# Interaction

## `mouseX` and `mouseY`

The `mouseX` and `mouseY` variables always store the current x and y coordinates of the mouse relative to the origin of the canvas. So if the mouse was currently at the x position of 150 pixels and the y position of 200 pixels, the value of the `mouseX` variable would be 150 and the value of the `mouseY` variable would be 200.

```
function draw() {
  // The ellipse's x and y positions follow
the mouse
  ellipse(mouseX, mouseY, 100, 100);
}
```

## The `mouseIsPressed` Variable

`mouseIsPressed` is a built-in boolean variable that is `true` when the mouse button is pressed, and `false` when it is not pressed. The `mouseIsPressed` variable is commonly used in `if` statements to perform actions based on whether the mouse button has been pressed or not.

```
// Draws ellipse if mouse is pressed
if (mouseIsPressed) {
    ellipse(200, 200, 180,180);
}
```

## The `mousePressed()` Function

The `mousePressed()` function is called once after each mouse press, meaning that the code block within the function will not loop if the mouse is held down.
In the above code example, the `mousePressed()` function is used to set a new random value for the `grayValue` variable once after each mouse press.

```
let grayValue = 0;

function draw() {
  background(grayValue);
}

// Generate a random number between 0 and 255
for the grayValue variable once after each
mouse press
function mousePressed() {
  grayValue = random(255);
}
```

## The `dist()` Function

The `dist()` function returns the distance between two points given four arguments: x and y coordinates for two endpoints. This function is versatile and is often used to calculate the distance between stationary and moving points in a p5.js sketch.
The code example above illustrates how the `dist()` function can be used to calculate the distance between a static point on the canvas at (10, 50) and the mouse position.

```
// Calculates the distance between a point at
(10, 50) and another at (mouseX, mouseY)
let distance = dist(10, 50, mouseX, mouseY);
```

## The key Variable

The `key` variable stores the value of the most recently pressed key. It is most commonly used to check if a specific alphanumeric key has been pressed.

In the code example above, the `key` variable is used to check if the `'a'` key is pressed. If the `'a'` key is pressed, the ellipse's fill color is set to `'red'`. If other keys are pressed, the fill color is set to `'green'`.

```
function draw() {
  if (key === 'a') {
    // If the 'a' key is presed, set fill
color to 'red'
    fill('red');
  } else {
    // If a key other than the 'a' key is
pressed, set fill color to 'green'
    fill('green');
  }

  ellipse(width / 2, height / 2, 100, 100);
}
```

## The keyCode Variable

`keyCode` is a built-in variable that can be used to detect if a special key, such as `BACKSPACE`, `RETURN`, and `RIGHT_ARROW`, has been pressed. The `keyCode` variable returns the decimal ASCII value of the most recently pressed key.

It is commonly used as the condition of `if` statements to check if a specific key has been pressed. The code example above checks if a key with an ASCII value of 32, the spacebar key, has been pressed. If the spacebar is pressed, the ellipse is drawn on the canvas. If another key is pressed, no shape is drawn on the canvas.

```
function draw() {
  background(255);
  fill(255, 0, 255);

  // Check if the keyCode of the most
recently pressed key is 32 (spacebar)
  if (keyCode === 32) {
    // If spacebar is pressed, draw ellipse
        ellipse(width / 2, height / 2, 100,
100);
    }
}
```

## Key Events

Key events are registered through keyboard event functions such as `keyPressed()`, `keyReleased()`, `keyTyped()`, and `keyIsDown()`. Each of these functions runs when a specific type of key interaction occurs. The key event functions can be combined with the `key` and `keyCode` variables to perform actions when specific keys have been pressed.

The above code example uses the `keyReleased()` function to trigger its code block when a key has been released. In this case, the code block checks if the `'m'` key has been released to set new random values for the `posX` and `posY` variables used to draw the rectangle.

```
let posX = 100;
let posY = 75;

function draw() {
  rect(posX, posY, 100, 75);
}

//Each time the 'm' key is released, randomly set the position of the rectangle
function keyReleased() {
  if (key === 'm') {
    posX = random(400);
    posY = random(400);
  }
}
```

## The keyTyped() Function

The `keyTyped()` function is called every time a key is pressed but ignores special keys like Backspace, Delete, Ctrl, and Shift. Any non-special keys typed will trigger the code block within the `keyTyped()` function. The `keyTyped()` function is triggered once per key press.

In the code example above, the `keyTyped()` function is used to set random position and size for the `square()` function. Whenever a non-special key is pressed, a new square of random size will appear somewhere on the canvas.

```
let posX, posY, size;

function draw() {
  square(posX, posY, size);
}

// Runs once whenever a non-special key is pressed
function keyTyped() {
  // Randomly set position and size for the square
  posX = random(width);
  posY = random(height);
  size = random(200);
}
```

## The `keyPressed()` Function

The `keyPressed()` function runs the code block within its function each time any key is pressed. The function will only run once per key press regardless of how long the key is pressed down.

The `keyPressed()` function is used in the code above to decrement the green value of the ellipse's fill color. The `greenVal` variable decrements by 10 once every time a key is pressed. When `greenVal` becomes less than 0, it is reset to 255.

```
let greenVal = 255;

function draw() {
  fill(120, greenVal, 100);
  ellipse(width / 2, height / 2, 200, 200);
}

// Each time a key is pressed, the green
value of ellipse's fill color decreases by 10
// When greenVal reaches below 0, it resets
to 255
function keyPressed() {
  if (greenVal >= 0) {
    greenVal -= 10;
  } else {
    greenVal = 255;
  }
}
```