**Lesson 6** Add and Subtract with keyPressed()

**How can we use key presses to control a sketch?**

| **Overview** | |
| --- | --- |
| This lesson introduces students to the keyPressed() function so that they can control a sketch using the keyboard. Students will also use the assignment operators **+=** and **-=** to change shapes. | |
| **Lesson Objectives** | |
| Students will be able to:   * Use keyboard events to control a p5 sketch * Use conditionals to refine user interaction * Use assignment operators to increase and decrease variables | |
| **Suggested Duration** | |
| Two periods (90 minutes) | |
| **Blueprint Foundations Student Outcomes (**https://blueprint.cs4all.nyc/outcomes/) | |
| Abstraction  Analyze | **Describe** how I might use patterns to express an idea. |
| Abstraction  Prototype | **Explain characteristics** or patterns that informed a function or an interface I created. |
| Algorithms  Prototype | **Demonstrate** the benefit of using an event, conditional or loop in my prototype. |
| Algorithms  Communicate | **Compare and contrast** how conditionals or loops were used in classmates’ prototypes. |
| Programming  Communicate | **Present** the challenges, and benefits of using tools to program collaboratively. |
| **Vocabulary** | |
| * **Event**: An action detected by a computer program, like clicking a mouse or pressing a key. * **+= Operator**: Adds some value to a variable. * **-= Operator**: Subtracts some value from a variable. | |
| **Planning Notes** | |
| * In the student activity, they will make a [color battle game](https://editor.p5js.org/mparker/sketches/iSjur3AMm). You may choose to have them work in groups of 3-4 (as with the traffic light activity in [Lesson 4](https://docs.google.com/document/d/1pd-SJZ5_WjUOZ6JAajiMzAahBASvNMVQ0xbdfkOmDbM/edit?usp=sharing)) or in pairs.   + If working in groups, you may provide **chart paper** for their pseudocode. * For pair programming, decide on pairs in advance, preassign first roles, and make sure students switch roles every 5 - 10 minutes. | |
| **Resources** | |
| * Interactivity with p5.js: Section on [Key Events](https://p5js.org/learn/interactivity.html) | |
| **Assessments** | |
| * Assess the **Student Activity**. Check for the ability to:   + Use keyboard events to control a p5 sketch   + Use conditionals to refine user interaction   + Use += and -= to increase and decrease variables * Assess the **Wrap Up**. Check for the ability to:   + Describe debugging strategies | |

| **Do Now:** |
| --- |
| * **[Design Journal]** List different ways that people are able to **interact** with a website or smartphone application. * Have students share their responses (e.g., typing, clicking/moving the mouse, tapping a touch screen, giving voice commands, etc.) |
| **Discussion: Events** |
| * Tell students that in programming, user actions (like typing, clicking, etc.) are called **events**. In this lesson, they will use a new p5 function that detects keyboard events, called keyPressed(). |
| **Teacher Demo: keyPressed()** |
| * Code along: With your students, build out the following example that changes the fill of an ellipse when a key is pressed (see completed sketch [here](https://editor.p5js.org/mparker/sketches/6Ng2WZt_U)).   + Note: Be clear that the keyPressed() function should be written **after** the draw loop, not inside it.      * Note: Before the sketch will respond to key presses, students will have to click on the canvas (so p5 knows that you’re trying to control the sketch and not type into the editor).   + Ask students what they notice. * Students may point out that once the size increases to 200 pixels, the ellipse is stuck at that size. Tell students that you’re going to use **new syntax** so that the ellipse continues to increase its size every time a key is pressed:      * Explain that the **+=** operator **adds** some value to a variable whenever it is used. In this case, the value is 2. Whenever a key is pressed, the ellipse’s size will increase by 2. * Try out values like 10 or 25 to make the ellipse grow at a faster rate. To keep track of how big the ellipse is, use the text function **inside draw** to display the current size on the canvas:      * Now tell students to replace **+=** with **-=** (using the “minus” sign next to the zero on the keyboard) and run the code again.   + Ask what they notice about the program.      * Explain that the **-=** operator **subtracts** some value from a variable whenever it is used. * Tell students right now, any key that is pressed will decrease the size. In order to use a **specific** key to decrease the size, they need to use a conditional:      * Note: If your students are unfamiliar with **==**, explain that this is used to test if two values are **equal** to each other. Remind students that they must use a **double** equals sign inside conditionals.   + Now the only key that will decrease the size variable is “a” * Independent Exercise: Ask students to write a second conditional that **increases** the size variable whenever the “b” key is pressed.   + Call on a student to share their solution. The code should be similar to this:      * Note: There are many different keyboard events in the p5 reference (see below). To avoid confusion, tell students to use the keyPressed() function in their sketches, **not** keyIsPressed (which is not a function, but rather a built-in p5 variable). |
| **Student Activity: Color Battle Game** |
| * In this student activity, students will work in groups or pairs to create their first p5 game, in which each player presses a key to fill the canvas with their color:      * Before students begin coding, they should develop a plan using pseudocode either on **chart paper** or in their **design journals**.   Instructions   1. Make a color battle game by using ONE rectangle and ONE variable to control the rectangle. 2. Use keyPressed() to control the game with TWO different keyboard keys. 3. Use conditionals to **increase** (+=) and **decrease** (-=) the variable when different keys are pressed. 4. CHALLENGE #1 - RESTART THE GAME    1. Add a third conditional inside keyPressed() that uses a new key to restart the game by setting the variable to its original value. 5. CHALLENGE #2 - ANNOUNCE THE WINNER    1. Add two new conditional statements inside **draw()** that check to see if a color has won. HINT: Use the **==** operator.    2. Use the [text() function](https://p5js.org/reference/#/p5/text) to write a message announcing the winner and which key to hit to restart the game.      1. CHALLENGE #3 - DISABLE THE KEYS    1. Update the first two conditionals inside keyPressed() so that they check to see if either color has “won” by filling up the canvas. HINT: use the **&&** operator.    2. The keys should stop working (the variable can’t increase or decrease) once a color wins and until the game restarts.  * Note: If students are completely stuck or struggling unproductively, tell them to create one “color” by making a rectangle the height of the canvas. The other “color” is the background. The rectangle should grow and shrink by changing its **width** when different keys are pressed. For additional help, have them use this [starter code](https://editor.p5js.org/mparker/sketches/_clrvRgMF). * [Solution](https://editor.p5js.org/mparker/sketches/iSjur3AMm) for teacher reference. |
| **Wrap Up** |
| Share Out:   * [**Design Journal**] First have students reflect on the following:   + What was the most challenging part of making this?   + What strategies did you use to get unstuck?   + How might you change or add onto this project in the future? * Invite a few pairs to come up and share their process:   + Describe your project and how it works.   + Share your responses to the reflection questions. * Make sure students share their code with each other and with you before they leave. |
| **Extension: N/A** |
|  |