JavaScript Documentation

Printing to Console

```
println(str);
print(str);

// Example:
println("Hello world.");

// Printing without a new line:
print("Hello world. ");
print("How are you?");
```

Variables

```
// Declare a variable
var myVarName;

// Initialize a variable
var myVarName = 5;

// Assign to an existing variable
myVarName = 10;

// Print a variable
println(myVarName);
println("The value is: " + myValue);
```

Functions

```
// Functions can take in values, called parameters.
// The function below takes in a parameter called
// 'input' and prints it.
function printText(input) {
    println(input);
}

// Functions can also return a value.
// The function below takes in a value,
// adds two to it, and returns it.
function addTwo(number) {
    return number + 2;
}
```

User Input

```
// Read a string
var str = readLine(prompt);

// Read an integer
var num = readInt(prompt);

// Read a float
var cost = readFloat(prompt);

// Read a boolean
var bool = readBoolean(prompt);

// You should replace the word prompt with
// the question you'd like to ask. For example:
var name = readLine("What is your name? ");
var age = readInt("What is your age? ");
var finishedWork = readBoolean("Is your work done? ");
```

Comparison Operators

```
// Comparison operators return booleans (true/false values)

x == y      // is x equal to y
x != y      // is x not equal to y
x > y       // is x greater than y
x >= y      // is x greater than or equal to y
x < y       // is x less than y
x <= y      // is x less than or equal to y

// Comparison operators in if statements
if(x == y){
    println("x and y are equal");
}

if(x > 5){
    println("x is greater than 5.");
}
```

Math

```
// Operators:
+   Addition
-   Subtraction
*   Multiplication
/   Division
%   Modulus (Remainder)
()  Parentheses (For order of operations)

// Examples
var z = x + y;
var w = x * y;

// Increment (add one)
x++

// Decrement (subtract one)
x--

// Shortcuts
x = x + y;      x += y;
x = x - y;      x -= y;
x = x * y;      x *= y;
x = x / y;      x /= y;

// Absolute value
var abs = Math.abs(x);

// Square root
var sqrt = Math.sqrt(x);

// Rounding
// Math.round() can be used to round numbers
var pi = 3.14;
var roundedPi = Math.round(pi);
println(roundedPi);             // prints out: 3

var goldenRatio = 1.618;
var roundedGoldenRatio = Math.round(goldenRatio);
println(roundedGoldenRatio);        // prints out: 2

// Floor Division
// Math.floor() can be used to perform floor
// division. With floor division, only the
// integer portion of the quotient is returned.

// For example, 5/2 is 2.5, but with floor division,
// the result is 2 and the .5 is discarded.
var result = Math.floor(5/2);
println(result);                    // prints out: 2
```

Random Numbers

```
// Random integer between low and high
Randomizer.nextInt(low, high);
Randomizer.nextBoolean();
Randomizer.nextFloat(low, high);
Randomizer.nextColor();

var roll = Randomizer.nextInt(1,6);

var color = Randomizer.nextColor();
```

Graphics

Canvas

```
// returns the width of the canvas
getWidth();

// returns the height of the canvas
getHeight();

// Examples
var height = getHeight();

// Returns the x coordinate of the center
// of the canvas
var center = getWidth() / 2;

// Removes all objects from the canvas
removeAll();
```

All Graphics Objects

```
// To get the type of the object
var type = obj.getType();

/*
could return:
'Circle'
'Rectangle'
'Text'
'Line'
*/
```

Circles

```
// To make a circle
var circle = new Circle(radius);

// To set the location of the center of the circle
circle.setPosition(x, y);

// Example, red circle with 50px radius with center at (100, 200)
var circle = new Circle(50);
circle.setPosition(100, 200);
circle.setColor(Color.red);

// Get the radius
circle.getRadius();                 // returns 50
var curRadius = circle.getRadius(); // store in variable

// Change the radius
circle.setRadius(100);

// Get the position of the center of the circle
var x = circle.getX();  // x is 100
var y = circle.getY();  // y is 200

// Change the location of the circle
var x = getWidth() / 2;
var y = getHeight() / 2;
circle.setPosition(x, y);   // circle center is in the center of the screen

// Adding to and removing from screen
add(circle); // Add to screen
remove(circle); // Remove from screen

// Move the circle dx horizontally and dy vertically
circle.move(dx, dy);
```

Rectangles

```
// To make a rectangle
var rect = new Rectangle(width, height);

// To set location of the upper left corner of rectangle
rect.setPosition(x, y);

// Example, 200x50 blue rectangle with upper left corner at (100, 200)
var rect = new Rectangle(200, 50);
rect.setPosition(100, 200);
rect.setColor(Color.blue);

// Get location of the upper left corner of the rectangle
var x = rect.getX(); // x is 100
var y = rect.getY(); // y is 200

// Change location of the rectangle
var x = getWidth() / 2;
var y = getHeight() / 2;
rect.setPosition(x, y)  // upper left corner is at center of screen

// Adding to and removing from screen
add(rect); // Add rectangle
remove(rect); // Remove rectangle

// Move the rect dx horizontally and dy vertically
rect.move(dx, dy);
```

Arcs

```
// To make an arc
var myArc = new Arc(radius, start, end, unit);

// More specifically, the parameters are:
//    1. radius of the arc
//    2. starting angle of the arc
//    3. ending angle of the arc
//    4. angle unit (0 for degrees, 1 for radians)

// To set the position of the center of the arc
myArc.setPosition(x, y);

// Example, a 90-degree arc with
// radius of 50 and color of red:
var myArc = new Arc(50, 0, 90, 0);
myArc.setPosition(100, 200);
myArc.setColor(Color.red);

// Get the location of the center of the arc
var x = myArc.getX();   // x is 100
var y = myArc.getY();   // y is 200

// Change the location of the center of the arc
var x = getWidth() / 2;
var y = getHeight() / 2;
myArc.setPosition(x, y);    // arc center is at center of screen

// Adding to and removing from screen
add(myArc); // Add arc
remove(myArc); // Remove arc
```

Lines

```
// To draw a line from (x1, y1) to (x2, y2)
var line = new Line(x1, y1, x2, y2);

// Set the line color to green
line.setColor(Color.green);

// Set the line width to 10 pixels
line.setLineWidth(10);

// Adding to and removing from screen
add(line);
remove(line);

// Move the line dx horizontally and dy vertically
line.move(dx, dy);

// Change the starting point of the line to (x1, y1)
line.setPosition(x1, y1);

// Change the end point of the line to (x2, y2)
line.setEndpoint(x2, y2);

//Get the starting point of the line
var x = line.getX();    // x has same value as x1
var y = line.getY();    // y has same value as y1
```

Ovals

```
// To make an Oval
var oval = new Oval(width, height);

// To set location of the center of the oval
oval.setPosition(x, y);

// Example, 200x50 blue oval with center at (100, 200)
var oval = new Oval(200, 50);
oval.setPosition(100, 200);
oval.setColor(Color.blue);

// Get location of the center of the oval
var x = oval.getX(); // x is 100
var y = oval.getY(); // y is 200

// Change location of the oval
var x = getWidth() / 2;
var y = getHeight() / 2;
oval.setPosition(x, y)  // oval's center is at center of screen

// Adding to and removing from screen
add(oval); // Add oval
remove(oval); // Remove oval

// Move the oval dx horizontally and dy vertically
oval.move(dx, dy);
```

Text

```
// To make a graphical text object
var txt = new Text(label, font);

// To set the position of the lower left corner of the text
txt.setPosition(x, y);

// Example
var txt = new Text("Hello, world!", "30pt Arial");
txt.setPosition(100, 200);
txt.setColor(Color.blue);

// Change what the text says
txt.setText("Goodbye!");

// Get the location of the lower left corner of text
var x = txt.getX(); // x is 100
var y = txt.getY(); // y is 200

// Change the location of the text
var x = getWidth() / 2;
var y = getHeight() / 2;
txt.setPosition(x, y)   // text's lower left corner is
                        // in the center of the screen

// Adding to and removing from screen
add(txt); // Add text
remove(txt); // Remove text

// Move the text dx horizontally and dy vertically
txt.move(dx, dy);
```

Images

```
// A web image can be added to the graphics canvas
// as a WebImage. WebImages are created, sized,
// and positioned much like other graphics objects.

// To create a new WebImage, use a URL that links
// directly to the image on the Internet (you will
// not be able to use an image that is stored
// only on your computer).

var copter = new WebImage("https://upload.wikimedia.org/" +
                          "wikipedia/commons/4/41/" +
                          "Bell_206L3_%28D-HASA%29.jpg");
copter.setSize(300, 150);
copter.setPosition(getWidth()/4, getHeight()/2);
add(copter); // Adding copter to screen
remove(copter); // Removing copter from screen

// Note that the URL to the image must be directly
// to the image file itself. It should end with
// something like .png, .jpg, or another image file type.
```

Images Hosted on CodeHS

- https://codehs.com/static/img/library/characters/penguin.png
- https://codehs.com/static/img/library/characters/monkey.jpg
- https://codehs.com/static/img/library/characters/leopard.jpg
- https://codehs.com/static/img/library/characters/chameleon.jpg
- https://codehs.com/static/img/library/characters/lizard.jpg
- https://codehs.com/static/img/library/characters/butterfly.jpg
- https://codehs.com/static/img/library/objects/icicle.png
- https://codehs.com/static/img/library/objects/helicopter.png
- https://codehs.com/static/img/library/objects/asteroid.png
- https://codehs.com/static/img/library/objects/soccerBall.png
- https://codehs.com/static/img/library/landscapes/flowers.jpg

Color

```
// There are many color constants. You can set an objects
// color like
obj.setColor(color);

// Specifically,
obj.setColor(Color.RED);

// List of available colors:
Color.RED
Color.GREEN
Color.BLUE
Color.YELLOW
Color.CYAN
Color.ORANGE
Color.WHITE
Color.BLACK
Color.GRAY
Color.PURPLE

// You can also make your own color by giving a red, green,
// and blue component like
var color = new Color(r, g, b);

// The values are between 0-255 for each component. After making
// a new color, you can use it to set the color of an object.

// For example, to set an existing rectangle called
// rect to be brown:
var brown = new Color(139, 69, 19);
rect.setColor(brown);

// Another way to set the color of an object is to use a
// string with the hexadecimal color value with setColor.
// For example, to set a rect object to be pink:
rect.setColor("#FF66CC");

// Other fun functions
// Return a random color
var color = Color.randomRed();
var color = Color.randomGreen();
var color = Color.randomBlue();

// Get a random color from the randomizer
var color = Randomizer.nextColor();
```

Rotation

```
/**
 * The following graphic objects can be rotated:
 * - Rectangle
 * - Arc
 * - Line
 * - Oval
 * - Text
 * - WebImage
 */

// Set the rotation of the rectangle with these parameters:
//    1. angle to rotate
//    2. angle unit (0 for degrees, 1 for radians)
//       This will default to degrees.
rect.setRotation(45, 0);           // Sets rotation of the rectangle to 45 degrees
rect.setRotation(45);              // Does the same thing.

rect.setRotation(Math.PI / 2, 1);  // Sets rotation of the rectangle to Math.PI/2 radians

// Add rotation with these parameters:
//    1. angle to rotate
//    2. angle unit (0 for degrees, 1 for radians)
//       This will default to degrees.
rect.rotate(45, 0);           // Rotates the rectangle by 45 degrees
rect.rotate(45);              // Does the same thing.

rect.rotate(Math.PI / 2, 1);  // Rotates the rectangle by Math.PI/2 radians
```

Booleans

```
// A boolean is either true or false
var myBoolean = true;

var anotherBoolean = false;

var result = readBoolean("Question? ");

// Not Operator
var x = !y;     // x gets the opposite of y

// And Operator
var andExp = x && y;

// Or Operator
var orExp = x || y;

// You can combine many booleans!
var boolExp = x && (y || z);
```

If Statements, If/Else, If/Else If/Else

```
if(BOOLEAN_EXPRESSION){
    // code to execute if true
}

if(BOOLEAN_EXPRESSION){
    // code if true
} else {
    // code if false
}

if(x < 0){
    println("x is negative.");
}

if(color == "red" || color == "blue" || color == "yellow"){
    println("Primary color.");
} else {
    println("Not a primary color.");
}

// You can use else if if you have multiple
// conditions, but only one should happen.
if(condition_1){

} else if(condition_2) {

} else if(condition_3) {

} else {

}


// You can always write these using nested
// if/else. For example:
if(condition_1){
    // code here runs if condition 1 is true
} else {
    if(condition_2){
        // if condition 2 is true
    } else {
        // and here if condition 2 is false
    }
}
```

For Loops

```
var COUNT = 5;

for(var i = 0; i < COUNT; i++){
    /* Repeat code betweeen the brackets 5 times,
     * as the COUNT variable is 5. */
}

// Print numbers 0-9
for(var i = 0; i < 10; i++){
    println(i);
}
```

While Loops

```
while(boolean expression){
    /* Repeat code betweeen brackets while
     * 'boolean expression' is true */
}

// Countdown from from 15 to 10
var i = 15;
while(i > 10){
    println(i);
    i--;
}
```

```
// This is a loop and a half format
while(true){
    // code
    if(condition){
        break;
    }
}
```

Timers

```
setTimer(fn, delay); // Create a timer
stopTimer(fn);       // Stop a timer

// Example: call moveBall every 40 milliseconds

function moveBall() {
    ball.move(x, y);
}

function start() {
    setTimer(moveBall, 40);
}
```

Mouse Events

```
// Mouse events are used to create programs
// that respond to users' mouse clicks, drags,
// and movements.

// When the mouse event occurs, the function
// registered with the event will be called. Note
// that you leave out the parentheses () when
// passing the name of the function.

// Here is a list of mouse events that can be used:
mouseMoveMethod(functionToCall);    // on mouse movement
mouseClickMethod(functionToCall);   // on mouse clicks
mouseDragMethod(functionToCall);    // on mouse drags
mouseDownMethod(functionToCall);    // mouse button depressed
mouseUpMethod(functionToCall);      // mouse button released

// Sample program using mouse events
function start() {
    // Set up mouse callbacks
    mouseMoveMethod(onMouseMove);
    mouseClickMethod(addBall);
    mouseDragMethod(updateLine);
}

function onMouseMove(e) {
    println("Mouse is at (" +
            e.getX() + ", " +
            e.getY() + ").");
}

function addBall(e) {
    var ball = new Circle(20);
    ball.setPosition(e.getX(), e.getY());
    add(ball);
}

function updateLine(e) {
    line.setEndpoint(e.getX(), e.getY());
}
```

Keyboard Events

```
function start() {
    // Set up keyboard callbacks
    keyDownMethod(keyDown);
    keyUpMethod(keyUp);
}

function keyDown(e) {
    if (e.keyCode == Keyboard.LEFT) {
        ball.move(-5, 0);
    }
    if(e.keyCode == Keyboard.letter('K')){
        println("You pressed K");
    }
}

function keyUp(e) {
    println("You lifted up a key");
}
```

Audio Files

```
// To add a sound file to a program, first create a variable
// to store the sound you want. Be sure to use a link
// directly to the audio file itself (for example,
// if it's an mp3, the link should end with .mp3).
// The link must be a full URL to a sound file that
// is available on the internet.
var mySong = new Audio("link_to_sound_file.mp3");

// To play the file, use .play()
mySong.play();

// To pause a file, use .pause()
mySong.pause();

// To loop a file, first play the file,
// then set .loop to true:
mySong.play()
mySong.loop = true;
```

Sound

```
/*
 * Create your own sound waves!
 */

// Construct a new Sound with a given note and sound wave type
var sound = new Sound("C4", "square");
var sound2 = new Sound("C1", "sawtooth");

// Set the tone to either a frequency value or a note value
sound.setFrequency(440);   // 440 Hz
sound.setFrequency("C4");  // Middle C note
sound.setFrequency("A2");  // Low A note
sound.setFrequency("A#8");  // High A sharp note

/*
 * Set the oscillator type for the sound wave. Options are:
 * Basic waves: "square", "sine", "triangle", or "sawtooth"
 */

sound.setOscillatorType("sine");
sound.setOscillatorType("square");

// Set the volume (in decibels)
sound.setVolume(2);

/*
 * Get information about the sound
 */

var currentVolume = sound.getVolume();
var currentNote = sound.getFrequency();
var currentOscillatorType = sound.getOscillatorType();


/*
 * Adding effects to the sound
 * Options are: "distortion", "reverb",
 * "tremolo", "vibrato", or "chebyshev"
 */

// Add a distortion effect at full capacity
sound.setEffect("distortion", 1);

// Add a tremolo effect at half capacity
sound.setEffect("tremolo", 0.5);

// Add a vibrato effect at 0 capacity
sound.setEffect("vibrato", 0);

/*
 * Starting and stopping the sound
 */

// Play the sound continuously
sound.play();

// Play the sound for 3 seconds
```

```
sound.playFor(3);

// Stop playing the sound immediately
sound.stop();
```

Misc

```
function start() {
    mouseClickMethod(turnRed);
}

// If you click on an object, turn it red.
function turnRed(e) {
    var elem = getElementAt(e.getX(), e.getY());
    if (elem != null) {
        elem.setColor(Color.red);
    }
}
```

Arrays

```
// Create an empty array
var arr = [];

// Create an array with values
var arr = [1, 2, 4];

// An array can have any type
var arr = [4, "hello", x];

// Access an element in an array
var elem = arr[i];

var firstElem = arr[0];

// Set an element in an array
arr[4] = 9;

// Looping over an array
for(var i = 0; i < arr.length; i++){
    var cur = arr[i];
    // process cur
}

// length of an array
var length = arr.length;

// Add to an array
arr.push(elem);

// Remove last element from array
var last = arr.pop();

// Finding the index of an element in a list
var index = arr.indexOf(5);

// Remove an element from a list at index i
arr.remove(i)
```

Maps/Objects

```
// Object literal
var obj = {
    name: "Jeremy",
    color: "blue"
};

// Objects/Maps have a collection of key, value pairs

// Set a value
obj["Jeremy"] = "123-4567";

// Get a value for a key
var phone = obj["Jeremy"];

// Loop over an object
for(var key in obj){
    var val = obj[key];
    // process key and val
}
```

Sets

```
// Make a new set named "newSet"
var newSet = new Set();

// Add to a set
newSet.add(5);

// Does a set contain a value
newSet.contains(5); // returns a boolean

// Number of elements in the set
newSet.size(); // returns an integer

// Make a new set named "setA"
var setA = new Set();
// Add 2 numbers to the set
    setA.add(1);
    setA.add(2);
// Make a new set named "setB"
var setB = new Set();
// Add 2 numbers to the set
    setB.add(2);
    setB.add(3);

// Assign setA to a new variable named "mutualSet"
var mutualSet = setA;
// Call the intersect function on "mutualSet" and pass in "setB"
mutualSet.intersect(setB);
return mutualSet; // returns a set with the shared values from "setA" and "setB"
```

Grids

```
// Create a grid named "newGrid"
var newGrid = new Grid(rows, cols);

// Get a value in a grid
var elem = newGrid.get(row, col);

// Set a value in a grid
newGrid.set(row, col, val);

// Getting dimensions
var rows = newGrid.numRows();
var cols = newGrid.numCols();

// Is a row, col pair inbounds
newGrid.inBounds(row, col);

// Set all values in grid to initial value
newGrid.init(0); // sets all grid values to 0

// Initialze a grid from an array
newGrid.initFromArray([
    [6, 3, 2],  // 0th row
    [2, 5, 1],  // 1st row
    [4, 3, 9],  // 2nd row
    [1, 5, 1]   // 3rd row
]);
```