- Markers or colored pencils
- Aluminum foil
- Pipe cleaner
- Scissors



TEACHER PREP WORK

1. Ensure the internet connection is working, and connect your laptop to a projector or screen.
2. Preload videos and slideshow to save time.
3. Code the button keyboard in Step 2 and upload it to a Micro:bit as an example.
4. Set aside for each group: 2 momentary and 2 locking buttons, plus 4 alligator clips.
5. Print the **Troubleshooting Tips** at the end of the lesson and post in the classroom.

FACILITATION TIPS

Electronics: Tinkering with electrical connections can be tricky, especially when connecting multiple alligator clips to one terminal on the Micro:bit. It can be helpful to extend the metallic terminal from Pin or GND with a small piece of aluminum foil or other conductive material (see the images in Step 3).

Inspiring creativity: Take note and celebrate surprising discoveries. Give makers opportunities to share the cool things they figure out with the rest of the class.

Collaboration: Let smaller issues work themselves out. Record specific positive examples that you can share with makers in the moment or at the end of the project. These examples provide models for all learners.

Frustration: When frustration levels aren't high, let learners figure it out or keep facilitation low touch by asking a question and walking away. When frustration levels are high, intervene more directly to help makers find some success.

Circulate among the makers and monitor for both collaboration and frustration.

ADDITIONAL RESOURCES

Connecting Audio to the Micro:bit

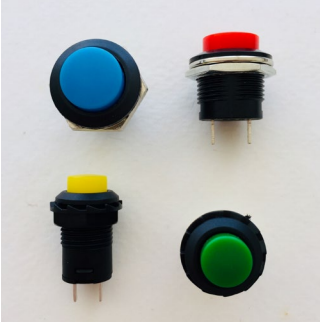
Creating a Controller for a Micro:bit Game

CODING SOUND & SWITCHES

STEP 1

5 MINUTES

Introduce buttons: momentary vs. locking.



EXPLAIN

Today, we'll continue programming the Micro:bit and connecting switches to play with the UI/UX. Let's check out some of these cool arcade switches and buttons. They might look familiar, similar to something you see on your game controller at home or at an arcade.

Next:

- Give each group 2 **locking buttons** and 2 **momentary buttons**.
- Let them play around with clicking them and see if they feel a difference.
- Ask them to draw one in their notebook, noticing what parts the buttons have and what they might do.

They'll probably notice that one of the buttons clicks but doesn't stay down, while the other clicks and stays locked. They have to press it again to unlock it.

EXPLAIN

The button that clicks but doesn't stay down is a **momentary button,** and it only completes the circuit when the button is held down. The button that clicks and stays locked is a **locking button.** When you press this button, the circuit stays connected until pressed again. These buttons will be available as you continue to design UI/UX for your own games.

STEP 2

15 MINUTES

Code pins and music!

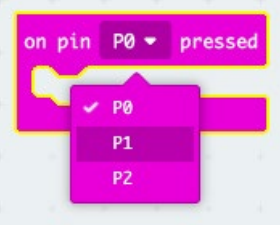
EXPLAIN

Now we'll continue exploring the Micro:bit, and learn how to program 2 pins to play different sounds. Then we'll connect 2 switches and a **speaker**! Choose your driver and navigator roles and let's begin. Navigators, watch closely and instruct your driver to follow these steps.

DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

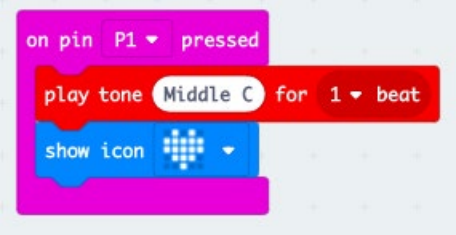
1. Start by opening a new project. Then, click and drag an **on pin pressed** block from the **Inputs** menu into the coding space.

2. Click on the **P0 dropdown menu** and change it to **Pin 1**.

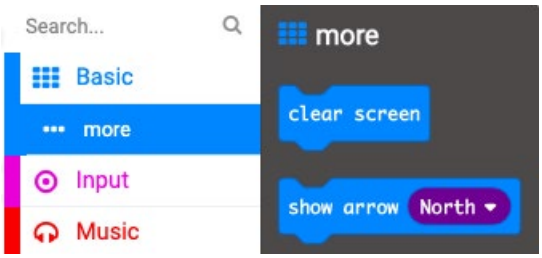


3. Next, go to the **Music** menu, drag over a red **play tone** block, and nest it into the first block.

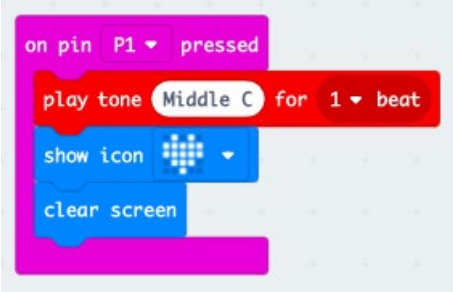
4. Drag over a **show icon** block from the **Basic** menu and add it to the sequence.



5. Then, go to the **Basic** menu and click on **more** to find the **clear screen** block.



6. Drag over a **clear screen** block to add into the sequence.

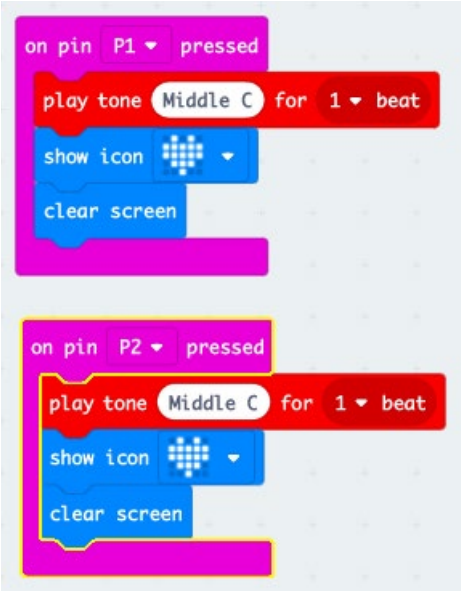


7. Next we want to duplicate this entire sequence of blocks. We can do this a few ways:

   - Repeat Steps 1–6.
   - Go to the top **on pin pressed** block and use two fingers to press down on the trackpad.
   - Hold down the alt key and click on the block to show a dropdown menu. Then choose the Duplicate option.

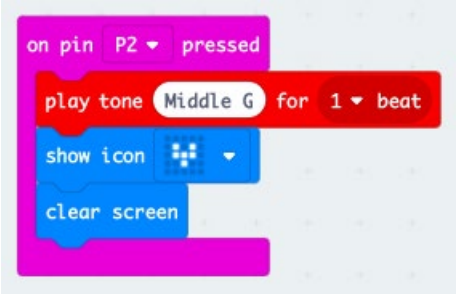
8. You should see a copy of these blocks, but one is greyed out because it's inactive.

9. Change the first block to be **on pin P2 pressed**, and the blocks should be no longer grayed out.



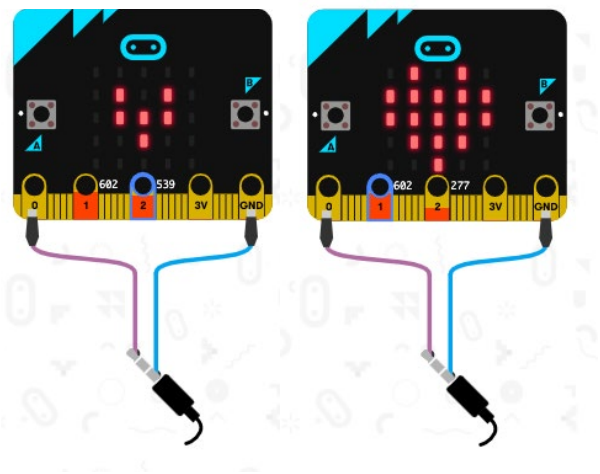
10. Click on the text "Middle C" to change the note**.** You'll see a keyboard pop up. You can choose any second note to play.

11. Next, change the **show icon** block display to be a small heart.

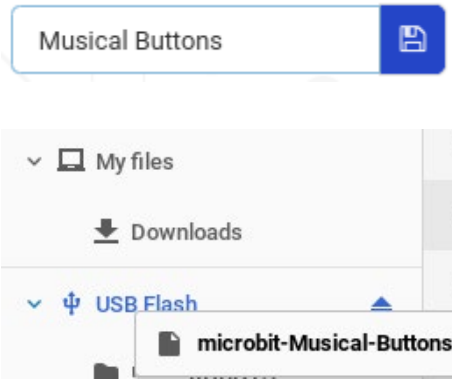


You should now have the two sequences of code shown here.

12. Next, in the simulator, test the code by clicking on **Pin 1** and **Pin 2.** You should hear the tones playing through the computer speaker!



13. Notice that the simulator shows **Pin 0** and **GND** connected by wires to the end of a plug. This is showing us how to connect to a speaker. We'll do that later, after we upload the code to our board.
14. Once we've coded the two sequences:
    - Name the file with a unique name.
    - Save the file to the USB flash drive.
    - Connect the Micro:bit with the USB cord.
    - Click and drag the file into the Micro:bit to upload.



STEP 3 20 MINUTES

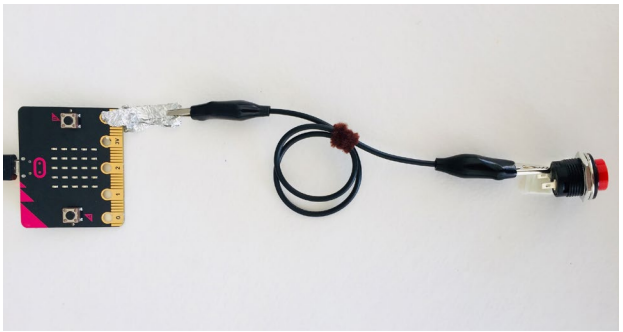
Connect the pins.

Note: Makers should leave the board connected to the laptop while doing this step to reduce troubleshooting issues from the battery pack.

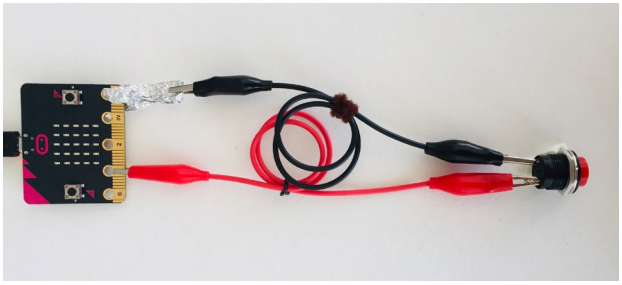
It's time to connect 2 buttons to the board. Give each group 4 alligator clips. Have makers swap driver and navigator roles from the last step. (Drivers are the ones with their hands on the board with wires. Navigators instruct and support their partner to follow the steps below.)

DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

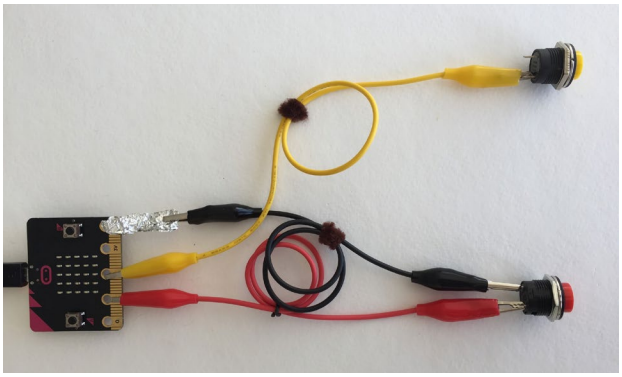
1. Connect an alligator clip to **GND** and the other side to one terminal of the **momentary button**. It can be helpful to twist some aluminum foil and loop it through **GND** because we'll be adding more alligator clips to **GND**.

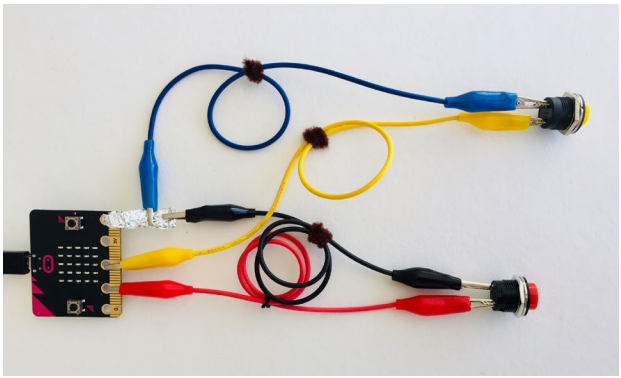


2. Next connect an alligator clip to **Pin 1** and the other end of the alligator clip to the other side of the momentary button terminal.



3. Repeat Steps 1 and 2 but connect to **Pin 2** with a second button. You can connect to the aluminum foil on **GND** or connect to the metal alligator clip that's already on **GND** because the metal clip is conductive.



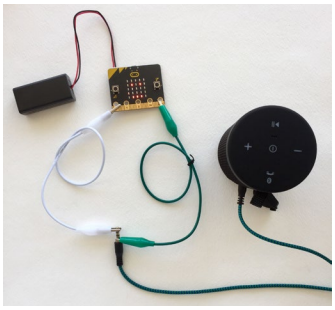


4. When you press the button connected to Pin 1 and Pin 2, you'll see the animation of the heart but you won't be able to hear the music. This is because the Micro:bit isn't connected to a speaker. We'll connect a speaker during the next session.

Note: In Steps 3 and 4, anticipate that connections to the board and alligator clips will be tricky and inconsistent. Encourage makers to check to see if the connections are secure as they work. You can use a pipe cleaner to twist the alligator clips together so it's easier to see how the wires are connected.

STEP 4

Connect a speaker to the Micro:bit.

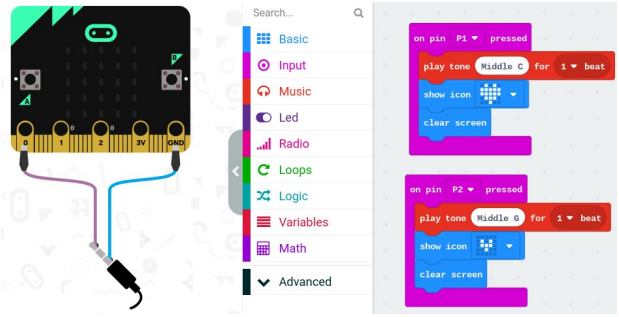


EXPLAIN

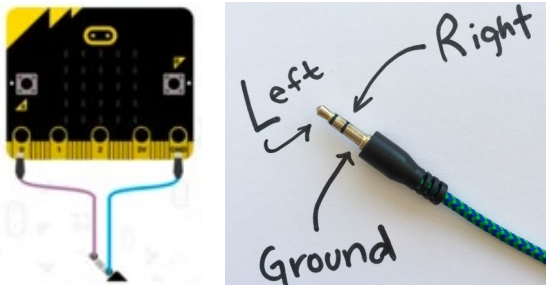
The Micro:bit doesn't have a speaker on it, so even if the code is running, the audio signal has nowhere to be heard or amplified. Next, we'll connect to an external speaker with an audio cable just as is shown on the simulator.

DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

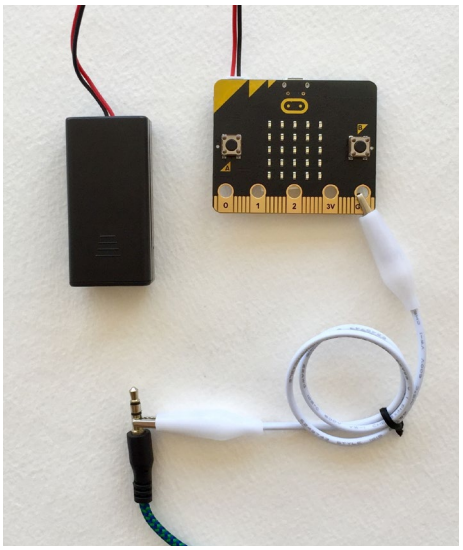
1. Notice that in the simulator there's an image of an audio cable connected to **Pin 0** and **GND.** This is the same type of cord you might find on your headphones or a speaker at home.



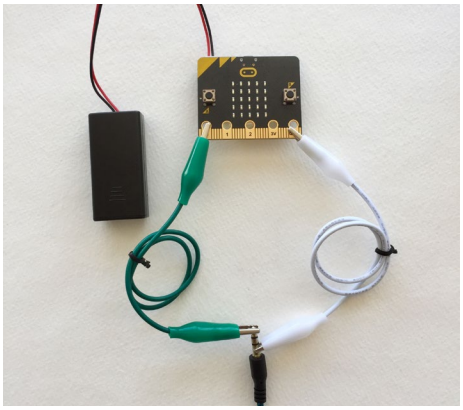
2. If you look closely at the cord from the speaker, you'll see 2 black stripes on the tip of the cable. On this audio plug, the bottom row is ground, and the middle and top of the cable are right and left audio.

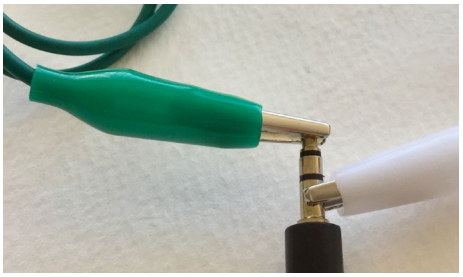


3. The bottom row is ground, so that will connect to **GND** on the Micro:bit. Connect an alligator clip from **GND** on the board to the lower section of the ⅛" plug (ground).



4. Next, connect an alligator clip from **Pin 0** on the board to the top section of the ⅛" plug (left audio).

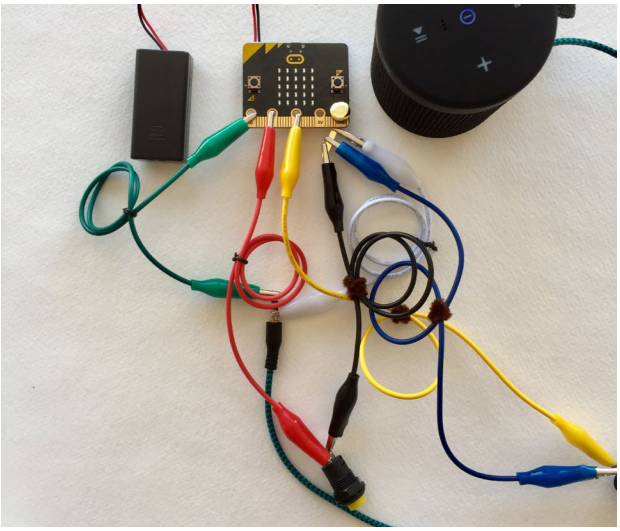




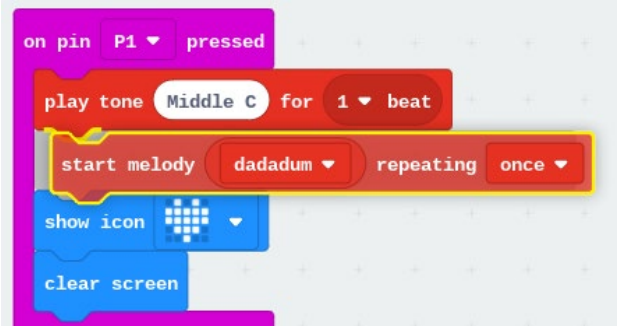
Now, test the sound again by pressing the buttons. You should hear sounds through the speaker!

STEP 5 10 MINUTES

Remix code and add switches and buttons.



Once makers have successfully connected the speaker in Step 3 and can control the sounds with the buttons, they can explore remixing the code. With the remaining time, they should connect the Micro:bit to the computer and explore remixing the code or creating new MakeCode projects.



STEP 6 5 MINUTES

Clean up.

Makers will:

- Make sure to save their files with a unique name.
- Safely disconnect the Micro:bit from the computer.
- Safely disconnect alligator clips from the Micro:bit.
- Put materials away in their bins.
- Put technology away and make sure laptops are charging.
- Return tools and materials that can be used again to the right place.
- Clear tables of garbage and recycling.

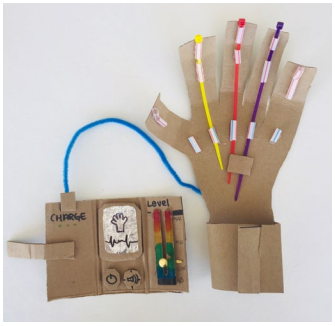
TROUBLESHOOTING TIPS

The board isn't showing what we coded.	**File version check** • Check to see that you've uploaded the most recent copy of the code. • Resave the latest version and drag and drop onto the Micro:bit.
The code isn't doing what we expected.	**Check for bugs** • Read through the code. • Read it out to a friend. • Check to see if there are extra blocks that aren't supposed to be there.
The LED on the Micro:bit is not flashing when we click Upload.	**Bad cable or port** • If the Micro:bit isn't showing up in the computer menu, try a different cable. • Try a different USB port on the laptop.
Our code isn't uploading correctly to the board. The board feels hotter than usual.	**Burnt board** • Try pressing the reset button on the board. • Try uploading to a new Micro:bit board.
The board isn't turning on when connected to the battery pack.	**Battery** • Check the batteries to see if they're charged. • Check to see if the batteries are flipped.
We have alligator clips connected to the board, but the code isn't running.	**Alligator clips** • Make sure alligator clips are secure on the correct pins and are touching the metallic parts. • Try switching alligator clips.
It's hard to keep the connections in place.	**Alligator clips** • Use painter's tape to hold alligator clips in place. • Try using foil to extend the metal parts of the board.

TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

WEEK 6 CARDBOARD ENGINEERING & PROTOTYPES



INTRODUCTION

This week makers are introduced to cardboard engineering and cardboard prototyping.

ESSENTIAL QUESTIONS

- What is a cardboard prototype?
- How do artists, engineers, and makers solve problems when they're working?

LEARNING OUTCOMES

1. Explore how to build and prototype with cardboard.
2. Engage in project-based learning through brainstorming and developing a prototype.

VOCABULARY

Engineering: Applying science and math to solve problems and design machines, structures, and technology

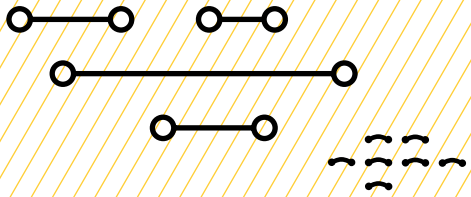
Prototype: Early sample, model, or release of a product built to test a concept or process

User: People using the game or device

User interface (UI): Design of how the user interacts with the game or device

User experience (UX): How natural and enjoyable the experience of using a game or device is

Troubleshooting: Using resources to solve issues as they arise



MATERIALS LIST

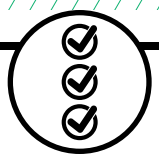
EACH PAIR OF MAKERS NEEDS:

- Scissors
- Cardboard scissors
- Tape (masking, duct, etc.)
- Notebook
- Planning Your Prototype worksheet

ALL MAKERS NEED ACCESS TO:

- Cardboard (corrugated and flat)
- Assorted paper
- Zip ties, rubber bands, popsicle sticks
- Paper fasteners, binder clips, paper clips
- Pipe cleaners, googly eyes, pom-poms, etc.
- Markers or colored pencils
- Bottle caps/recycling (optional)
- Straws (optional)
- Hot glue (See Facilitation Tips.)
- Box cutter and mat (**Teacher use only**. See Facilitation Tips.)

Items can be portioned out per table or set up in an area where students can access them freely.



TEACHER PREP WORK

1. Ensure the internet connection is working and connect your laptop to a projector or screen.
2. Preload videos and slideshow to save time.
3. Arrange maker materials in an area where makers can easily access them.
4. Prepare a few examples of the cardboard-joining techniques shown in the slides (optional).
5. Print and post **Safety Agreement** and **Troubleshooting Tips** (Edit to include your own modifications.)
6. Set up a hot glue station and a box cutter station, covered with newspaper or butcher paper.
7. Print copies of the **Planning Your Prototype worksheet** for each pair of makers.

FACILITATION TIPS

Inspiring creativity: If makers have trouble generating ideas, try brainstorming with them, or ask them, "What devices do you use often? What might those devices look like in 30 years?" It can be helpful to ask them to create a list of "problems" and then brainstorm ideas for devices that could help solve those problems.

Materials management: It's up to you as the educator to decide what works best for your class. You can portion out maker materials into paper trays for each table or have a dedicated area where makers can access materials freely as needed.

Safety: Using hands-on tools is an empowering part of this curriculum. However, practicing safety when working is crucial when using hot glue and sharp tools. You know your makers best, so make adjustments and adaptations as necessary. If makers misuse any tools, have them take a break from the tool and return at your discretion.

Note: If you don't feel comfortable letting makers use hot glue on their own, you can set up an area where you help them hot glue connections they can't achieve in other ways.

Box Cutter (Teacher Use Only): If you're comfortable using a box cutter, you can help makers with cardboard cuts they can't do on their own with scissors. Ask them to draw a visible line with a marker where they want the cut. Encourage them to use the regular and cardboard scissors for most of their other cuts.

Guidelines for using the box cutter:

- Extend the blade of your box cutter out to the minimum needed to cut your material.
- Be sure that the pathway of the knife is not in line with any part of your body, including your other hand and your legs.
- Don't push down hard—instead, take multiple passes to make a cut.
- Retract the knife fully when not in use.
- Pass the knife only when retracted.
- Change dull or dirty blades.

CARDBOARD ENGINEERING AND PROTOTYPES

STEP 1 (5 MINUTES)

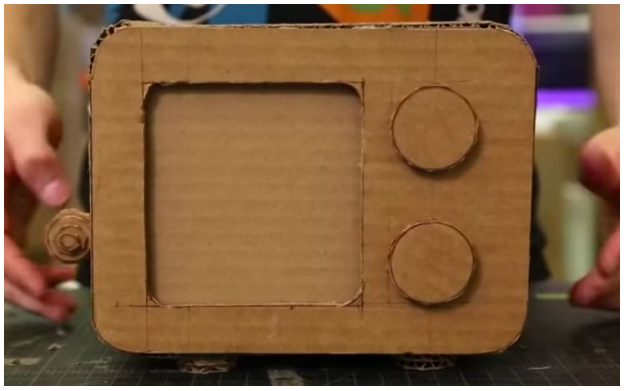
Revisit UI/UX.

Review the following concepts from the past weeks:

- **User:** Person using the game or device
- **User interface (UI):** Design of how the user interacts with the game or device
- **User experience (UX):** How natural and enjoyable the experience of using a game or device is

EXPLAIN

UI/UX designers and engineers think through the design of devices such as game controllers, phones, and appliances to make them easy and fun to use. In order to explore ideas, they **prototype** with simple materials, like cardboard and paper.



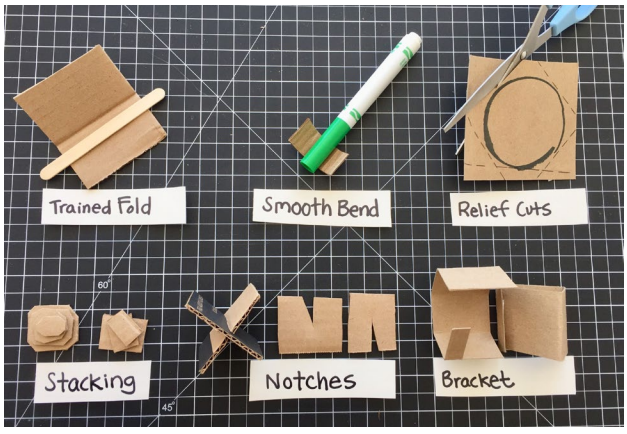
"How to make a cardboard prototype" on YouTube, uploaded by Quirky, 12/11/2014

On a projector, show this video, which explains the purpose of cardboard prototyping. While watching, ask makers to answer the following questions in their journals and then share the answers out loud:

1. What's the first step of prototyping before using cardboard?
   (A: Draw out your ideas to get an estimate of size and scale.)
2. What's the purpose of prototyping?
   (A: To get the idea out of your head and in a form that people can see and experience.)

STEP 2 (10 MINUTES)

Introduce cardboard engineering and safety.



First, go through the slideshow of cardboard **engineering** techniques

(trained folds, smooth bend, relief cuts, stacking, notches, bracket, etc.), emphasizing how they could be useful.

EXPLAIN

Now that we've programmed and connected buttons and switches to the Micro:bit, we'll explore prototyping devices with cardboard. Prototyping is all about exploring new ideas, so don't be afraid to try something and learn from any "mistakes." Even the most experienced engineers build off of "failed" prototypes.

Next, introduce the safety agreements of working in an active "make space." Review the safety agreements with makers. Add and modify as needed for your class, and then print out and post up where they can be clearly seen.

SAFETY AGREEMENT

1. Take care when walking with scissors or sharp things (hold with point facing down).
2. One maker at a time per tool prevents accidents.
3. Be mindful of space from others when using tools.

GLUE GUN SAFETY

- Only 1–2 makers at the hot glue station at a time.
- Don't touch the tip of the glue gun.
- Don't point the glue gun at another person.
- Work at the protected glue gun station.
- Keep the glue gun close to your work.
- If the glue gun jams, ask an adult for support.

EXPLAIN

Before we get started, we need to cover some important safety agreements. This will help everyone stay safe and do their best work while sharing space, tools, and materials. After reviewing these, if anyone isn't following the agreements, they will need to take a break from using the tool, not as "punishment" but as a way of creating safe habits.

STEP 3 5 MINUTES

Introduce the Prototype for the Future project.

Go through the slides, emphasizing how cardboard prototypes are a way of exploring an idea for a device and not necessarily intended to be fully functional.

Explain to makers that their prototype designs must:

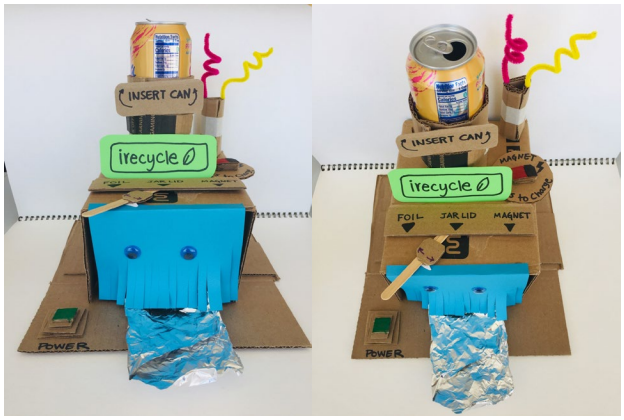
1. Use 3 or more cardboard engineering techniques.
2. Explain how the user would interact with the device.
3. Include ideas for how to add a Micro:bit to their prototype.

EXPLAIN

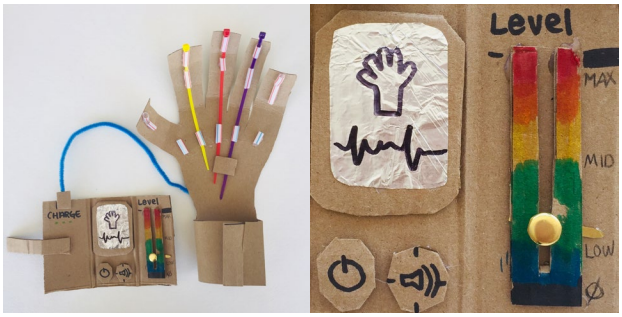
In the year 2050, what kinds of inventions do you think would be useful that don't exist now? You and

your partner will brainstorm ideas of devices for everyday life in the future.

Today, you and your partner will design and make a prototype using cardboard building techniques. In the next session, you'll add a Micro:bit and maybe even buttons or switches!



An example of an invention is "iRecycle," an at-home recycler that turns aluminum cans into foil, jar lids, or magnets.



Another example is this **robotic helping hand** with a remote that can be programmed. This device could be helpful for situations where using human hands would be dangerous or not strong enough.

Makers will:

- Brainstorm with their partner about what they want to design/make and take notes using this worksheet.
- Determine who will do what to start and then gather materials.

STEP 4 35 MINUTES

Make cardboard prototypes.

Makers use this time to build with cardboard engineering techniques.

- Circulate throughout the room and watch for safe working habits. Take note and verbally point out when safety agreements are being followed. If any maker misuses a tool, have them take a break or remove the tool temporarily (2–3 minutes) to help reinforce a culture of safety.
- Circulate throughout the room noting and celebrating when makers are using the various building techniques. Offer suggestions of techniques that are

appropriate for what the makers are building.

- If you're assisting makers by making cuts with the box cutter, create a signup list on the board. Send makers to try again when you think they can handle the cut on their own using scissors. Be sure to keep the box cutter closed and in your possession.

STEP 5

Clean up.

Makers label and save their prototypes to continue working on the next session.

Makers will:

1. Put projects away in bins. Label parts they want to keep that don't fit in the bins.
2. Return tools and materials that can be used again to the right place.
3. Clear tables of garbage and recycling.

TROUBLESHOOTING TIPS

Cardboard is difficult to cut.

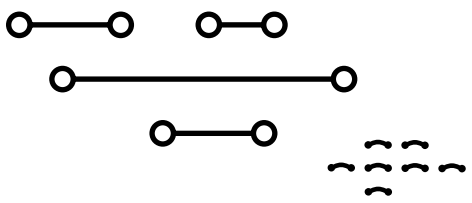
- Using the inside of the scissors instead of the tip can make cutting easier.
- Cutting pieces away from the edge of the cardboard is easier than cutting out a shape from the middle of the cardboard.
- If you're really having a hard time, ask a classmate or adult to help you with cuts.

Hot glue isn't holding stuff in place.

- Hot glue dries quickly, so try to apply the glue a little at a time instead of large amounts.
- After gluing, hold the pieces in place for at least 20 seconds before releasing.
- Support two pieces with an L-bracket, or bridge with glue or tape.

Cardboard won't hold the shape.

- Try experimenting with a different joining technique.
- Try different thicknesses of cardboard or layers of cardboard.



TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

PLANNING YOUR PROTOTYPE

Title_____

Draw a picture of your **Prototype for the Future** below.

1. Describe what your prototype does and how the user interacts with it.

2. What are some of the cardboard building techniques you will use?

3. How could you use a Micro:bit in your project?

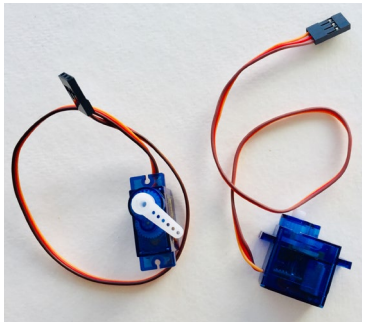
SAFETY AGREEMENT

1. Take care when walking with scissors or sharp things (hold with point facing down).
2. One maker at a time per tool prevents accidents.
3. Be mindful of space from others when using tools.

GLUE GUN SAFETY

1. Only 1 or 2 makers at the hot glue station at a time.
2. Don't touch the tip of the glue gun.
3. Don't point the glue gun at another person.
4. Work at the protected glue gun station.
5. Keep the glue gun close to your work.
6. If the glue gun jams, ask an adult for support.

WEEK 7 INTRODUCTION TO SERVOS



INTRODUCTION

In this session, makers are introduced to programming servo motors with the Micro:bit.

The servo motor can be a neat addition to the options for coding the Micro:bit, in preparation for the Cyber Arcade projects they'll be working on in the following weeks.

ESSENTIAL QUESTIONS

- What is a servo motor and how can I control it with code?
- How do artists, engineers, and makers solve problems when they're working?

LEARNING OUTCOMES

1. Learn how to use code to program a servo motor.
2. Engage in project-based learning through problem-solving and troubleshooting by creating a game using a Micro:bit and code.

VOCABULARY

Servo motor: Motor that can be programmed with an electrical signal to move to a specific position

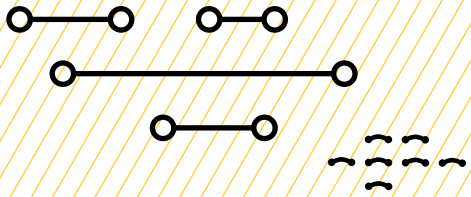
Servo mounting horn: Attaches to the servo to allow other objects to be connected to the servo (e.g., a wheel or an arm)

User: Person playing and interacting with the game

User interface (UI): Physical and digital design of how the user interacts with the game

User experience (UX): How natural and enjoyable the experience is

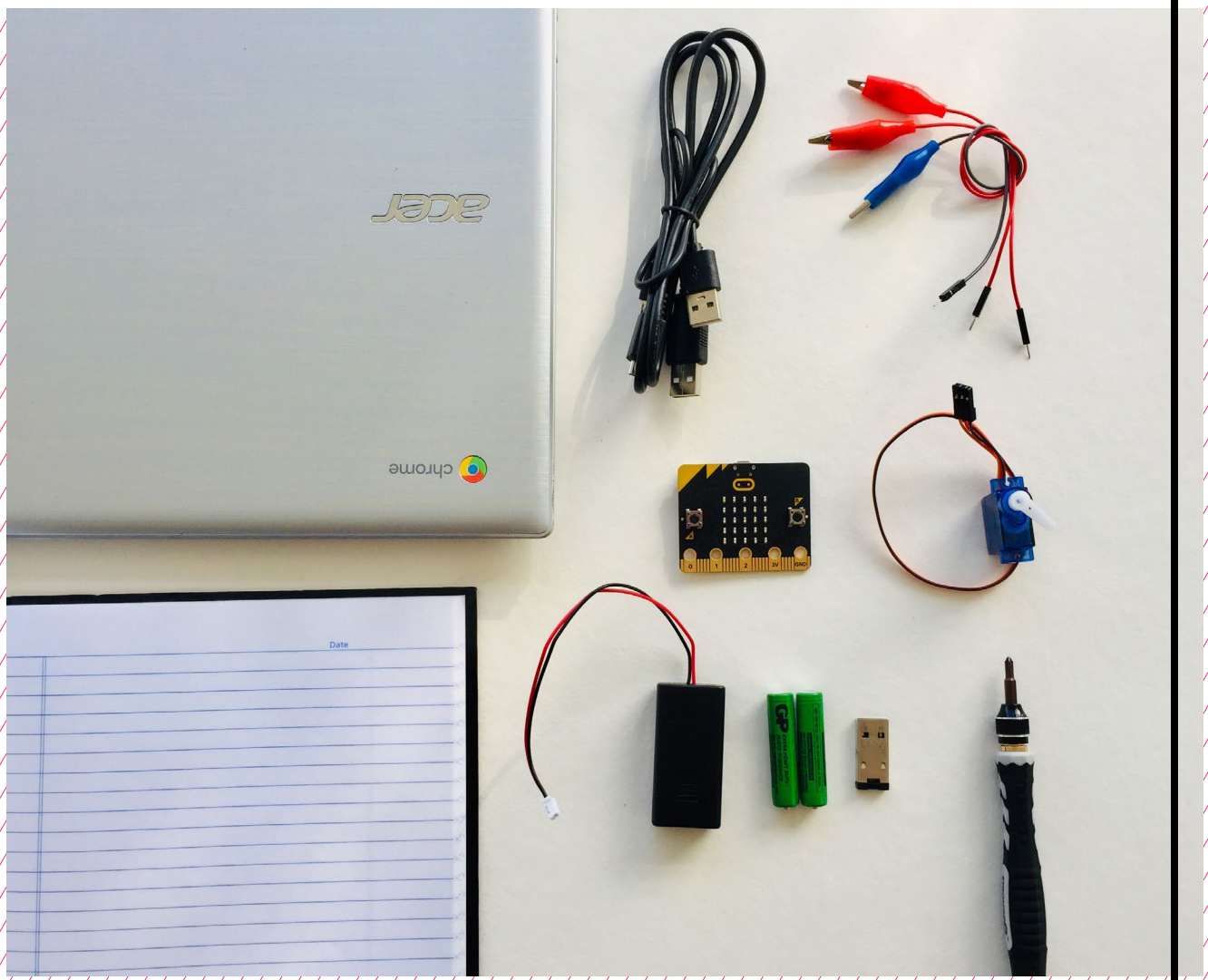
Troubleshooting: Using resources to solve issues as they arise



MATERIALS LIST

EACH PAIR OF MAKERS NEEDS:

- Micro:bit microcontroller
- AAA batteries (2)
- External battery pack
- Alligator-to-pin wires (3)
- Servo motor with servo horns
- Laptop with internet connection
- USB to micro-USB cord
- USB flash drive
- Notebook
- Mini screwdriver



TEACHER PREP WORK

1. Prepare the projector and make sure the internet is working.
2. Preload the slideshow and videos to save time.
3. Prepare an example of a servo with code from Steps 2 and 3 (optional).
4. Attach servo mounting horns to servo motors using a small screwdriver.
5. Print the Troubleshooting Tips at the end of the lesson and post in the classroom.

FACILITATION TIPS

Tinkering with electronics: Tinkering with electrical connections on a servo can be a bit tricky and won't work if things aren't securely connected. If students get frustrated, encourage them to check the color coding of the wires throughout the process and to also use the Troubleshooting Tips. They can use painter's tape or other non-permanent ways of securing connections.

Materials management: It's up to you as the educator to decide what works best for your class. You can portion out maker materials into paper trays for each table, or have a dedicated area where makers can access materials freely as needed.

Collaboration: Let smaller issues work themselves out. Record specific positive examples that you can share with makers in the moment or at the end of the project. These examples provide models for all learners.

Frustration: When frustration levels aren't high, let learners figure it out or keep facilitation low touch by asking a question and walking away. When frustration levels are high, intervene more directly to help makers find some success.

Circulate among the makers and monitor for both collaboration and frustration.

ADDITIONAL RESOURCES

Behind the MakeCode Hardware: Servo Motors with Micro:bit

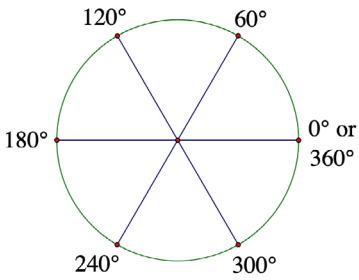
Driving a Servo with the Micro:bit

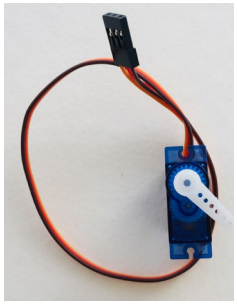
Connecting a Servo Motor to the Micro:bit

INTRODUCTION TO SERVOS

STEP 1 (5 MINUTES)

Introduce the servo motor.





Ask makers:

- What is a motor? *(A: Device that converts electrical energy into rotating mechanical energy.)*
- What kinds of devices have motors in them? (Students should brainstorm.)

Give each group of makers a **servo motor** with a **servo horn** attached. Makers will draw and list parts they notice in their notebook.

Note: Advise makers to handle gently and to not force the turning of the arm, as it could break.

Ask makers to:

- Study the servo parts closely.
- Describe what they see/notice.

The parts to emphasize are: the 3 colored wires, gears, plastic arms, and the fact that the arm turns.

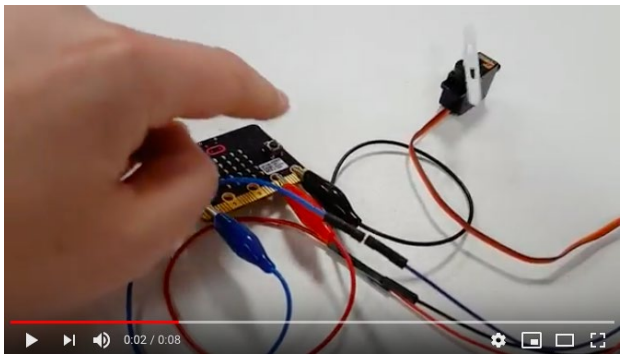
EXPLAIN

Today we'll connect and program movement using a special motor called a servo motor. A servo motor can be programmed to turn a specific number of degrees up to 180°, a full circle. We can use servos in addition to everything else you've done so far to create fun and interesting **user interface/user experience (UI/UX)** for interactive games or devices.

STEP 2 (20 MINUTES)

Program the servo motor.

Show makers this short **video** that demonstrates Micro:bit buttons A/B controlling a moving servo motor.



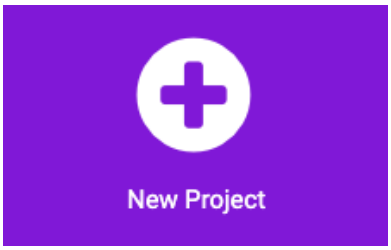
"**Servo Attached to Micro:bit**" on YouTube, uploaded by Teach with ICT, 6/25/2018

EXPLAIN

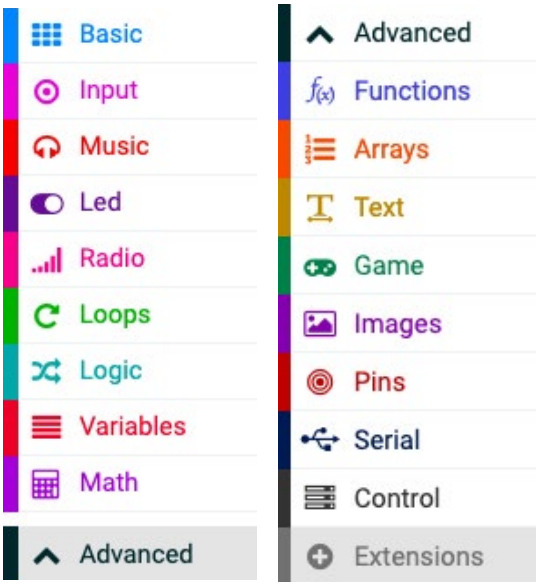
We can program a servo to turn a specific amount by connecting it to the Micro:bit board and using special servo coding blocks. First, let's work with the code in the simulator. Then, we'll upload it to the board and connect the servo to the board.

DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

1. Go to the MakeCode editor URL (makecode.microbit.org) and open a new project.



2. In the blocks menu, click on **Advanced** and then scroll down to click on **Extensions**.



3. This will take you to a new page. Click on "servo A micro-servo library". This will add the additional blocks we need to program the servo.

servo

A micro-servo library

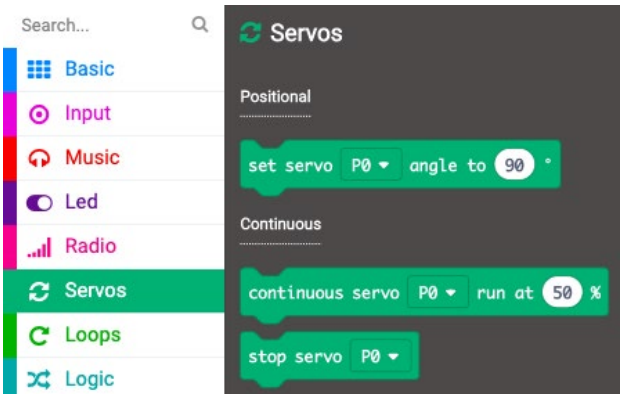
4. Click and drag over a **when button A is pressed** from the **Inputs** menu.

5. Drag over a **pause** block from the **Basic** menu and nest it in the first block.

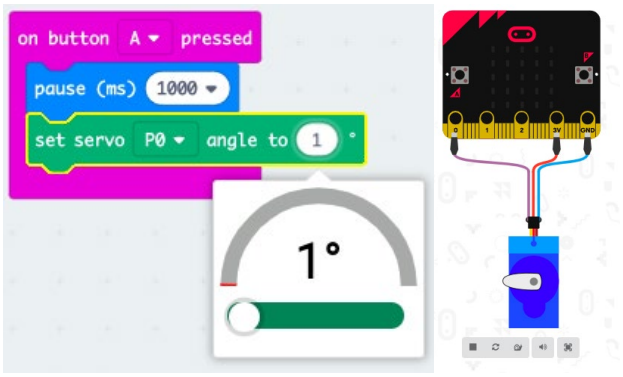
6. Change the pause time to be 1 sec (or 1000 ms).

7. Notice there's a new **Servos** block menu. Click and drag over a **set servo**

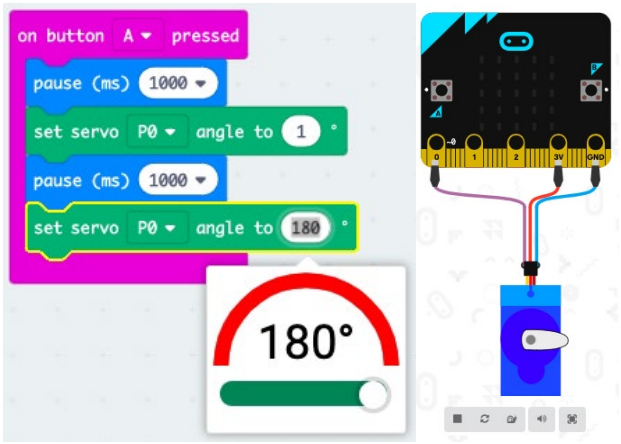
P0 angle to block under the **pause** block.



8. Click on the number 90 within the block and change it to be the number 1. Then you'll notice a servo appear in the simulator.



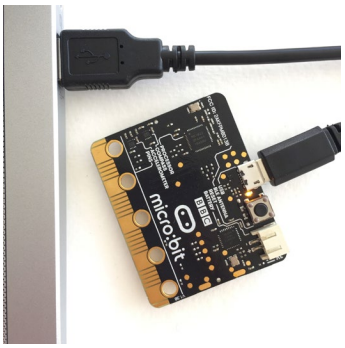
9. Repeat the **pause** and **set servo** blocks by duplicating or dragging over new blocks. This time, change the value in the **set servo** block to be 180. Click on **button A** in the simulator, and you'll notice that the servo will move to the opposite side.



10. Repeat Steps 8 and 9 two more times. You'll have a series of blocks that look like the blocks below. Again, click on **Button A** in the simulator and you should see the servo arm move back and forth twice.

11. Name your file with a unique name and save the file to the USB flash drive. Connect the Micro:bit with the USB cord.

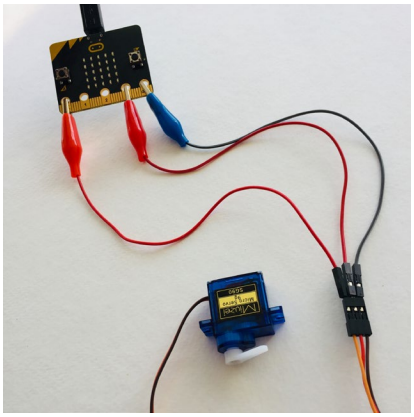
Click and drag the file on to the Micro:bit to upload.



Note: Leave the Micro:bit connected to the laptop for the next step.

STEP 3 20 MINUTES

Connect the servo motor.



Once makers have successfully programmed a servo in the simulator and uploaded the code to the Micro:bit, it's time to connect the physical servo to the board. The Micro:bit should still be connected to the laptop.

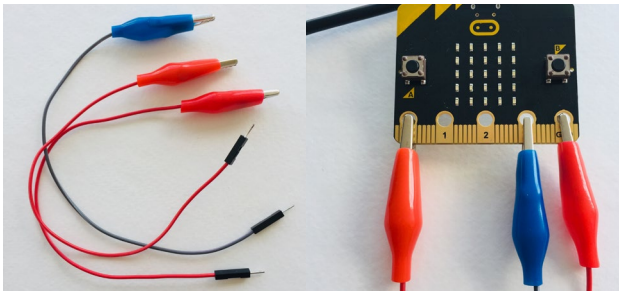
Give each group 3 alligator-to-pin wires, one servo, and a battery pack.

EXPLAIN

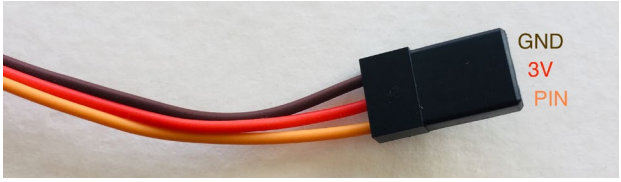
Now that we've tested our code and uploaded it to the Micro:bit, we'll connect the servo.

DEMONSTRATE AND HAVE MAKERS FOLLOW ALONG:

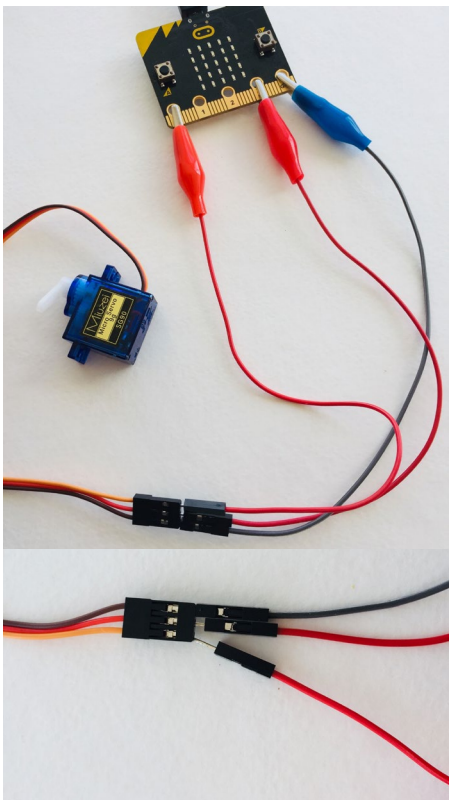
1. In order for the servo to get the code, we need to connect 3 of the alligator-to-pin wires to the GND, 3V, and PIN 0 terminals on the Micro:bit board using the alligator side of the wire.



2. Next, notice on the servo there are 3 colored wires: black, red, and orange. These wires are important to keep track of. The brown wire is the ground wire (GND) or the (–) terminal and the red wire is the 3V or (+) terminal. These help to deliver electricity and power to the servo. The orange wire is the signal wire, or the wire that actually delivers the code from the programmed pin on the Micro:bit to the servo.

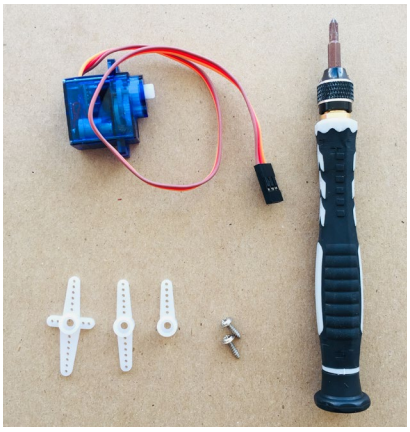


3. Connect your servo wires to the Micro:bit board following the table below. If you can match the colors of the servo motor, great! If not, the colors of the alligator-to-pin wires don't actually matter—it's just helpful to use the same colors so we can easily track where the wires are going.

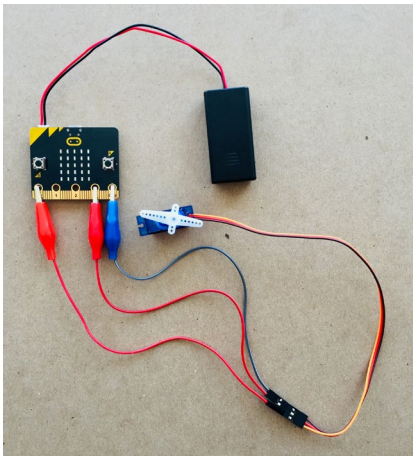


GND	Brown
3V	Red
PIN	Yellow

4. Once you think you have the connections right, try pressing Button A and check if the servo is moving or not. You'll hear a mechanical rotating sound if the servo is working. If your servo doesn't already have a plastic mounted "horn" piece, then use a mini screwdriver to attach one so that it's easier to see the movement.

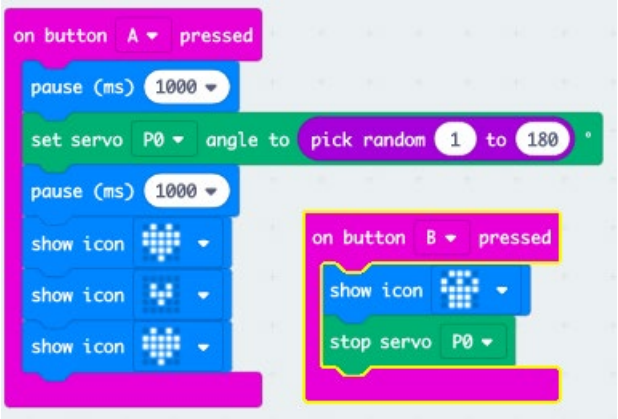


5. If the servo works as displayed on the computer simulator, you can disconnect the Micro:bit and connect it to a battery pack.
6. If the servo isn't moving, check your wiring to see if the connections are right (Steps 2 and 3). If the servo is still not moving from 0 to 180 degrees, then check the code on the screen.



STEP 4 (10 MINUTES)

Further explore programming the servo.



Once makers have successfully programmed and connected a servo, have them go back and tinker with the code (e.g.,. change inputs, add displays, move a different amount of degrees, etc.). Remind them that they'll need to plug their Micro:bit back into their computer.

STEP 5 (5 MINUTES)

Clean up.

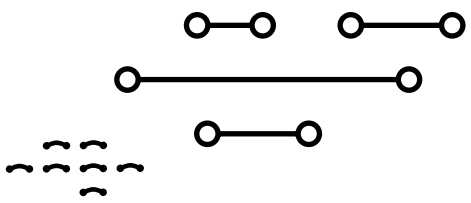
Makers will:

- Save their file with a unique name onto the USB flash drive.
- Safely disconnect the Micro:bit from the computer.
- Put materials away in their bins.
- Put away technology and make sure laptops are charging.
- Return tools and materials that can be used again to the right place.
- Clear tables of garbage and recycling.

TROUBLESHOOTING TIPS

The servo isn't moving or responding.	**Alligator-to-pin wires** • Check to see that the wires are properly connected to the servo (GND to brown, 3V to red, and Pin to orange). • Make sure alligator clips are secure on the correct pins and are touching the metallic parts. Try extending from the board with a paper fastener. • Try switching out the alligator clips for new ones. **Power** • The servo might not be getting enough power from the battery pack. Try powering the Micro:bit using the USB to the laptop. If it works, replace the batteries or the battery pack and try again. • Try pressing the reset button on the Micro:bit.
The servo is making some sound but not rotating.	**Bugs in the code** • Read through the code. • Check to make sure there's a short pause block in the code before and after every turn. The servo cannot move without a pause before and after moving. • The servo gears could be burnt out or broken. Try replacing the servo.
The servo isn't turning the amount we thought it should.	**Servo type** • Try connecting a different servo and try again. • Servo gears could be uncalibrated or broken. Replace the servo and try again.

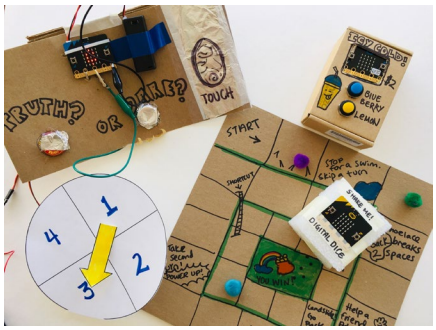
The LED on the Micro:bit isn't flashing when we click Upload.	**Bad cable or port** • If the Micro:bit isn't showing up in the computer menu, try a different cable. • Try a different USB port on the laptop.
Our code isn't uploading correctly to the board. The board feels hotter than usual.	**Burnt board** • Try pressing the reset button on the board. • Try uploading to a new Micro:bit board. • Check to see if there's conductive material touching the board. It could be causing a short circuit.
The board isn't turning on when connected to the battery pack.	**Battery** • Test the batteries to see if they're charged. • Check to see if the batteries are flipped.
The LED on the Micro:bit isn't flashing when we click Upload.	**Bad cable or port** • If the Micro:bit isn't showing up in the computer menu, try a different cable. • Try a different USB port on the laptop.
The connections won't stay in place.	**Alligator clips** • Use painters tape to hold the alligator clips in place. • Try using foil or paper fasteners to extend the metal parts of the board (Pin, GND, 3V).



TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

WEEK 8 INTRODUCING THE CYBER ARCADE



INTRODUCTION

This week makers begin planning the culminating event, the Cyber Arcade! Makers work together with a project-based approach, designing interactive games using the Micro:bit and maker materials. For the final three weeks, makers synthesize ideas and concepts explored in the past weeks through a repeating cycle of designing, making, and discussion.

ESSENTIAL QUESTIONS

- What is a game or interactive project we want to share with our community?
- What technology and tools do we want to explore further?
- How do artists, engineers, and makers solve problems when they're working?

LEARNING OUTCOMES

1. Design, plan, and build interactive projects for a community event.
2. Engage in project-based learning through problem-solving and troubleshooting by creating a game using a Micro:bit microcontroller and code.

VOCABULARY

Project planning: Process of clarifying goals and listing the steps and materials required to complete a project

Brainstorm: Thinking about and coming up with many ideas or solutions, either on your own or in a group

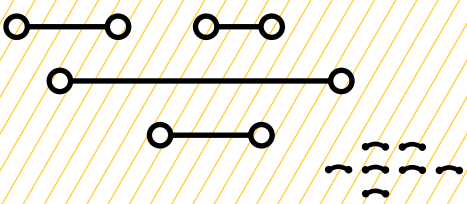
Game mechanics: Basic actions, processes, visuals, and control mechanisms that are used to make a game

Game designer: Person responsible for designing game storylines, plots, objectives, scenarios, degree of difficulty, and character development

Game engineer: Specialized software engineer who designs and programs video games

Pseudocode: Detailed, informal description of what a computer program must do

Troubleshooting: Using resources to solve issues as they arise



MATERIALS LIST

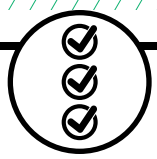
EACH PAIR OF MAKERS NEEDS:

- Micro:bit microcontroller
- External battery pack
- AAA batteries (2)
- Laptop with internet connection
- USB to micro-USB cord
- USB flash drive
- Notebook
- Markers
- Colored pencils
- Scissors
- Cardboard scissors

ALL MAKERS NEED ACCESS TO:

- Alligator clips
- Alligator-to-pin wires
- Buttons (momentary and locking)
- Servos
- Tape (masking, painters, duct)
- Aluminum foil
- Assorted cardboard
- Assorted paper
- Pipe cleaners, pom-poms, popsicle sticks
- Misc bottle caps/recycling (optional)
- Hot glue gun and glue sticks (see Facilitation Tips)

Items can be portioned out per table or set up in an area where students can access them freely.



TEACHER PREP WORK

1. Ensure the internet connection is working and connect your laptop to a projector or screen.
2. Preload videos and slideshows to save time.
3. Set up an equitable system for student access of materials (either as a materials area or distributed evenly per table).
4. Set up a hot glue station and/or box cutter station, covered with newspaper or butcher paper.
5. Print the **Brainstorm and Project Planning worksheet** for each group, along with a few extras.
6. Print the **Troubleshooting Tips** the end of the lesson and post in the classroom.

FACILITATION TIPS

Inspiring creativity: If makers have trouble generating ideas, try brainstorming with a pair that is struggling or with the whole group, prompting them with questions like "What games or activities would you see at a real arcade or carnival?" If they continue to struggle, share a list of **examples**.

Safety: Using hands-on tools is an empowering part of this curriculum. However, practicing safety while working is crucial when using hot glue and sharp tools. You know your makers best, so make adjustments and adaptations as necessary. If makers misuse any tools, have them take a break from the tool and return at your discretion.

Note: If you don't feel comfortable letting makers use hot glue on their own, you can set up an area where you help them hot glue connections they can't achieve in other ways.

Box cutter (Teacher Use Only): If you're comfortable using a box cutter, you can help makers with cardboard cuts they can't do on their own with scissors. Ask them to draw a visible line with a marker where they want the cut. Encourage them to use the regular and cardboard scissors for most of their other cuts.

Guidelines for using the box cutter:

- Extend the blade of your box cutter out to the minimum needed to cut your material.
- Be sure that the pathway of the knife is not in line with any part of your body, including your other hand and your legs.
- Don't push down hard—instead, take multiple passes to make a cut.
- Retract the knife fully when not in use.
- Pass the knife only when retracted.
- Change dull or dirty blades.

ADDITIONAL RESOURCES

Wonderful Idea Co: Computational Carnival

Driving a Servo with the Micro:bit

INTRODUCING THE CYBER ARCADE

STEP 1 5 MINUTES

Introduce the Cyber Arcade.

"Caine Monroy: Inventors Challenge 2019" on YouTube, uploaded by Imagination Fdn, 5/24/2019

Show this video of Caine Monroy to introduce the idea of a cardboard challenge and arcade games.

EXPLAIN

Caine Monroy became famous for his creativity at the age of nine, but he has never stopped learning and imagining. Here, he explains why Arcade Games are important to him and his community.

We've done so much learning and exploring with the Micro:bit together in the last 7 weeks! First, give your partner, classmates, and yourself a big high five!

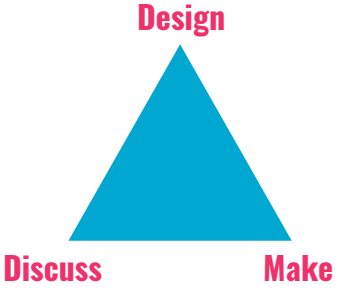
We'll use the rest of the sessions to apply what we've learned to create Micro:bit interactive arcade games to host a Cyber Arcade event! We'll have 3 weeks to design and make, and on the last day we'll have an open arcade and play each other's games!

With the corresponding slides, briefly review the skills and concepts that have been covered in the last 7 weeks.

- Week 1: Designed and connected **LED circuits**
- Week 2: Programmed a Micro:bit with **inputs** and **outputs**
- Week 3: Designed games with **conditionals** and **number ranges** and **game mechanics**
- Week 4: Connected and programmed **homemade switches** and **arcade buttons**
- Week 5: Connected **speakers** and **programmed sounds**
- Week 6: Explored **cardboard engineering** to build **prototypes**
- Week 7: Connected and programmed **servo motors**

STEP 2 10 MINUTES

Brainstorm.



Give each group a copy of the **Brainstorm for the Cyber Arcade worksheet**.

First, makers work together to **brainstorm** at least 4 different ideas. Then, through discussion, they choose one idea to work on. If makers have trouble generating ideas, try brainstorming with a pair that is struggling or with the whole group, prompting them with questions like "What games or activities would you see at a real arcade or carnival?" If they continue to struggle, share a list of **examples**.

EXPLAIN

For the next 3 weeks, you and your partner will design, make, and code an interactive game or activity for visitors of the Cyber Arcade! Think about what kind of game or experience you want to design. Who are you designing for? What kinds of things would you see at a real arcade, amusement park, or carnival?

Here's the criteria for what your project will include:

1. **Construction:** Use cardboard engineering techniques.
2. **Inputs:** Include at least one programmed input.
3. **Programming**: Use either conditional and/or number ranges for the game mechanics.
4. **Electronic components**: Include at least one programmed button or switch, speaker, or servo. You can include as many as you like.

There are 3 phases we cycle through as we work: **design**, **make**, and **discuss**.

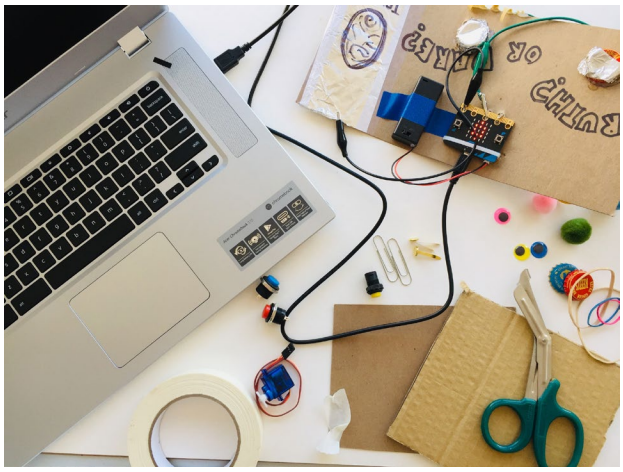
1. **Design:** Brainstorm ideas. What kind of game or experience do you want the user to have? What are its inputs and outputs? What materials will you need? How will you program it?
2. **Make:** You and your partner are doing the work of programming, making and building. You're troubleshooting and solving problems as you work. You can assign tasks for this so it's clear who is doing what and when.
3. **Discuss:** It's important that you and your partner communicate, especially when troubleshooting and solving problems. You may need to stop to reflect and make changes to your designs as you work.

To begin, you and your partner will brainstorm a list of ideas for a Cyber Arcade project. Once you decide on an idea, you'll work together to create a **project plan** to help keep you on track for the next few weeks.

In Step 1 on your worksheet, come up with at least 4 different project ideas and describe how the user would interact with each one.

STEP 3 40 MINUTES

Create a project plan and make.



Once groups have chosen an idea to work on, it's time to create a project plan by filling out the **Project Plan** section of the worksheet. Makers will share their idea with the rest of the group to ensure that groups aren't doing the same project. All the groups are working collaboratively to create a variety of games for the Cyber Arcade!

EXPLAIN

Once you and your partner choose an idea you want to work on, it's time to write up a project plan. In almost any professional job or sport, teams use project plans to make sure that all the different tasks get done to complete a project. Be open to changing your original plan as you make. We'll have limited time to work on these, so you might need to simplify for the most important parts of your project. Also, figure out who will do what in order to get your project ready in time to present.

Makers work together to fill out the brainstorming and project plan sheets. Once makers have filled out these sheets and shared them with you for review, they can begin gathering materials.

STEP 4 5 MINUTES

Clean up.

Makers will:

- Put any materials they want to keep to use in their partner bin.
- Put away technology and make sure laptops are charging.
- Return tools and materials that can be used again to the right place.
- Clear tables of garbage and recycling.

PROJECT IDEAS

Jukebox	Program 2 buttons to play music and animate a display.
Food Stand	Use a homemade switch as an input to place an order, and use an animated display or sound to show when food is ready.
Strength Test	Use a homemade switch to choose a random number to show how "strong" the user is.
Digital Dice	Program the Micro:bit as a dice, and create game mechanics for the numbers.
Racing Game	Use 2 Micro:bits programmed to show random numbers that display how many spaces the cars should move.
Truth or Dare	Program 2 switches, where the user chooses T or D and the numbers mean different truth or dares.
Fortune Teller	The Servo Spinner chooses a random number to turn to, which points to a fortune.
Spaceship Ride	Build a mini cardboard spaceship with buttons and an interactive LED screen.
Cotton Candy	Program a switch to move the servo that moves cotton candy to the user.
Ticket Counter	Program the Micro:bit to show the number of tickets to give to a user.

DESIGN AND DISCUSS — BRAINSTORM FOR THE CYBER ARCADE

Step 1: Brainstorm.

Make a list of ideas. This is not the time to judge ideas or think too much about it. Write down as many ideas as you can come up with (for example, Truth or Dare, Strength Meter, Fortune Teller).

What is the project idea?	How would the user interact with it?
1.	1.
2.	2.
3.	3.
4.	4.

Step 2: Discuss and decide.

From the list above, discuss with your partner and come to an agreement on which idea you want to move forward with. Use the space below to draw and take notes as you talk.

DESIGN AND DISCUSS — PROJECT PLAN FOR THE CYBER ARCADE

Title_____

1. In a few sentences, describe your Cyber Arcade project.

2. How will you build it? What engineering techniques do you plan to use?

CYBER ARCADE — TECHNICAL PLAN

3. How will you program your game? (Write the pseudocode of the number ranges or conditionals.)

4. List what inputs and outputs you'll program.
Inputs: How will the user interact with the project? (buttons, shake, etc.)
Outputs: What happens when they interact with it? (shows animation, plays sound, etc.)

INPUTS *(Example: Press a momentary switch.)*	**OUTPUTS** *(Example: Will give you a fortune.)*

5. Which parts will you use? (switch, button, speaker, servo)

SAFETY AGREEMENT

1. Take care when walking with scissors or sharp things (hold with point facing down).
2. One maker at a time per tool prevents accidents.
3. Be mindful of space from others when using tools.

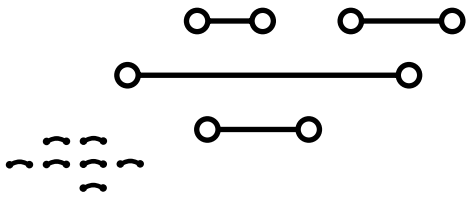
GLUE GUN SAFETY

1. Only 1 or 2 makers at the hot glue station at a time.
2. Don't touch the tip of the glue gun.
3. Don't point the glue gun at another person.
4. Work at the protected glue gun station.
5. Keep the glue gun close to your work.
6. If the glue gun jams, ask an adult for support.

TROUBLESHOOTING TIPS

The cardboard is difficult to cut.	• Using the inside of the scissors instead of the tip can make cutting easier. • Cutting pieces away from the edge of the cardboard is easier than cutting out a shape from the middle of the cardboard. • If you're really having a hard time, ask a classmate or adult to help you with cuts.
The hot glue isn't holding stuff in place.	• Hot glue dries quickly, so try to apply the glue a little at a time instead of large amounts. • After gluing, hold the pieces in place for at least 20 seconds before releasing. • Support two pieces with an L-bracket or a bridge, using glue or tape.
The cardboard won't hold the shape.	• Try experimenting with a different joining technique. • Try different thicknesses of cardboard or layers of cardboard.
The board isn't showing what we coded.	**File version check** • Check to see that you uploaded the most recent copy of the code. • Resave the latest version and drag and drop it onto the Micro:bit.
Our code isn't doing what we expected.	**Check for bugs** • Read through the code. • Read it out to a friend. • Check to see if there are extra blocks that aren't supposed to be there.

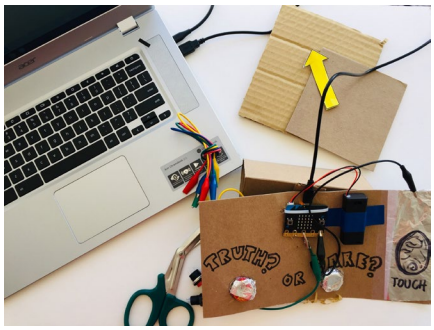
The LED on the Micro:bit isn't flashing when we click Upload.	**Bad cable or port** • If the Micro:bit isn't showing up in the computer menu, try a different cable. • Try a different USB port on the laptop.
Our code isn't uploading correctly to the board. The board feels hotter than usual.	**Burnt board** • Try pressing the reset button on the board. • Try uploading to a new Micro:bit board.
Our board isn't turning on when connected to the battery pack.	**Battery** • Check the batteries to see if they're charged. • Check to see if the batteries are flipped.
We have alligator clips connected to the board, but the code isn't running.	**Alligator clips** • Make sure alligator clips are secure on the correct pins and are touching the metallic parts. • Try switching alligator clips.



TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

WEEK 9 DAY 1: CYBER ARCADE WORK SESSIONS



INTRODUCTION

This week makers continue working on their interactive games with a project-based approach. For the final two weeks, makers are experimenting with and synthesizing concepts explored in past weeks.

ESSENTIAL QUESTIONS

- What is a game or interactive project we want to share with our community?
- What changes do we make to our project as we design, make, and discuss?
- How do artists, engineers, and makers solve problems when they're working?

LEARNING OUTCOMES

1. Design, plan, and build interactive projects for a community event.
2. Engage in project-based learning through problem-solving and troubleshooting by creating a game using a Micro:bit microcontroller and code.

VOCABULARY

Project planning: Process of clarifying goals and listing the steps and materials required to complete a project

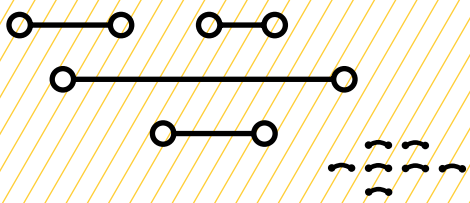
Game mechanics: Basic actions, processes, visuals, and control mechanisms that are used to make a game

Game designer: Person responsible for designing game storylines, plots, objectives, scenarios, the degree of difficulty, and character development

Game engineer: Specialized software engineers who design and program video games

Pseudocode: Detailed, informal description of what a computer program must do

Troubleshooting: Using resources to solve issues as they arise



MATERIALS LIST

EACH PAIR OF MAKERS NEEDS:

- Micro:bit microcontroller
- External battery pack
- AAA batteries (2)
- Laptop with internet connection
- USB to micro-USB cord
- USB flash drive
- Notebook
- Markers
- Colored pencils
- Scissors
- Cardboard scissors

ALL MAKERS NEED ACCESS TO:

- Alligator clips
- Alligator-to-pin wires
- Buttons (momentary and locking)
- Servos
- Tape (masking, painters, duct)
- Aluminum foil
- Assorted cardboard
- Assorted paper
- Pipe cleaners, pom-poms, popsicle sticks
- Misc bottle caps/recycling (optional)
- Hot glue gun and glue sticks (see Facilitation Tips)

Items can be portioned out per table or set up in an area where students can access them freely.

TEACHER PREP WORK

1. Set up an equitable system for student access of materials (either as a materials area or distributed evenly per table).
2. Set up a hot glue station covered with newspaper or butcher paper.
3. Print the Troubleshooting Tips and Safety Agreement at the end of the lesson and post in the classroom.

FACILITATION TIPS

Safety: Using hands-on tools is an empowering part of this curriculum. However, practicing safety when working is crucial when using hot glue and sharp tools. Box cutters are for teacher use only! If you're comfortable with using one, you can help makers with cardboard cuts they can't do on their own. If you don't feel comfortable with letting makers use hot glue on their own, you can set up an area where you help them. You know your makers best, so make adjustments and adaptations as necessary. **If makers misuse any tools, have them take a break from the tool and return at your discretion.**

Supporting a maker mindset: Project-based making and coding is all about learning by making mistakes and encountering challenges. When makers get frustrated that something doesn't work out as planned, remind them that feeling frustrated is normal and that professional game designers, engineers, and artists experience similar challenges every day. Encourage makers to discuss with their partner and classmates, and redesign as necessary. Remind makers of the cycle of "design, make, discuss." Explain that it's a cycle because often when creating something new, makers need to redesign after learning from things that don't work out the first time.

Managing technology, electronics, and making: Part of the excitement of this project is the combination of using computers, code, electronics, and hands-on making materials together. Remind makers to clean up their work areas as they go. It may be helpful to circulate around the room to remind makers to establish work zones and help clear cardboard and recycling around groups as they work.

Collaboration: Ensure that all makers participate in all aspects of the project (coding, designing, and making). Often makers will stick with an area they're comfortable in. While acknowledging their particular skill in one area, use pair programming as needed to switch roles, or have makers switch roles at least once in the session. Encourage makers to stretch out of their comfort zone to gain more experience with aspects they're less comfortable with.

ADDITIONAL RESOURCES

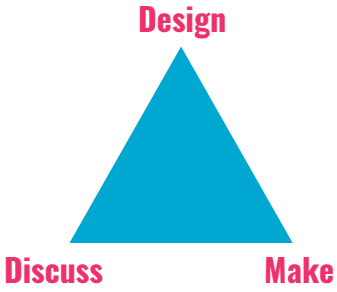
Micro:bit Game Design with Conditionals

Wonderful Idea Co: Computational Carnival

CYBER ARCADE WORK SESSIONS

STEP 1 (5 MINUTES)

Reflect on the last session



Makers reflect on their journal entry from the last session with their partner. Ask them to revisit what they answered last week, and discuss the changes and goals they want to set for the day.

EXPLAIN

This week you and your partner continue working on your projects as **game designers** and **game engineers**. Next week, we'll be planning and setting up our classroom as a Cyber Arcade. Then we'll invite people from the community to come and play the games we're creating!

Throughout these 9 weeks, we've been asking ourselves, "How do artists, engineers, and makers solve problems when they're working?"

When we invent and create, it's normal to feel challenged and frustrated! Professional artists, engineers, and makers feel exactly how you do every day.

At the end of the last session, we reflected on:

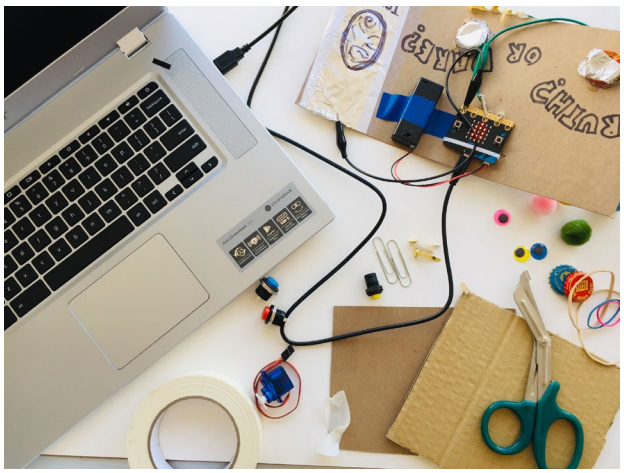
1. Challenges you faced and worked through
2. Changes you and your partner will make to your original plan
3. What you plan to start with today

Take 3 minutes to get out your **project plan** and talk with your partner about what each of you will work on and any changes you plan to make today.

Ask for volunteers to share with the group their responses from their notebooks, pausing to acknowledge and make suggestions for some of the comments they share.

STEP 2 (45 MINUTES)

Time to make.



This block of time is for makers to work on their projects. Make sure every group has enough space to safely work on coding and making.

- Circulate through the room and facilitate safe working habits by celebrating safe behaviors and calling attention to any unsafe behaviors (see Facilitation Tips).
- Help groups troubleshoot only after they've tried on their own for some time—and after they've asked classmates.
- Give suggestions to groups on how to improve their engineering, coding, etc., on an organic and 1:1 basis.
- Help students with making cuts on their cardboard if you're comfortable using the box cutter (see Facilitation Tips).
- If makers get frustrated or express conflict, point out when they should stop making to discuss and redesign as needed.

STEP 3 3 MINUTES

Reflect and discuss.

Before cleaning up for the day, makers pause to reflect. Ask makers to reflect and share about various points of the design, make, discuss cycle. Call out specific examples of partner groups that you observed working together successfully through problems in both making and programming.

1. **Make:** Were there any challenges or successes you can share about your making and programming today?
2. **Discuss:** How did you and your partner communicate today? Share examples of how you worked together to get through a tough problem.
3. **Design/Redesign:** How did executing your design work out? Did you make any changes to your original design?

Note: Remind makers to save their files onto their USB flash drive before cleaning up.

STEP 4 7 MINUTES

Clean up.

Makers will:

- Disconnect the battery pack.
- Put supplies and technology in their assigned bins.
- Return laptops and plug them in for charging.
- Return tools and materials that can be used again to the right place.
- Clear tables of garbage and recycling.

SAFETY AGREEMENT

1. Take care when walking with scissors or sharp things (hold with point facing down).
2. One maker at a time per tool prevents accidents.
3. Be mindful of space from others when using tools.

GLUE GUN SAFETY

1. Only 1 or 2 makers at the hot glue station at a time.
2. Don't touch the tip of the glue gun.
3. Don't point the glue gun at another person.
4. Work at the protected glue gun station.
5. Keep the glue gun close to your work.
6. If the glue gun jams, ask an adult for support.

TROUBLESHOOTING TIPS

The cardboard is difficult to cut.

- Using the inside of the scissors instead of the tip can make cutting easier.
- Cutting pieces away from the edge of the cardboard is easier than cutting out a shape from the middle of the cardboard.
- If you're really having a hard time, ask a classmate or adult to help you with cuts.

The hot glue isn't holding stuff in place.

- Hot glue dries quickly, so try to apply the glue a little at a time, instead of large amounts.
- After gluing, hold pieces in place for at least 20 seconds before releasing.
- Support two pieces with an L-bracket or bridge, using glue or tape.

The cardboard won't hold the shape.

- Try experimenting with a different joining technique.
- Try different thicknesses of cardboard or layers of cardboard.

The board isn't showing what we coded.

File version check

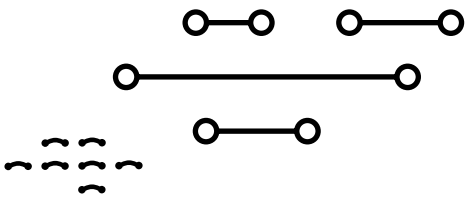
- Check to see that you uploaded the most recent copy of the code.
- Resave the latest version and drag and drop it onto the Micro:bit.

Our code isn't doing what we expected.

Check for bugs

- Read through the code.
- Read it out to a friend.
- Check to see if there are extra blocks that aren't supposed to be there.

The LED on the Micro:bit isn't flashing when we click Upload.	**Bad cable or port** • If the Micro:bit isn't showing up in the computer menu, try a different cable. • Try a different USB port on the laptop.
Our code isn't uploading correctly to the board. The board feels hotter than usual.	**Burnt board** • Try pressing the reset button on the board. • Try uploading to a new Micro:bit board.
Our board isn't turning on when connected to the battery pack.	**Battery** • Check the batteries to see if they're charged. • Check to see if the batteries are flipped.
We have alligator clips connected to the board, but the code isn't running.	**Alligator clips** • Make sure alligator clips are secure on the correct pins and are touching the metallic parts. • Try switching alligator clips.



TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

WEEK 10 CYBER ARCADE & EVENT PLANNING

INTRODUCTION

This is the last session makers have to continue working on their interactive Cyber Arcade projects. They'll prepare their projects to share in a community Cyber Arcade event at the end of the day. Inviting members of the community to come participate is a great motivator and creates a genuine experience for makers to share their work.

ESSENTIAL QUESTIONS

- What is a game or interactive project we want to share with our community?
- What changes do we need to make to our project to be ready for the Cyber Arcade?
- How do artists, engineers, and makers solve problems when they're working?

LEARNING OUTCOMES

1. Engage in project-based learning through problem-solving and troubleshooting by creating a game using a Micro:bit microcontroller and code.
2. Build, test, and complete interactive projects for a community event.

VOCABULARY

Project planning: Process of clarifying goals and listing the steps and materials required to complete a project

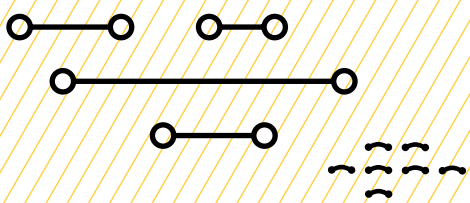
Game mechanics: Basic actions, processes, visuals, and control mechanisms that are used to make a game

Game designer: Person responsible for designing game storylines, plots, objectives, scenarios, the degree of difficulty, and character development

Game engineer: Specialized software engineers who design and program video games

Pseudocode: Detailed, informal description of what a computer program must do

Troubleshooting: Using resources to solve issues as they arise



MATERIALS LIST

EACH PAIR OF MAKERS NEEDS:

- Micro:bit microcontroller
- External battery pack
- AAA batteries (2)
- Laptop with internet connection
- USB to micro-USB cord
- USB flash drive
- Notebook
- Markers
- Colored pencils
- Scissors
- Cardboard scissors

ALL MAKERS NEED ACCESS TO:

- Alligator clips
- Alligator-to-pin wires
- Buttons (momentary and locking)
- Servos
- Tape (masking, painters, duct)
- Aluminum foil
- Assorted cardboard
- Assorted paper
- Pipe cleaners, pom-poms, popsicle sticks
- Misc bottle caps/recycling (optional)
- Hot glue gun and glue sticks (see Facilitation Tips)

Items can be portioned out per table or set up in an area where students can access them freely.

TEACHER PREP WORK

1. Set up an equitable system for student access of materials (either as a materials area or distributed evenly per table).
2. Set up a hot glue station covered with newspaper or butcher paper.
3. Print and cut copies of the **Project Checklist** (1 slip for each group).
4. Print out copies of the **Cyber Arcade Project Description** (1 for each group, plus extras).
5. Print the **Troubleshooting Tips** and post in the classroom.

FACILITATION TIPS

Inspiring creativity: To build excitement for the Cyber Arcade, invite external community members (students, staff, parents, etc.) if possible. Having an authentic audience makes the experience of sharing their hard work more fun and significant. Makers may be at a point of wanting to decorate their projects or create props to make the experience more exciting and fun. If possible, help organize and acquire additional materials when appropriate.

Safety: Using hands-on tools is an empowering part of this curriculum. However, practicing safety when working is crucial when using hot glue and sharp tools. You know your makers best, so make adjustments and adaptations as necessary.

If makers misuse any tools, have them take a break from the tool and return at your discretion.

Managing technology, electronics, and making: Part of the excitement of this project is the combination of using computers, code, electronics, and hands-on making materials together. Remind makers to clean up their work areas as they go. It may be helpful to circulate around the room to remind makers to establish work zones and help clear cardboard and recycling around groups as they work.

Collaboration: Ensure that all makers participate in all aspects of the project (coding, designing, and making). Often makers will stick with an area they're comfortable in. While acknowledging their particular skill in one area, use pair programming as needed to switch roles or have makers switch roles at least once in the session. Encourage makers to stretch out of their comfort zone to gain more experience with the other areas they're less comfortable in.

ADDITIONAL RESOURCES

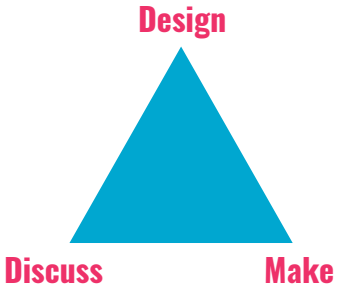
Micro:bit Game Design with Conditionals

Wonderful Idea Co.: Computational Carnival

CYBER ARCADE & EVENT PLANNING

STEP 1 (5 MINUTES)

Complete a Project Checklist.



Makers reflect on where they left off in the last session and discuss what they need to do today in order to be ready for the Cyber Arcade event.

EXPLAIN

This is the last session you and your partner have to complete your projects before we present and share our work in the interactive Cyber Arcade event. At the end of the day we will set up and host our Cyber Arcade and experience each other's projects.

Depending on how far along your project is, you may have to simplify or add to your project to meet the deadline. You can also work on making your project more presentable by decorating or creating signs and props. You and your partner will go through a project checklist to make sure you have completed all the steps to be ready to present. If you still have work to do on the list, make sure you and your partner work together to complete it before the event begins.

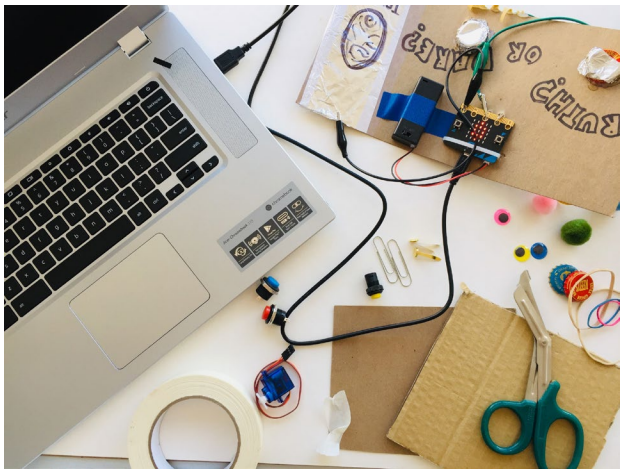
Give each group a printed Project Checklist slip they can fill out together and tape into their notebook.

PROJECT CHECKLIST

- ☐ Coding is complete, tested, and saved onto the USB flash drive.
- ☐ Code is successfully uploaded to the Micro:bit.
- ☐ Test connections to the Micro:bit (servos, buttons, switches, batteries, speakers) and secure with tape, pipe cleaners, or zip ties where necessary.
- ☐ Cardboard construction is complete and sturdy.
- ☐ Surface decorations, signs, and props are complete.

STEP 2 (30 MINUTES)

Time to make.



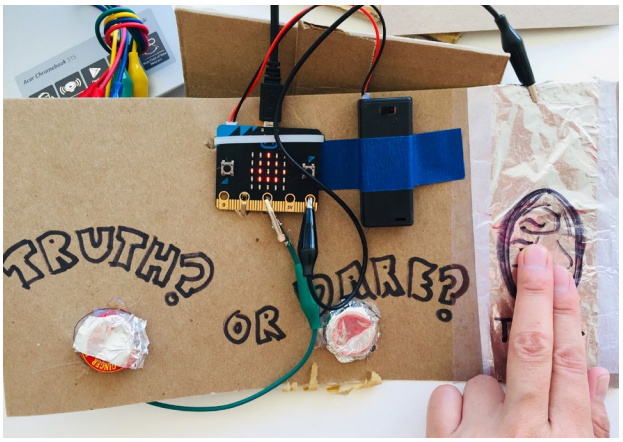
This block of time is for makers to complete any work on their projects. Depending on how far along makers are, they may need to simplify their project in order to finish, or they may be done and want to add more surface decorations. Groups that are finished can help other groups or make signage or decorations for the event, using the maker supplies.

Make sure every group has enough space to safely work on coding and making.

- Circulate around the room and facilitate safe working habits by celebrating safe behaviors and calling attention to any unsafe behaviors (see Facilitation Tips).
- Assist with troubleshooting issues related to coding, electronics, and building with cardboard only after makers have tried on their own for some time, and after they've asked classmates.
- Give suggestions on what parts to edit out or prioritize in order to have a project to present in the next session.
- Help students with making cuts on their cardboard if you're comfortable with using the box cutter (see Facilitation Tips).
- If makers get frustrated or express conflict, point out when they should stop making to discuss and redesign as needed.

STEP 3 10 MINUTES

Set up the Cyber Arcade.



This is the time for makers to make any last-minute changes and set up their projects before the event starts. If there is time, have makers fill out the **Cyber Arcade Project Description** and put it next to their project.

Set up the room in a way that safely allows the flow of visitors and transforms the space to feel exciting. Makers who are finished and ready can help decorate the space with streamers, balloons, etc. Give makers specific tasks or roles for the event.

EXPLAIN

You have 10 minutes to make any final touches and set up your projects for the arcade! Don't worry if your project is not as far along as you had hoped. We have worked so hard and explored so many high-level skills! If you're finished with your project, you can

help set up the space with decorations or practice how you'll interact with the visitors.

STEP 4 10 MINUTES

Run the Cyber Arcade.

Makers present and share their work with community members. Depending on how much time is available, this may be done in group presentation fashion or science fair style, where people freely circulate interacting with the projects.

When makers present, have them share the following:

1. Explain what their project is and how it works.
2. Share the challenges they worked through together in their coding and making.

STEP 5 5 MINUTES

Clean up.

Makers will:

- Disconnect the battery pack.
- Put supplies and technology in their assigned bins.
- Return laptops and plug them in for charging.
- Return tools and materials that can be used again to the right place.
- Clear tables of garbage and recycling.

SAFETY AGREEMENT

1. Take care when walking with scissors or sharp things (hold with point facing down).
2. One maker at a time per tool prevents accidents.
3. Be mindful of space from others when using tools.

GLUE GUN SAFETY

1. Only 1 or 2 makers at the hot glue station at a time.
2. Don't touch the tip of the glue gun.
3. Don't point the glue gun at another person.
4. Work at the protected glue gun station.
5. Keep the glue gun close to your work.
6. If the glue gun jams, ask an adult for support.

TROUBLESHOOTING TIPS

The board isn't showing what we coded.	**File version check** • Check to see that you uploaded the most recent copy of the code. • Resave the latest version and drag and drop onto the Micro:bit.
The code isn't doing what we expected.	**Check for bugs** • Read through the code. • Read it out to a friend. • Check to see if there are extra blocks that aren't supposed to be there.
The LED on the Micro:bit isn't flashing when we click Upload.	**Bad cable or port** • If the Micro:bit isn't showing up in the computer menu, try a different cable. • Try a different USB port on the laptop.
Our code isn't uploading correctly to the board. The board feels hotter than usual.	**Burnt board** • Try pressing the reset button on the board. • Try uploading to a new Micro:bit board.
Our board isn't turning on when connected to the battery pack.	**Battery** • Check the batteries to see if they're charged. • Check to see if the batteries are flipped.

TROUBLESHOOTING TIPS

Print and use the empty rows to fill in with other problems and solutions that can be shared.

PROJECT CHECKLIST

Print out (4 per page), cut out, and give one slip to each group.

PROJECT CHECKLIST

- ☐ Coding is complete, tested, and saved onto the USB flash drive.
- ☐ Code is successfully uploaded to the Micro:bit.
- ☐ Test connections to the Micro:bit (servos, buttons, switches, batteries, speakers) and secure with tape, pipe cleaners, or zip ties where necessary.
- ☐ Cardboard construction is complete and sturdy.
- ☐ Surface decorations, signs, and props are complete.

PROJECT CHECKLIST

- ☐ Coding is complete, tested, and saved onto the USB flash drive.
- ☐ Code is successfully uploaded to the Micro:bit.
- ☐ Test connections to the Micro:bit (servos, buttons, switches, batteries, speakers) and secure with tape, pipe cleaners, or zip ties where necessary.
- ☐ Cardboard construction is complete and sturdy.
- ☐ Surface decorations, signs, and props are complete.

PROJECT CHECKLIST

- ☐ Coding is complete, tested, and saved onto the USB flash drive.
- ☐ Code is successfully uploaded to the Micro:bit.
- ☐ Test connections to the Micro:bit (servos, buttons, switches, batteries, speakers) and secure with tape, pipe cleaners, or zip ties where necessary.
- ☐ Cardboard construction is complete and sturdy.
- ☐ Surface decorations, signs, and props are complete.

PROJECT CHECKLIST

- ☐ Coding is complete, tested, and saved onto the USB flash drive.
- ☐ Code is successfully uploaded to the Micro:bit.
- ☐ Test connections to the Micro:bit (servos, buttons, switches, batteries, speakers) and secure with tape, pipe cleaners, or zip ties where necessary.
- ☐ Cardboard construction is complete and sturdy.
- ☐ Surface decorations, signs, and props are complete.

CYBER ARCADE PROJECT DESCRIPTION

Welcome to our Cyber Arcade! Our game is called:

Here is how you play our game:

This is how we coded our game:

INPUTS	OUTPUTS

Here are some cool things we used in our game (switches, servos, etc.):

Thank you for coming to our Cyber Arcade! (Add makers' names below.)

_____ and _____