

<b>MISSION 2: Introducing CodeX</b>	<b>Time: 30-60 minutes</b>
<p><b>Overview:</b></p> <p>These first missions are all about getting to know the <b>Codespace</b> user interface and the <b>CodeX</b> hardware. Before students finish, they will plug in the hardware and write some code to make it do something!</p> <p>In this early stage, the most important guidance is to <b>carefully read</b> the instructions. The answer to most problems is right there on the screen!</p>	<p><b>Cross Curricular:</b></p> <ul style="list-style-type: none"> <li>• Supports <b>language arts</b> through reflection writing</li> </ul>
<p><b>Materials Included in the learning portal <a href="#">Teacher Resources</a>:</b></p> <p><b>Mission 2 Sliddeck</b></p> <p>The slide deck is for teacher-led instructions that let you guide students through the material using the slides. It is an alternative to the students reading a lot of instructions in CodeSpace. The slides mirror the instructions, with simplified language that is chunked into smaller sections at a time. The information is shown on slides with “Objective”. The tasks to complete are on slides with “Mission Activity”. The slides also tell the students when to write something in their Mission Log.</p> <p><b>Mission 2 Workbook</b></p> <p>The workbook can be used instead of slides for student-led or independent work. It is an alternative to students reading a lot of instructions in CodeSpace. It mirrors the instructions (and the slide deck), with simplified language that is chunked into smaller sections at a time. Each objective is on its own page. The tasks to complete are labeled “DO THIS” and have a robot icon next to it. The workbook also tells the student when to write something in their Mission Log.</p> <p><b>Mission 2 Log</b></p> <p>This mission log is the worksheet for students to complete as they work through the mission. It includes warm-up questions (called pre-mission preparation) and wrap-up questions (post-mission reflection) as well as questions along the way. It should be printed and given to each student before the mission starts. They write on the mission log during the assignment and turn it in at the completion of the mission (assignment).</p> <p><b>Mission 2 Lesson Plan</b></p> <p>The lesson plan comes from the CodeX Teacher Manual and is included here for easy reference.</p>	
<p><b><a href="#">Teacher Resources</a></b></p> <p><b>Mission 2 Solution (Heart1)</b></p> <p>A basic final code solution to Mission 2 in a text file. You can copy and paste the code in CodeSpace and run it.</p>	<p><b>Formative Assessment Ideas:</b></p> <ul style="list-style-type: none"> <li>• Exit ticket</li> <li>• Mission log completion</li> </ul>
<p><b>Vocabulary:</b></p> <ul style="list-style-type: none"> <li>• <b>CPU:</b> Central Processing Unit or the brain of the computer</li> <li>• <b>Peripheral:</b> A device that interacts with the CPU (common peripherals are LED lights, display screen, buttons, mouse, keyboard, and printer)</li> </ul>	

## Preparing for the lesson:

Students do not need the Codex until halfway through the lesson.

- Look through the slide deck and workbook. Decide what materials you want to use for presenting the lesson. The slide deck can be projected on a large screen. The workbook (if used) can be printed or remain digital through your LMS.
- Be familiar with the mission log (assignment) and the questions they will answer.
- Print the Mission Log for each student.
- If you have a word wall, or another form of vocabulary presentation, prepare the new terms.
- Have the CodeXs ready for students, with a USB cable for each CodeX. You may want students to work together in pair programming, or they can work individually.

## Lesson Tips and Tricks:

### Teaching tip:

You can use a variety of discussion strategies to get the most engagement from your students. For example, you can have students write their answers before asking anyone for an answer. You can use one of many think-pair-share methods. You can have students write their answer and share with someone, and then have other students share answers they heard from their peers. You can randomly select students to answer.

### Mission 2 Discussion: slide 2, page 1

The lesson starts with a short intro into why students should learn programming. You can go over the slides or present it in your own unique way. Then hand out the Mission Log to each student and have the pre-mission discussion. Students will not need the CodeX until near the end of the mission.

### Pre-Mission Discussion: slides 3-4, page 1

Students can write in their log first and then share, or discuss first and then write in their log.

- What are some things you can attach to a computer or laptop?
  - Introduce peripherals -- answers could include keyboard, mouse, monitor, printer, camera, speaker, headphones, etc.
- Why do you think you should learn to program a computer? (answers will vary)

### Mission Activities:

Most of this lesson is on the computer, learning about the CodeX. Students do not need the CodeX until Objective 5.

- Each student will complete a Mission Log.
- Students could work in pairs through the lesson, or can work individually.


### Teaching tip: Objective #1 -- Slides 5-6, page 2


Students will click on one of three words and write about it in their log. Recommendation: students should click on all three words and choose one to write about.

### Teaching tip: Objective #2 -- Slides 7-8, page 3

This objective uses the 3D simulator. Students will need to rotate the Codex to see the back for the final lightning bolt.



 **Teaching tip: Quiz** -- Slide 9, page 4

Students take a  short quiz about keeping the CodeX safe. Quiz questions below. Quiz questions below. You can decide if you need to go over the question with your students.

 **Teaching tip: Objective #3** -- Slides 11-13, page 4

Students will learn about the CPU and write about it in the log. Then they will close the instruction panel and use the 3D simulator to find the CPU. It is on the back of the CodeX.

 **Teaching tip: Objective #4** -- Slides 14-15, page 5

This objective discusses connecting the CodeX to the computer, but students do not do the connection until the next objective. They use the 3D simulator to find the connection.

 **Teaching tip: Objective #5** -- Slides 16-18, page 6


Give students the CodeX and USB cable. They will connect the CodeX to the computer in this objective. When the CodeX is connected, a window will pop-up. It can be closed. It is not needed for anything; it just indicates that the CodeX is connected.

The second part of this objective is to connect the CodeX to CodeSpace. Students need to follow the step-by-step instructions. Usually they only need to do this once and not everytime they connect the CodeX. The key here is to look at the message at the bottom of the screen, left corner. It will indicate if the USB CodeX or simulator CodeX is connected. To run code, the USB CodeX needs to be connected. The objective handles this. When running code, make sure the objective is not looking for the simulator. If so, move to a different objective.


 **Teaching tip: Objective #6** -- Slides 19-20, page 7

Students create a file for their first program. A simple rule for filenames is discussed -- no spaces in the name, and it should be descriptive.

 **Teaching tip: Objective #7** -- Slides 21-23, page 8

The CodeTrek is introduced. It is found at the left bottom corner of the instruction panel. It will always give tips and hints for the code. Sometimes students need to type exactly what they see in CodeTrek. Sometimes it will give hints or #TODOs for the students. Usually CodeTrek is optional and is there as a support to students. Not mentioned, but also important -- there is an icon next  to CodeTrek. It gives hints and can also be used by students as a support.

 **Teaching tip: Quiz** -- Slide 24, page 9

Students take a second  short quiz about the CPU and CodeTrek. Quiz questions below. You can decide if you need to go over the question with your students.

 **Teaching tip: Objective #8** -- Slide 25-26, page 9

Students will type their first program. They use CodeTrek and copy the code. Punctuation is stressed. Python is case-sensitive, so lower-case and upper-case spelling makes a difference. Also, indenting is important in Python.

 **Teaching tip: Objective #9** -- Slide 27-29, page 10

Students change their code to display a different image. Students must use the MUSIC image to check off the objective, but then they can change the image again to any of the built-in images.

## **Mission Complete:**

This mission ends with a program. All missions will now end with completed code. You need to decide how you will use the program for assessment. You could:

- Go to each student and check-off their code
- Have the students download their code to a text file and turn it in using your LMS



- Have students print their code (either download and then print the text file, or print a screenshot)
- Have students switch computers and run each other's code. Fill out a simple rubric and turn in to teacher
- Any other way that works for you

### **Post-Mission Reflection:** Slide 30, page 11

The post-mission reflection asks students to think about the many ways a CodeX could be used. You can change the question if there is something else you want to emphasize with your students.

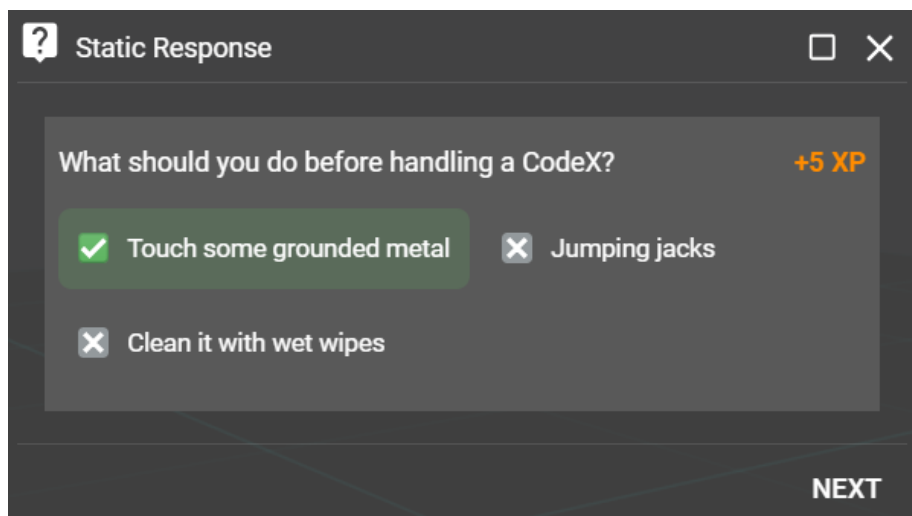
- What projects can you imagine using the CodeX for?

End by collecting the Mission Log and any formative assessment you want to include.




### **SUCCESS CRITERIA:**

- Identify major parts of the CodeSpace interface: Mission Bar, Objective Panel, text editor, CodeTrek, Toolbox, and Lesson Navigation Controls
- Successfully connect and disconnect the CodeX using the USB-C cable.
- Identify major parts of the CodeX: USB connector, LCD Grid, CPU
- Write a program, run it, and save it to the CodeX

### **Quiz #1 Questions**



### **Quiz #2 Questions**

 Questions TODO  

What is the CPU's job on the CodeX? +5 XP

Execute your code  Figure out what you were thinking

Provide +5 Volt power

Which of the following is an instruction in a code comment that you need to replace? +5 XP

`# x should be a float`  `# TODO: fix this`

`# good code below`

NEXT