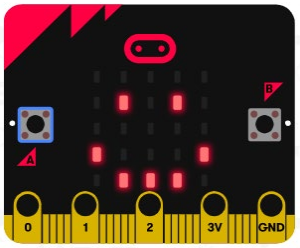
WEEK 3 DAY 1: DESIGNING GAMES WITH CONDITIONALS



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 and conditionals 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 make a game

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

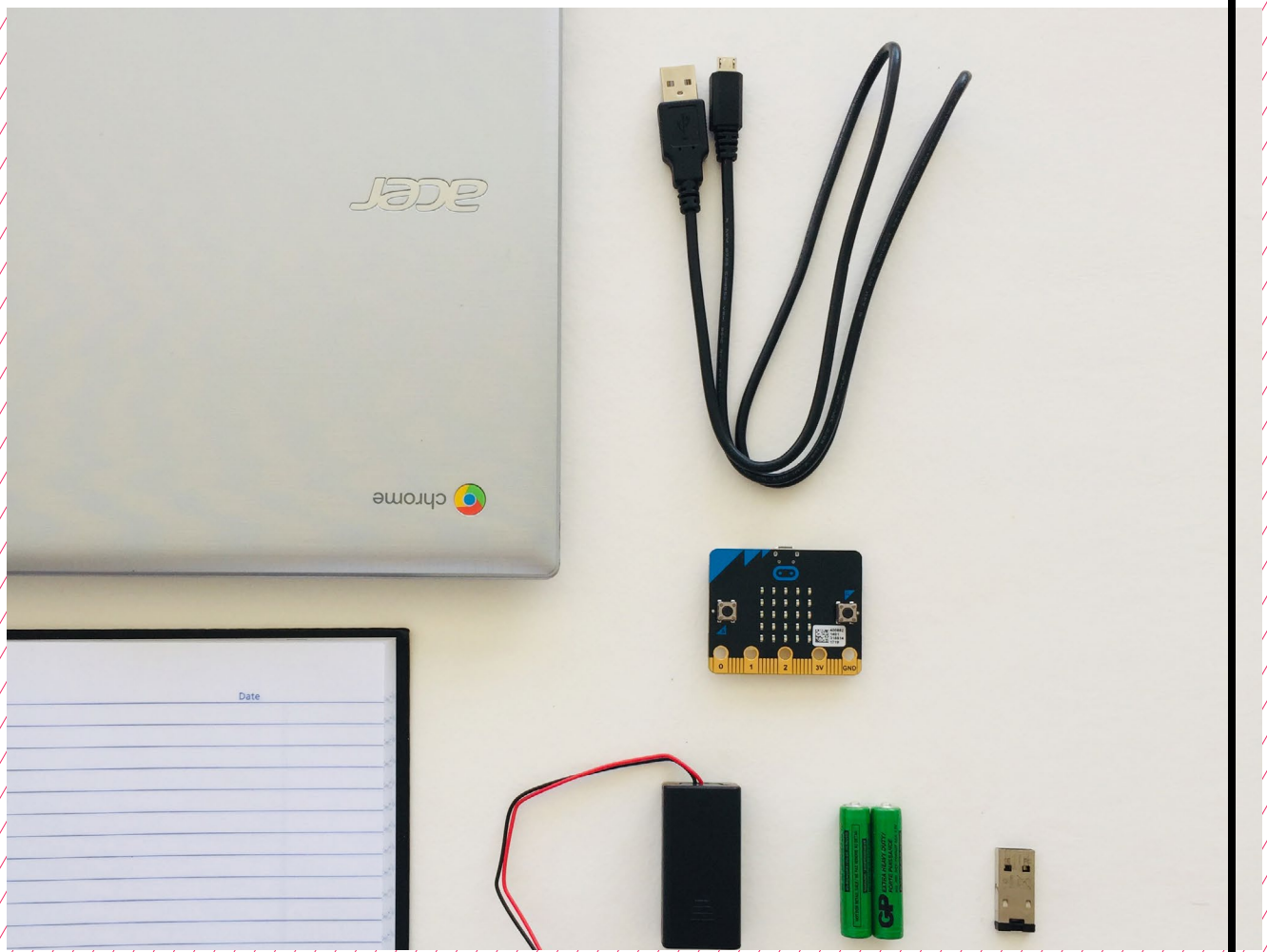
Game engineer: Specialized software engineers who design and program video games

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. 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

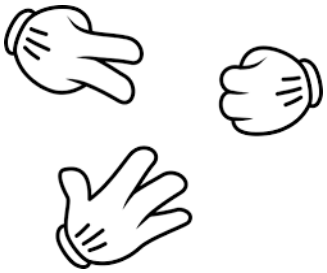
Micro:bit Game Design with Conditionals

I Love My Neighbor: Icebreaker Game with Conditionals

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

5 MINUTES

Learn about conditional statements in MakeCode.

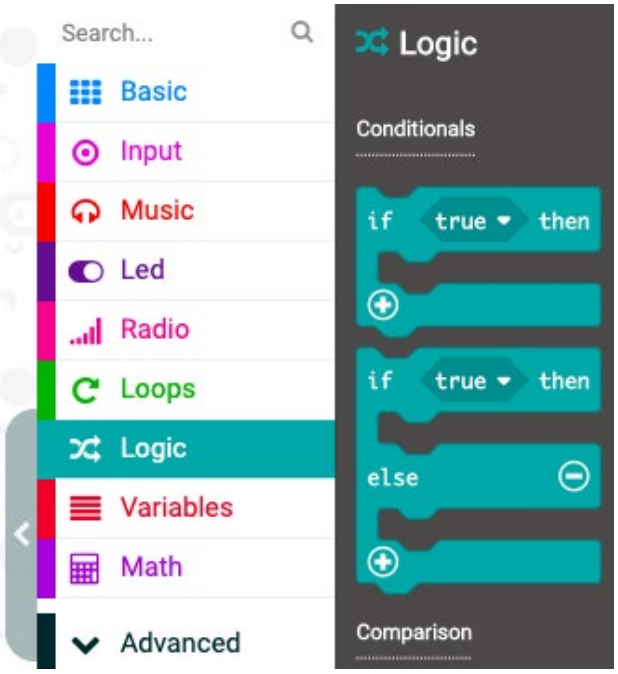
Show an example of a conditional statement in MakeCode with the Micro:bit simulator and talk through the logic.

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:

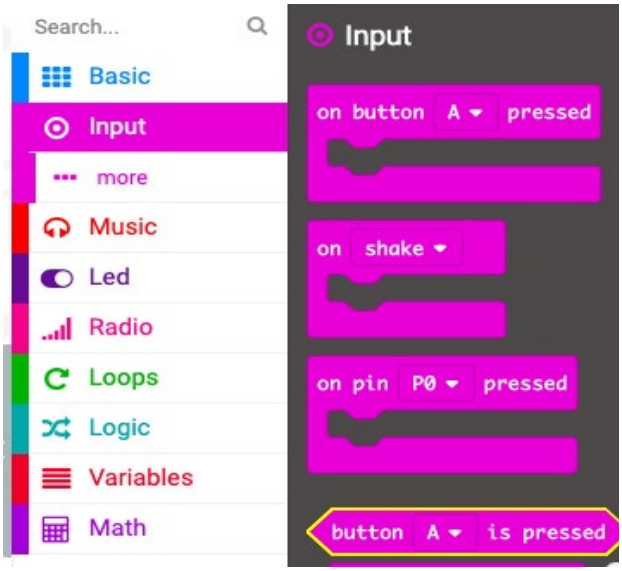
1. Click on the **Logic** menu of blocks. Click and drag over an **if_true_then** block to the coding space.



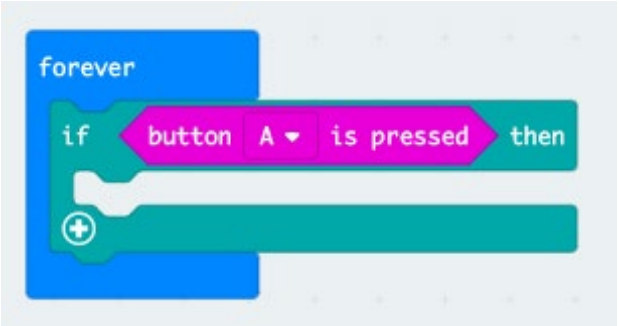
2. Notice the rectangular shape with the pointed ends that reads **true** inside of it. In the MakeCode software, you can drop any other block that

matches this shape into blocks with the same shape.

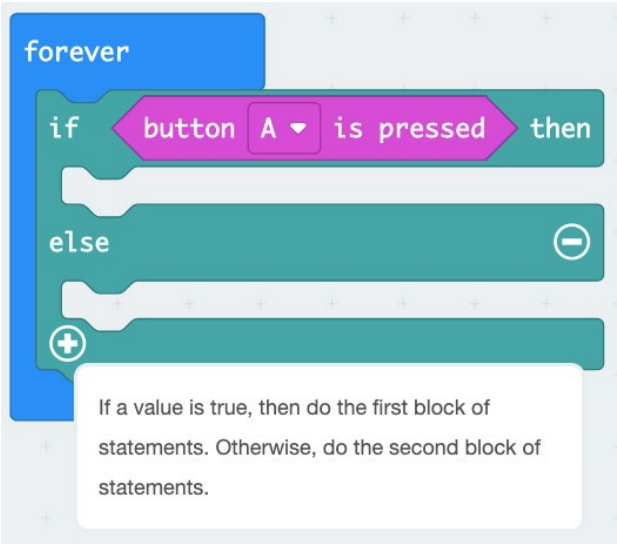
For example, click on the **Input** menu. Notice there's a block that matches the rectangle with the pointed ends (shown outlined in yellow).



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**.



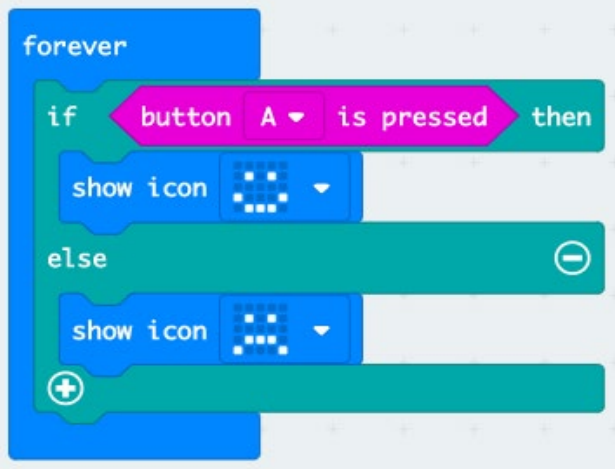
4. Next, click on the **+** in the **if_then** block. You'll see the green block grow with a space for an **else** statement.

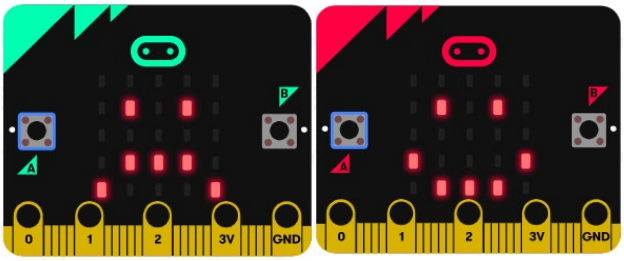


5. The example code shown could be the start of programming a digital pet game, similar to the Tamagotchi. For example, **button A** could be "feeding" the character, which then shows a smile on the LED display.

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

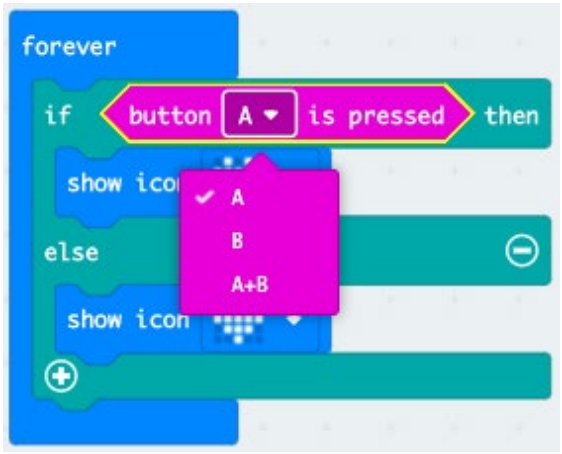
```
IF button A is pressed
THEN show smile
ELSE show frown
```

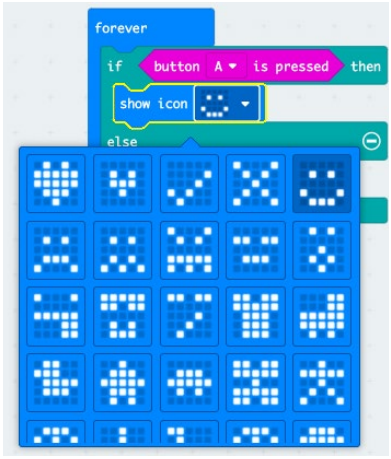




6. Remember, in order to save the code, we need to name and save the file, then connect the USB flash drive to move the file onto the USB flash drive.

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





STEP 4 20 MINUTES

Program conditional statements.

Makers will now pair program to code their own conditional statements in the MakeCode software. Makers will use the simulator to experiment within the **Logic**, **Basic**, and **Input** blocks. They shouldn't upload to the board until Step 5.

Makers will:

- Choose who will be the driver and the navigator (10 min each).
- Tinker with programming various conditional statements within the **Logic**, **Basic**, and **Input** blocks.

STEP 5 10 MINUTES

Download/upload to the board.

When makers are satisfied with the conditional statement codes they've created, it's time to save and upload their code onto the Micro:bit. (Review Steps 4 and 5 in Week 2, Day 1, for details on saving, downloading, and uploading to the Micro:bit.)

Makers will:

- Save their file with a unique name.
- Save their file to the USB flash drive.
- Download the code onto the laptop hard drive.
- Connect the Micro:bit to the laptop using the USB to micro-USB cord.
- Click and drag their file to the Micro:bit.
- Watch for the flashing yellow LED on the Micro:bit (flashes until code is completely uploaded).
- Carefully eject/disconnect the board.
- Connect the external battery pack.

STEP 6 10 MINUTES

Share and reflect.

Ask a few volunteers to:

1. Show their programmed Micro:bit.
2. Explain their code.

If possible, plug the maker's laptop into the projector display so that everyone can see their code as they explain it.

STEP 7 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.
- Clear tables of garbage and recycling.

WEEK 3 DAY 2: NUMBER RANGES IN GAME MECHANICS

INTRODUCTION

This week makers continue learning how to use the Micro:bit microcontroller, with the addition of coding Math blocks, to design interactive games.

ESSENTIAL QUESTIONS

- How can we use code and math to create an interactive game or experience?
- How do artists, engineers, and makers solve problems when they're working?

LEARNING OUTCOMES

1. Learn how to use code and simple math 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

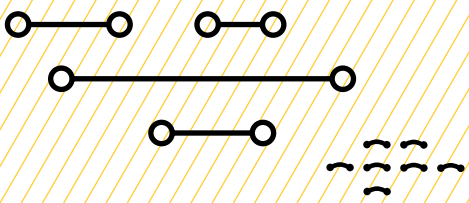
Game designer: Person responsible for designing game storylines, plots, objectives, scenarios, the degree of difficulty, and character development

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

FACILITATION TIPS

Student roles: Encourage students to switch roles between coding and making. Makers may initially feel more comfortable in one role or the other, but remind them that it's valuable that they expand beyond their comfort zone to gain experience with both making and coding.

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 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/ REFERENCES

Examples of Dice Games

How to Play Dice Games

What is an Accelerometer?

How an Accelerometer Works

NUMBER RANGES IN GAME MECHANICS

STEP 1 (5 MINUTES)

Introduce number ranges and 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."

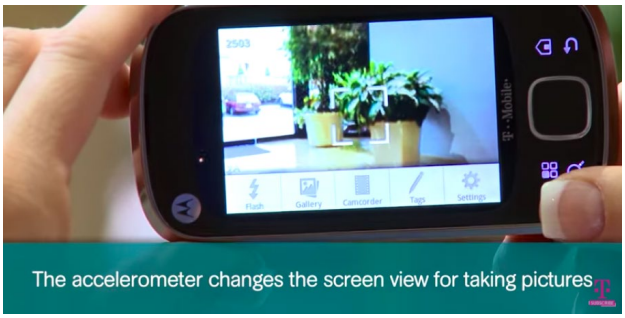
- Ask makers: "What are some games you've played that use dice?"

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.)



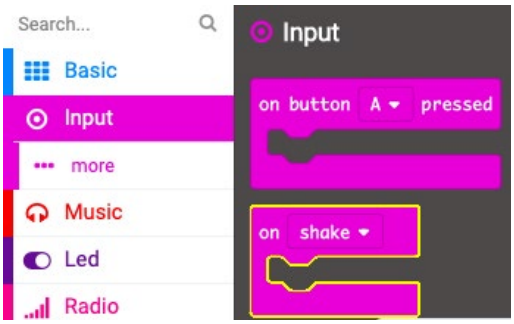
"**What is an Accelerometer?**" on YouTube, uploaded by T-Mobile, 7/29/2010

STEP 2 (10 MINUTES)

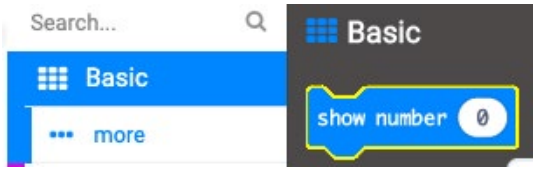
Code the Micro:bit dice.

DEMONSTRATE AND HAVE STUDENTS FOLLOW ALONG IN THE SOFTWARE:

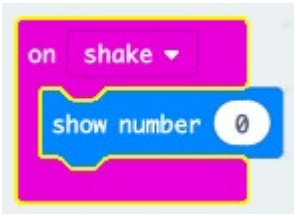
1. Go to the **Input** menu, and click and drag the **on shake** block over to the coding space.



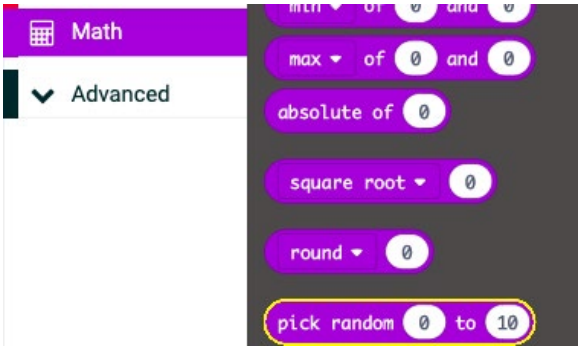
2. Next, in the **Basic** menu, drag over a **show number** block and put it in the first position.



3. Notice that in the **show number** block there's an oval space that reads "0". Just like when we programmed the conditional, we can put any oval-shaped blocks into this space.



4. In the **Math** blocks menu, drag over a **pick random** block and put it in the **show number** block.



5. Change the numbers to be "**pick random 1 to 6**".

6. Once we have this code, you can name and save your file to your USB flash drive, and then upload it to the Micro:bit to test the digital dice.

STEP 3

30 MINUTES

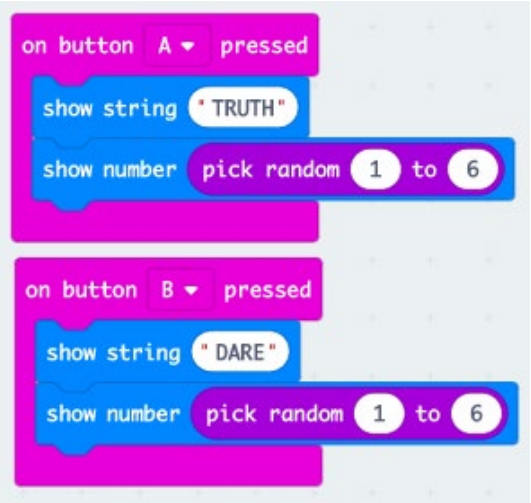
Makers become game designers and game engineers.

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 4

Document the games.

EXPLAIN

All board games come with a set of instructions explaining the rules or game mechanics. Once you and your partner have a game idea you plan to work on, in your notebooks, write down how your game works.

In your description, include:

1. Exactly how the code works (the **pseudocode**)
2. How the game works and how the Micro:bit is used in the game (the game mechanics)

Pseudocode: Explain step by step how the code works. (Example: When button A is pressed, show a smiling face, then choose a random number between 1–10.)
Game Mechanics: Explain the rules of your game. (Example: The number is how many spaces to move on the board.)

STEP 5

Share and reflect.

With any remaining time, have makers explain and play each other's games. This can be done in pairs or presentation style.

STEP 6

Clean up.

Makers will:

- Disconnect the battery pack.
- Put any materials they want to keep to use in assigned 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 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.