Crystal Cove® CONSERVANCY

Session 6: Learning to Code

Focus: Introduction to Coding
Grade Level: *7*
Session Length: Two sessions of 45-60 minutes

Driving Questions

- How can we use block-base coding to write instructions for a micro:bit?

NGSS Links

- Planning and Carrying Out Investigations
- Designing Solutions

Systems Thinking Characteristics

- Identifying Hidden Dimensions of the System

In the sixth session of Project Crystal Code, students are introduced to block-based coding as they learn the key coding concepts that they will need to code a soil moisture sensor..

Students learn more about the micro:bit computer and are introduced to the Microsoft MakeCode website for writing code. Students use the Make:Code website to practice using different coding skills: using blockcode to write instructions, using inputs and outputs, creating conditional statements, and defining variables. Optionally, students can also practice downloading code onto a real micro:bit to test their script directly.

In an optional extension, students write a short program to play Rock-Paper-Scissors with a micro:bit.

Learning Outcomes & Assessments

By the end of this module, students will be able to...	***You can assess this using...***
1. ***Use*** blockcode to write short programs for a micro:bit computer.	Field notebook entry; Student reflections
2. ***Explain*** the concepts of inputs, outputs, conditional statements, and variables.	Field notebook entry; Whole class discussion

Session Overview

Section	Description	Length	Format
Launch	Students meet Erick, another Crystal Cove Conservancy educator, who gives them an overview of the structure of Session 6.	5 minutes	Whole class
Explore	Students are introduced to four coding concepts. They practice using each coding concept with a real life example in their field notebook, watch a screencast to see how to use blockcode to apply the concept, and then use the MakeCode website to write and modify code. Optionally, if students have access to a micro:bit, they can also download the code and try it on a real micro:bit.	70-100 minutes	Individual or Research teams
Share	Students discuss the key concepts that they have used today.	5-10 minutes	Whole class
Reflect	Students reflect on their experience during Session 6 and their next steps.	5 minutes	Individual
Extend	In this optional extension, students apply what they've learned to write a program to play Rock-Paper-Scissors with their micro:bit.	30-45 minutes	Individual or Research teams

Overview of the Soil Moisture Sensor Project

In Sessions 5-7 of Project Crystal Code, students will be tasked with designing, coding, and building their own environmental sensors that can measure the amount of water in the soil in our mulch treatments. Although this part of the program is optional, it offers an opportunity for participants to practice computational thinking and coding skills while developing a deeper understanding of the environmental system.

Why We Do This

One component of systems thinking is being able to identify and understand invisible parts of the system. By building and coding their own soil moisture sensor from scratch, students will develop a better understanding of what they're measuring -- and thus how invisible parts of the system work -- than if they simply used a provided soil moisture meter.

This project also offers the opportunity for students to be introduced to block-based coding in the context of solving a real-world environmental problem. Often, coding and computational thinking is taught in isolation from other scientific disciplines. It is our hope that by integrating coding into an environmental science program, we will be able to both help students see how computer science skills have become an important part of scientific work, while also inspiring interest in coding amongst students who might not normally choose to take a traditional computer science class.

Where Today's Session Fits In

During Session 6, students are introduced to block-based coding as they practice using the four main coding concepts that they'll need to write a program for a soil moisture sensor in Session 7.

The goal of Session 6 is to help students develop a conceptual understanding of these four coding concepts. Later, in Session 7, they'll apply each of these concepts to plan and write a program that will allow the micro:bit to measure the amount of moisture in the soil.

Throughout all three sessions, it is important to remind students of the ultimate goal so that the sessions don't feel disjointed from the rest of the Project Crystal Code program. By learning how to build and code a soil moisture sensor, we will be able to measure how much moisture is in the soil, allowing us to test our hypothesis about how mulch affects soil moisture.

Session 5		Session 6	Session 7	
Exploring circuits, conductivity, and resistance	*Designing a plan to physically build a soil moisture sensor*	*Practice block-based coding skills*	*Writing a block-based program to turn a micro:bit computer into a soil moisture sensor*	*Assembling the soil moisture sensor, downloading the code, and testing it (optional)*

You are here!

Introduction to Block-Based Coding

In contrast to text-based programming, block-based coding involves dragging "blocks" of code to construct a program. It is popular in educational settings because it allows students to practice and experiment with conceptual ideas instead of worrying about syntax or typos in a coding language.

If you'd like to learn more about block-based coding in educational settings, ***this article*** has a good explanation!

The four concepts that students will use during Session 6 include:

- **Using blockcode to write instructions:** Computers need very specific instructions to tell them what to do. This section introduces students to the idea that they can drag blockcode to write instructions for a virtual micro:bit.
- **Using inputs to produce outputs:** This section introduces students to the idea of inputs and outputs. The micro:bit can take in information from a variety of inputs, including two buttons, a temperature sensor, and its pins. Readings from these inputs can then be displayed as outputs on the micro:bit's screen.
- **Creating conditional statements:** In computer science, conditional statements tell a computer what to do if a condition is true or false. They usually follow an *If/Then* structure: *If* [a condition is true], *then* [the computer will do this]. This section builds on students' understanding of inputs and outputs by layering on conditional statements: *if* the micro:bit receives a specific input (such as a button being pushed), *then* it will create a specific output (such as displaying a temperature reading on the screen).
- **Defining variables:** Just like in math, computer science uses variables as a shortcut to represent and recall a longer bit of information. This section introduces students to defining a variable so that it represents a longer bit of code so that they don't need to write out the longer code each time.

Virtual Materials

- Session 6 Google Slides Presentation: ***http://bit.ly/2MFW0Y2***
- Session 6 Download Code to a Micro:bit Extension Google Slides Presentation: ***http://bit.ly/3jy3ZSE***
- Session 6 Rock Paper Scissors Extension Google Slides Presentation: ***http://bit.ly/3p6A1X3***
- Session 6 Coding Instructions: ***http://bit.ly/3rypVQp***
- Session 6 Field Notebook Template (optional): ***http://bit.ly/3q4zJRQ***
- MakeCode Website: ***http://bit.ly/3a1sRiH***

Each student will need...

- A device with internet access (a computer, smartphone, or tablet will all work!)
- Field notebook and pencil
- Session 6 Coding Instructions
- Access to a micro:bit and a micro USB cable to download the code (optional)

Before You Start Teaching

- Review the format of the session and the four different coding concepts that students will be learning: (1) Using blockcode to write instructions (***Slides 5-8***), (2) Using inputs to produce outputs (***Slides 11-14***), (3) Creating conditional statements (***Slides 15-19***), and (4) Defining variables (***Slides 20-23***). Note that each mini-section follows the same format:
    - In the first slide, the conceptual idea is introduced to students in a short video.
    - In the second slide, students practice using the concept during a short exercise in their field notebook.
    - In the third slide, a screencast demonstrates how to put together that concept using blockcode.
    - In the fourth slide, students are given step-by-step instructions to write their own code on the Microsoft MakeCode website. This step also includes one additional challenge, where students must apply the concept and figure out how to make a modification to the code on their own.
- Decide how you want students to work on the Microsoft MakeCode website. Students can work independently or in research teams. Note that if students work virtually in teams, one student will need to share their screen with the others and be in charge of manipulating the platform.
- Review the structure of the session and decide whether you will break it up over multiple class meetings. There is also an optional extension at the end where students can practice applying the coding concepts that they've learned to write a short program to play Rock-Paper-Scissors.
- Although it is not necessary to have access to a real micro:bit to practice coding, decide whether you want to acquire micro:bits so that students can download their code. If you do want to have them practice downloading their code and testing it in the real world, ***Slides 9-10*** give specific instructions on how to do this.
- Copy over the Session 6 Slideshow for your chosen platform to your own Google Drive account. Test to make sure that the videos work. (If not, you may have to check the permissions on the Crystal Cove Conservancy Youtube Account.)
- Decide how you will assess student progress during this lesson. You can ask students to take a screenshot of each coding step and paste their code into their field notebook (or another virtual document). Alternatively, if you do not want to see each screenshot, you can monitor students as they work during the session or have them reflect in their field notebook at the end of the process to check for understanding.

Learning Sequence

Launch

Introduction to Block-Based Coding (5 minutes)

1. Open the ***Session 6 Slideshow*** and play the video on ***Slide 2*** for your class. In this video, Khai will briefly introduce Session 6 and the task for the day, which is to learn how to write instructions for a micro:bit computer using block code. Eventually, we will be able to code it to take readings from a soil moisture meter and display a soil moisture amount.

2. After watching the video, move on to ***Slide 3***, which gives an overview of what students will do and learn during Session 6.

3. Then, advance to ***Slide 4*** to meet Erick, another Crystal Cove Conservancy educator who will guide them through learning to code the micro:bit.

Erick introduces students to the structure that will be used throughout the session. First, he will introduce one of four new coding concepts in a video. Next, students will practice using that concept in their field notebooks. Afterwards, they'll watch a screencast where Erick will show the students how to use the Micro:bit MakeCode website to create that particular code. Then, finally, they'll get a chance to write the code on their own.

Explore

Concept #1: Using Blockcode to Write Instructions (30-35 minutes)

1. Move on to ***Slide 5*** and play the video of Erick. He introduces the idea that coding is writing specific instructions for a computer to follow. Those instructions have to be very specific and exact, because the computer doesn't have a mind of its own. It will do only what you tell it to do, no more, no less.

2. Advance to ***Slide 6*** and give the students a few minutes to practice writing clear instructions by having them write instructions for making a sandwich in their field notebooks.

If there is time, have students share their instructions with another student or with the whole class. You can go through each step exactly and see if you could make a successful sandwich, or if any key steps are left out that get you stuck (e.g.,if there are no instructions to lift the knife back off the bread, you end up with a knife in your sandwich!).

3. Then, move on to ***Slide 7*** where Erick will demonstrate how to use the Make:Code website to write instructions for the micro:bit to light up with a happy face.

4. Afterwards, advance to ***Slide 8*** to give the students time to try the task on their own. You can have students work independently or in their research teams. If they're working virtually in teams, one student will need to screenshare the MakeCode website with the others.

Once they have displayed a happy face, as a challenge, students can try to manipulate the blockcode to make the micro:bit display their names instead!

Concept #2: Using Inputs & Outputs

5. Next, advance to ***Slide 9*** where Erick will introduce the concept of inputs and outputs. An input is information that is put into the computer. It could be a button being pressed, sensor readings about the world around it, or something else. An output is whatever the computer produces as a result of the input.

6. Move on to ***Slide 10*** and let students practice using inputs and outputs as they write down which ingredients (inputs) are needed to produce certain types of sandwiches (outputs).

7. On ***Slide 11***, Erick will use the concept of inputs and outputs to demonstrate how to write code that tells the computer that when Button A is pressed (the input), the micro:bit should display a happy face on the screen (the output).

8. Advance to ***Slide 12*** to give students a minute to try this on their own. As a challenge, students can use the same skill to make the display show a different icon when button B is pressed as an alternate input.

Concept# 3: Using Conditional Statements

9. Move on to ***Slide 13*** and play the video where Erick introduces the concept of conditional statements. These are used in coding to tell a computer what to do if a certain condition is true or false.

10. Advance to ***Slide 14*** and let students complete a few conditional statements on their own. (Alternatively, you can come up with a few together as a class.)
Some examples include:

- *If* it is raining outside, *then* you bring an umbrella to school.
- *If* peanut butter is used in the sandwich, *then* jelly should be used too.
- *If* it is hot, *then* you have ice cream.
- *If* you want to say something in class, *then* you raise your hand.

11. Next, play the video on ***Slide 15***. Erick will demonstrate how to use the temperature sensor as an input to display an image showing how hot or cold it is, and then create conditional statements in blockcode so that *if* Button A is pushed, *then* the micro:bit display shows the numerical temperature.

One note here: Although the micro:bit platform refers to the temperature image as a "bar graph," the image displayed is *not* a typical bar graph. Instead, it is a graphical representation of the temperature. (I.e., if it is 100 degrees Celsius, the screen will be entirely black; if it is 50 degrees, the screen will be half black.) You will likely want to call this out to students directly so that they are not confused by the use of the term bar graph.

12. Advance through ***Slides 16-18*** and give students time to try coding the micro:bit to display the temperature themselves.
As an extra challenge, students can write a new conditional statement so that when

Button B is pushed, *if* the temperature is over 20 degrees Celsius, *then* the micro:bit display shows a happy face; *If else* (i.e., if Button B is pushed and the temperature is under 20 degrees Celsius), the display shows a sad face. This challenge will require making a conditional statement within a conditional statement, as shown in the step-by-step coding directions.

Concept #4: Using Variables

13. Finally, advance to ***Slide 19*** where Erick will introduce the last coding concept that students will need: variables. Essentially, a variable is like a code name that helps a computer remember information.

14. Advance to ***Slide 20*** to practice thinking of real life examples for defining variables. You can have students work through the examples in their field notebooks on their own or come up with examples as a whole class.

Information You Want to Remember	*Variable (What you or a computer uses to remember)*
Example: A meal deal with a hamburger, fries, and a soda on the menu at a restaurant	***Example:*** The number next to the meal deal on the menu that you can use to order
Someone's phone number	Their name in your contact list
Your favorite website address or YouTube channel	A bookmark on your web browser
Your favorite outfits in *Animal Crossing*	A magic wand in *Animal Crossing*

15. Next, advance to ***Slide 21*** where Erick will demonstrate how to set up a variable that converts the temperature in Celsius to Fahrenheit.

16. Advance through ***Slides 22-24*** and give the students time to practice setting a variable to convert Celsius to Fahrenheit on their own. As a challenge, they can put a few of their new coding skills together to see if they can insert their new Fahrenheit variable into their previous code so that if Button A is pressed, then the temperature is displayed in Fahrenheit.

Share

Sharing Our Thoughts (5-10 minutes)

1. Bring the group back together as a class. Open ***Slide 25*****,** and give students a few minutes to respond to the first question and share their impressions of coding. What new concepts did they learn? Was it fun? What was challenging for them?

2. Once students have had a chance to reflect on their experiences, ask them to respond to the next set of questions:

- What is an input? An output? Can you give an example of an input and output on the micro:bit computer?
- What is a conditional statement? Can you give an example of a conditional statement in real life? One that you wrote in blockcode today?
- What is a variable? Can you give an example of a variable in real life? How are variables useful when we're writing code?

3. Remind students again that they are learning these skills in order to build a soil moisture sensor, which will be used to measure the amount of moisture in the soil at our research site. In the next session, they'll apply the four coding concepts that they learned today in order to write a program to tell a micro:bit computer to measure the amount of moisture in the soil.

Reflect

Reflecting on Session 6 (5 minutes)

1. At the end of the discussion, advance to ***Slide 26*** in the slideshow and play the video, where Khai will invite them to spend a few minutes reflecting.

2. Move on to the final slide, which will share reflection questions. Ask students to spend five minutes reflecting on their experiences today in their field notebook.

3. Finally, thank the class for their time today. Tell them that when you gather again, they will be able to put everything together that they have learned and code the micro:bit to read and display soil moisture measurements!

Extend

Optional Extensions: Downloading and Testing Your Code on a Micro:Bit (5-10 minutes)

1. Open ***Slide 2*** of the ***Session 6 Extension: Download Code to a Micro:Bit Slideshow*** and watch Erick give a screencast demonstration of how to reset the micro:bit to make sure there isn't any previous code left behind. This is always a good step to go through at the beginning of every class period in order to make sure students are starting with a blank slate, not another student's previous code.

Take a minute with your class and have them start by clearing their micro:bits.

2. Next, on ***Slide 3***, Erick will demonstrate how to download their code onto the physical micro:bit. Give students a few minutes to try downloading their code onto a micro:bit, and test to see if it works!

3. If micro:bits aren't ejected properly, they can often get stuck in a maintenance mode. ***Slide 4*** gives directions for how to reset the mode if students run into this issue.

Rock Paper Scissors (30-45 minutes)

1. If you want your students to have more practice writing code and an extra challenge to put their skills to the test, you can have them try building a code to play rock paper scissors with their micro:bits! Open the Extension Slideshow to ***Slide 2***, where Erick will introduce the task.

2. Next, advance to ***Slide 3*** and share the link with directions to code the micro:bit with your students. Give the students 15-20 minutes to try building the code on their own. Have the students test their code by teaming up with another student or another research team, and see if they can play rock paper scissors with each other!

3. For a bonus challenge, you can move on to ***Slide 4*** and challenge your students to alter their code to display their hypothesis for how mulch will affect soil moisture. ***Slide 5*** has tips to help get students started.

They will need to create a new variable called "mulch treatment" just like they did for "hand," and set the 3 conditional options to read out the mulch type (woody, straw-like or none), and an image of the moisture level they predicted (very little, medium, or a lot).