CIS 111
Week 1

Phil Colbert
University of Oregon

Introduction

- CIS 111 focuses on introducing the JavaScript programming language
- JavaScript is widely used as the programming language of choice for web browser programming
- You will learn the basics of JavaScript, as well as introduce fundamental concepts and structure of an HTML web page

Programming?

- As most of you already know, programming is creating instructions that will make a computer do what we want
- Computers include mobile phones, laptops, tablets, and desktop computers
- Computers also include less obvious devices that also use computers, such as your car, printers, televisions, and the many devices classified as Internet of Things (IoT), such as a refrigerator with a browser, or a WiFi light

JavaScript

- JavaScript as a programming language may be used in many different settings
  - Within your browser
  - On a web server
  - Within an application that supports JavaScript
- We will focus on learning the basics of JavaScript in the following settings
  - Browser Console
  - HTML web page
- Other CIS/CIT classes focus on JavaScript on the web server

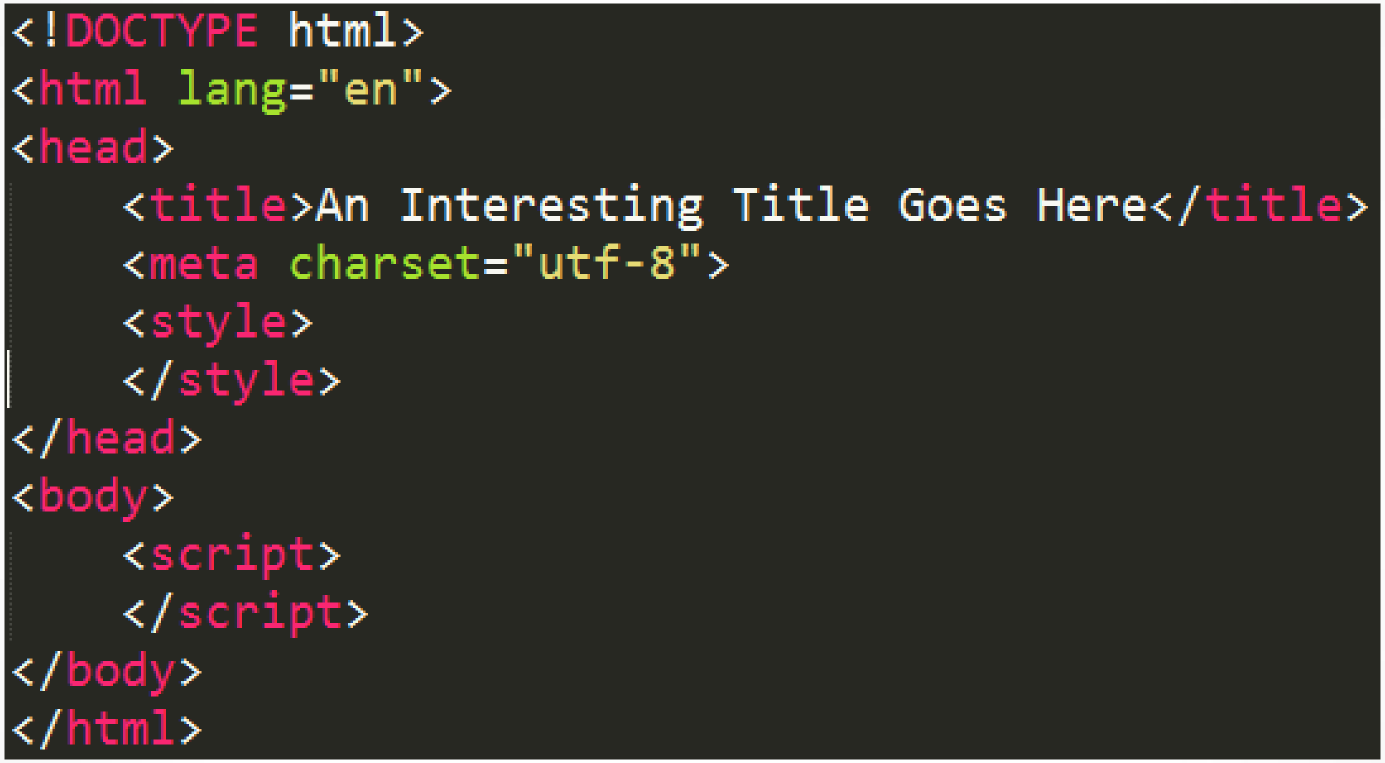
JavaScript

- Software programs such as JavaScript consist of lines of programming code that run, or execute
- JavaScript requires other software to run
- These JavaScript engines are found in most of the common web browsers today

Your First Program

- Let's introduce your first JavaScript program
- You will be creating a similar program in lab this week
- This first program will be contained within a web page and is adapted from the textbook (p. 9)
- **Note**: CIS 110 is a recommend prerequisite for CIS 111, but if you didn't take CIS 110, you should head over to the course syllabus in Canvas and follow the links to the recommended Code Academy online tutorial

Hello, World



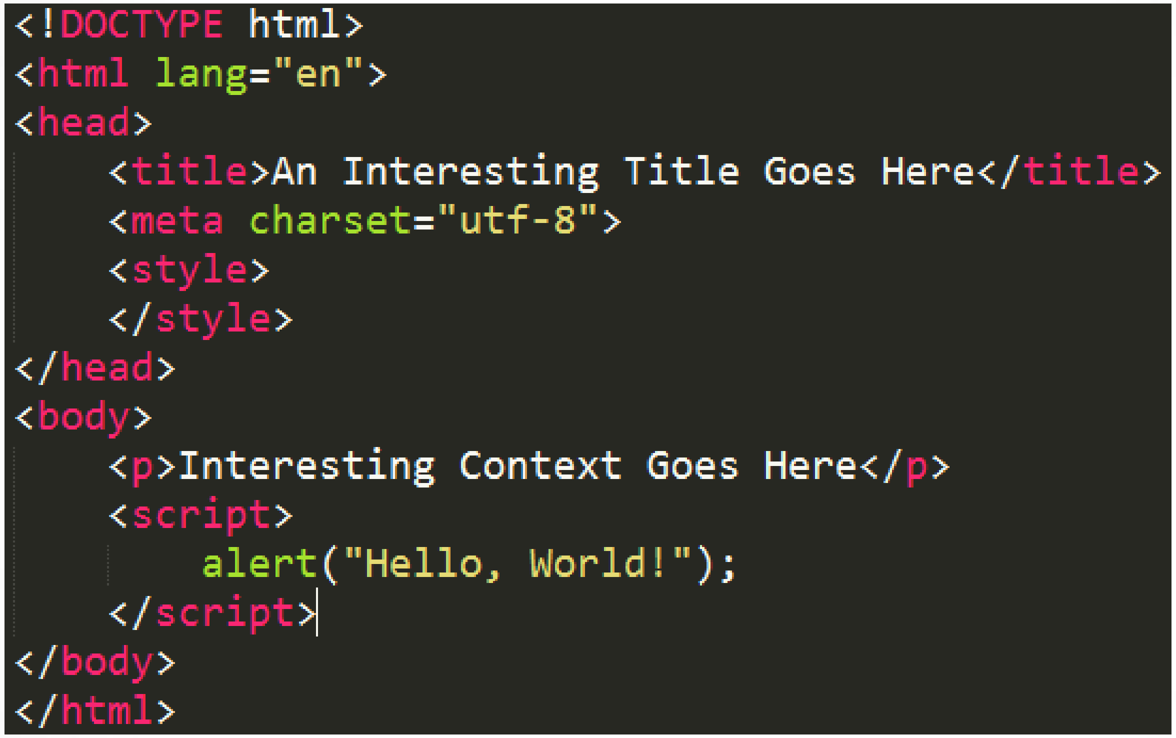
```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>An Interesting Title Goes Here</title>
    <meta charset="utf-8">
    <style>
    </style>
</head>
<body>
    <script>
    </script>
</body>
</html>
```

- The syntax highlighting is from the Sublime Text Editor

Hello, World

- If we display this web page in our browser, nothing will show up on the web page, and only the title will show up in the browser tab
- Let's add content to the web page, and JavaScript to display a dialog box

Hello, World



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <title>An Interesting Title Goes Here</title>
    <meta charset="utf-8">
    <style>
    </style>
</head>
<body>
    <p>Interesting Context Goes Here</p>
    <script>
        alert("Hello, World!");
    </script>
</body>
</html>
```

- Did you notice that the page content didn’t display until after the dialog box was dismissed?
- We’ll learn how to fix this issue later

Displaying Web Pages

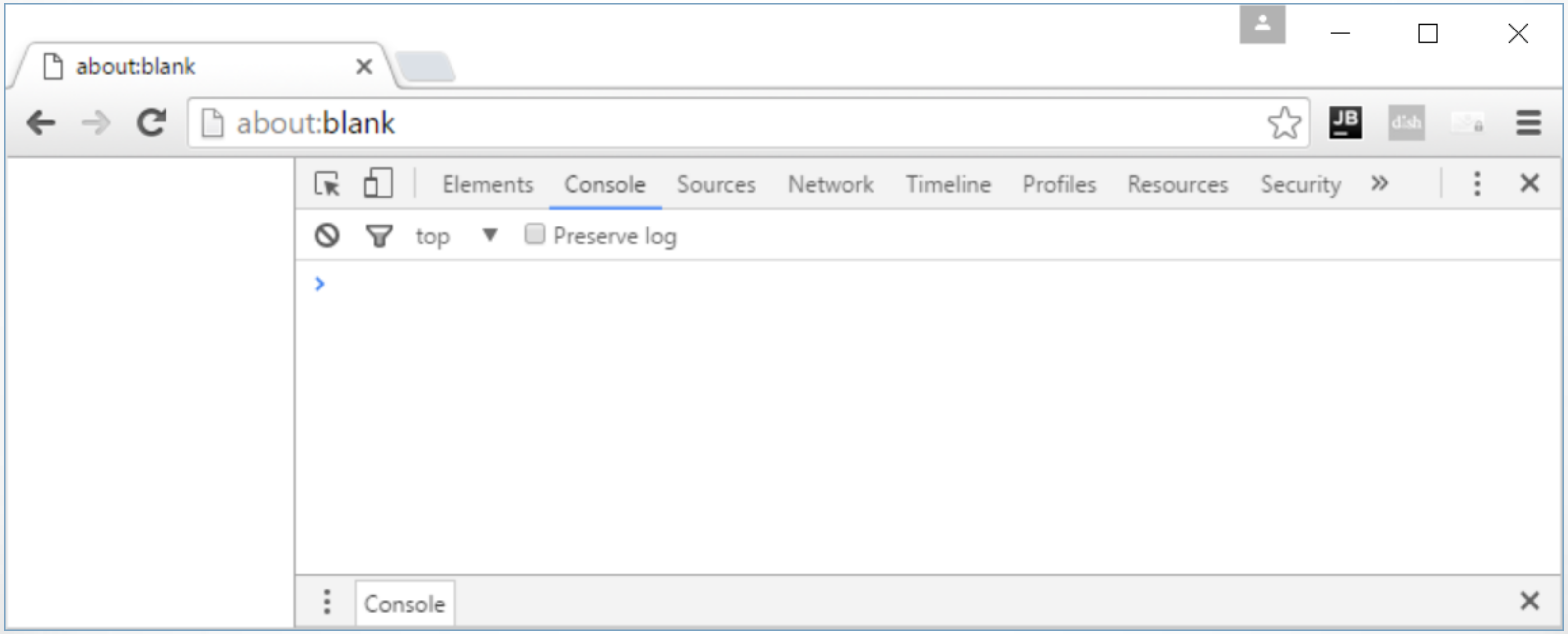
- We've all used browsers to navigate the web, but you can display web pages stored on your local computer
- How?
  - Double-click a web page (usually has an html file ending)
  - Drag the web page onto your browser
  - Use your browser File | Open menu options
  - Launch directly from your text editor

Learning JavaScript

- Although we've created our first web page with JavaScript, to learn JavaScript it's easier to focus on the language by simply entering JavaScript and have it execute (run)
- How? We'll initially be using the Console of the Chrome DevTools, functionality built into Chrome (and most other browsers as well)
- Remember, Chrome is the official browser of CIS 111, and will primarily be used for demonstrations throughout the term

Chrome Console

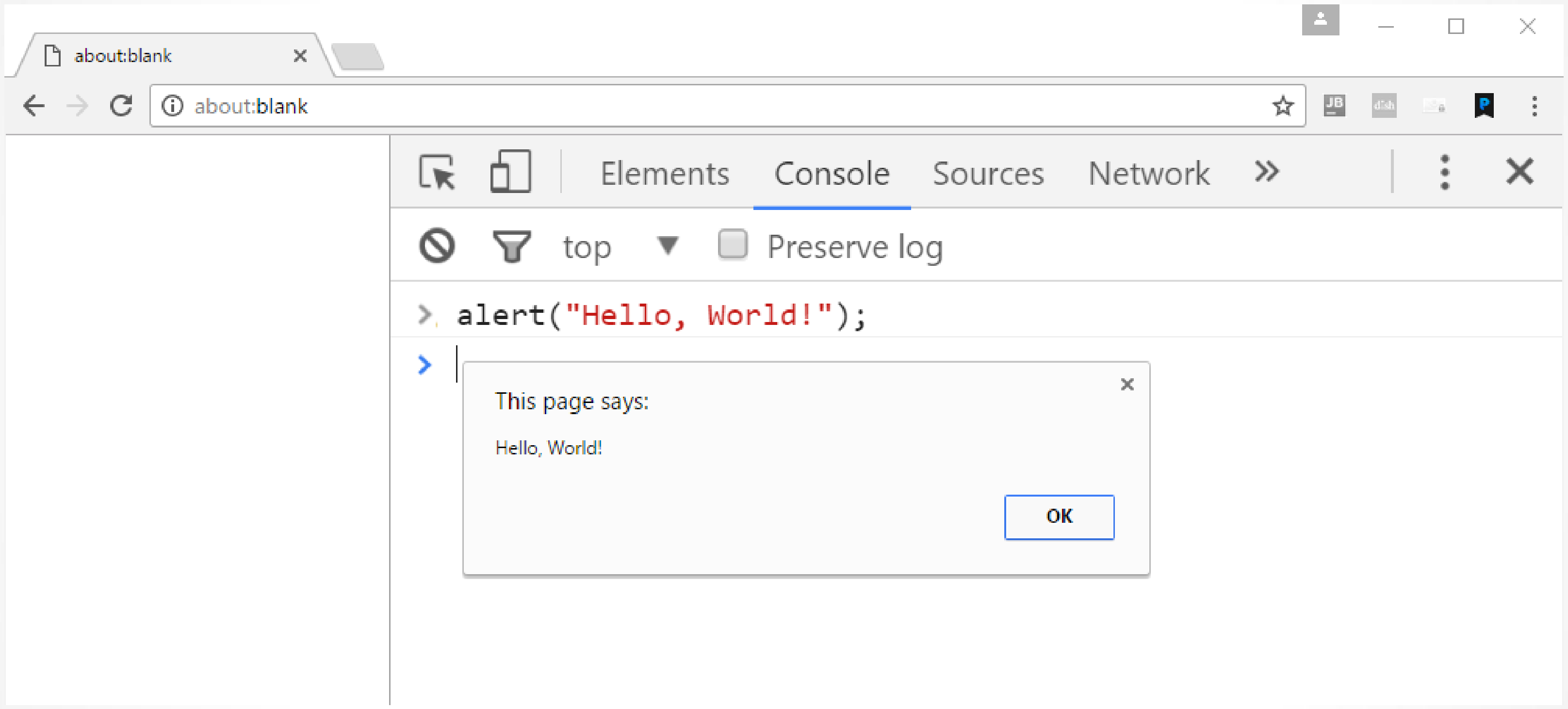
- Step 1: Open Chrome and enter about:blank as the URL
- Step 2: Access the Chrome JavaScript Console
  - Mac: Command key + Option key + J key
  - Windows/Linux: Control key + Shift key + J key



JavaScript and the Console

- Now that you have the Chrome JavaScript Console (console) open, you can type JavaScript directly into the console
- Let’s enter only the JavaScript contained within the script tag from our earlier Hello, World example into the Chrome Console, and press the enter key

Hello, World Via The Console



```
> alert("Hello, World!");
```

This page says:

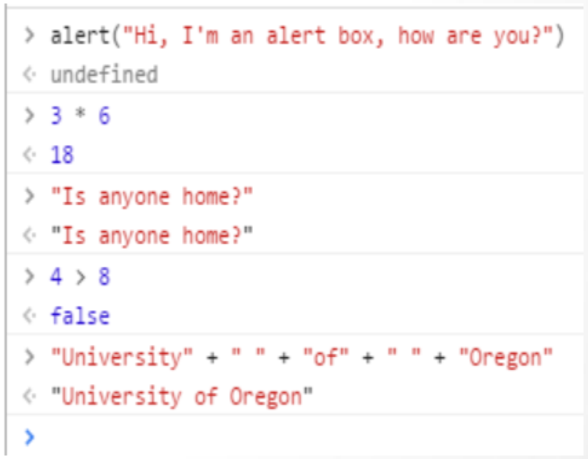
Hello, World!

- Note: The layout of the DevTools and Console may differ depending on your version and operating system

More Console JavaScript

- That was pretty easy!
- Let’s try a few more examples
- Type in the following JavaScript into the console and press the Enter/Return key

- 3 * 6
- “Is anyone home?”
- 4 > 8
- “University” + “ “ + “of” + “ “ + “Oregon”



```
> alert("Hi, I'm an alert box, how are you?")
< undefined
> 3 * 6
< 18
> "Is anyone home?"
< "Is anyone home?"
> 4 > 8
< false
> "University" + " " + "of" + " " + "Oregon"
< "University of Oregon"
>
```

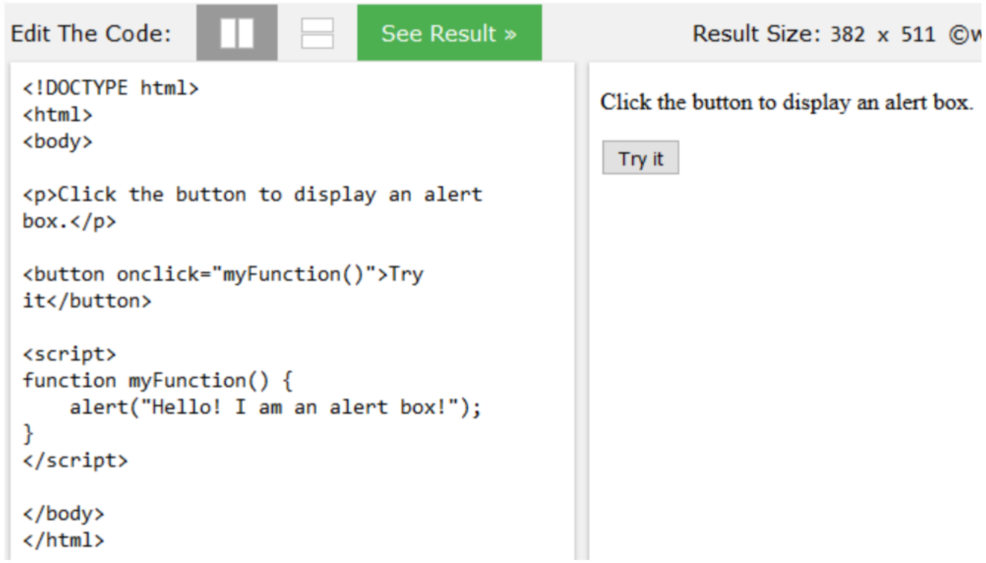
Web Browsers

- All of us have used a web browser (Chrome, Firefox, Safari, Internet Explorer)
- A web browser is a software application that
  - Runs on many devices
  - Uses the Internet as a communication medium
  - Sends and receives information using specific formation (such as http)
  - Understands and interprets HTML
  - Understands and interprets JavaScript
  - Understands and interprets CSS (Cascading Style Sheets)
  - HTML and CSS are the focus of CIS 110

Online Help

- The course textbook is a good JavaScript resource (although the author apparently finds JavaScript to be boring!)
- A great online resource for helping you learn JavaScript is the JavaScript Tutorial at the W3Schools website
- The W3Schools website provides numerous "Try it Yourself" links that demonstrate working examples of each topic
- The "Try it Yourself" examples also allow you to modify the code

Try It Yourself



```
<!DOCTYPE html>
<html>
<body>

<p>Click the button to display an alert
box.</p>

<button onclick="myFunction()">Try
it</button>

<script>
function myFunction() {
    alert("Hello! I am an alert box!");
}
</script>

</body>
</html>
```

Click the button to display an alert box.

- URL: W3Schools Try It Yourself from the Window alert() method page
- Tip: You can use Google to find specific JavaScript help at W3Schools using Google site search syntax:
  - site: w3schools.com JavaScript alert

JavaScript Tutorials

- Introduction
- Where To
- Output
- Syntax
- Statements
- Comments
- Variables
- Operators
- Arithmetic
- Assignment
- Data Types
- Function
- Scope
- Strings
- String Methods
- Numbers

JavaScript Tutorials

- Number Methods
- Math
- Dates
- Date Methods
- Arrays
- Array Methods
- Booleans
- Comparison and Logical Operators
- If...Else
- Switch
- For Loop
- While Loop
- Debugging
- Use Strict
- Style Guide and Coding Conventions
- Best Practices

Online Samples

- CSS 3D Solar System
- Animated Waterfall
- Alphabet Particle Animation
- Animated Adjustable Wire Cloth
- Blob Interaction
- Mouse Tracking Color Trails
- Growing Tree
- Animated Manipulating Hanging Cloth
- Geolocation Map
- Other demos

Coding

- Throughout the term, feel free to experiment with JavaScript by typing JavaScript directly into the browser Console
- We will also use a text editor for creating and editing your JavaScript and HTML files
- You have a number of editor choices, but I will primarily be using Sublime Text
- Sublime Text, as with many of the text editors, will use the filename extension (.js for JavaScript files) to perform syntax highlighting

Reminder: Code!

- Learning to program requires coding
- You should type and run each example in the textbook yourself (or as many as you can)
- Feel free to modify the examples!

Data Types

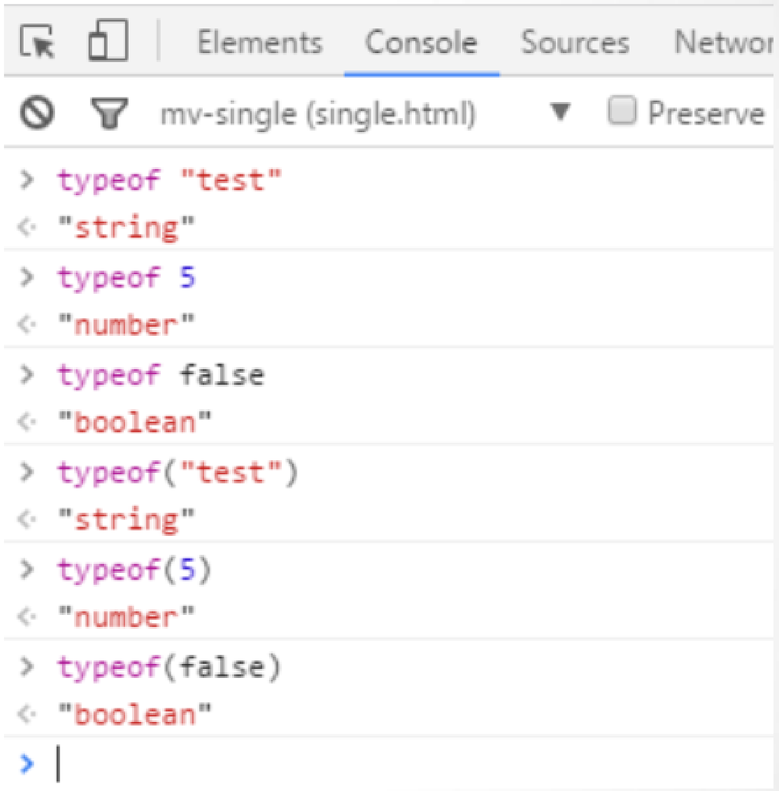
- Programming requires data interaction:
  - Store a name
  - Inquire about an age
  - Record the date of an event such as date of birth
  - Remember a preference such as providing helpful hints in a program
- Each of these data interactions require a different data type
- Data type: an expression of the fundamental characteristic of a piece of information

JavaScript Data Types

- Data type summary: We will learn the following JavaScript data types:
  - String: any sequence of characters such as you might type on a keyboard
  - Number: both integers (whole numbers) and floating points (numbers with a decimal place)
  - Boolean: true and false
  - Undefined: a value given to a variable not assigned a value, typically assigned by JavaScript
  - Null: no value given to a variable, especially objects, typically assigned by the programmer
  - Object: everything else, including dates

typeof Operator

- Before exploring further into data types, let's discuss a JavaScript operator we can use to determine "type"
- Use the **typeof** operator to determine the data type



```
> typeof "test"
<· "string"
> typeof 5
<· "number"
> typeof false
<· "boolean"
> typeof("test")
<· "string"
> typeof(5)
<· "number"
> typeof(false)
<· "boolean"
> |
```

Quotations Vs. Parenthesis

- Did you notice that we were able to use both quotation marks and parenthesis with the **typeof** operator?

```
typeof “test”    vs.    typeof(“test”)
```

- In both instances, “test” is information being processed by the **typeof** operator
- The parenthesis syntax is more common, and in cases we will discuss later, required

Strings

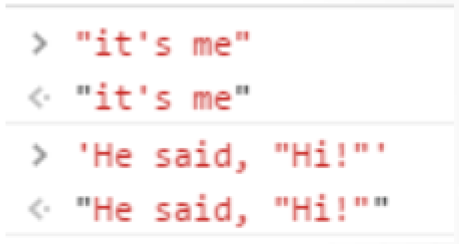
- A string is essentially a sequence of individual characters group together
- A common way to think of a string is any sequence of keys you can type on a keyboard
- Strings can contain more than “keyboard” characters

Strings

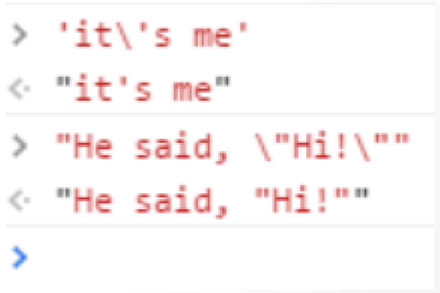
- Strings are defined by surrounding individual characters with either single or double quotation marks
- Regardless of quotation mark type, you must “pair” the quotation mark
  - ‘test’ or “test”
- Note: Word processor software often uses “curly quotations” so you may need to convert to straight quotations (Straight and Curly Quotes)

Strings

- To include the same type of quotation mark within a string you may
  - Use opposite quotation marks
  - Indicate to JavaScript to ignore the quotation mark character by preceding the quotation mark with a backslash, a technique known as “escaping”



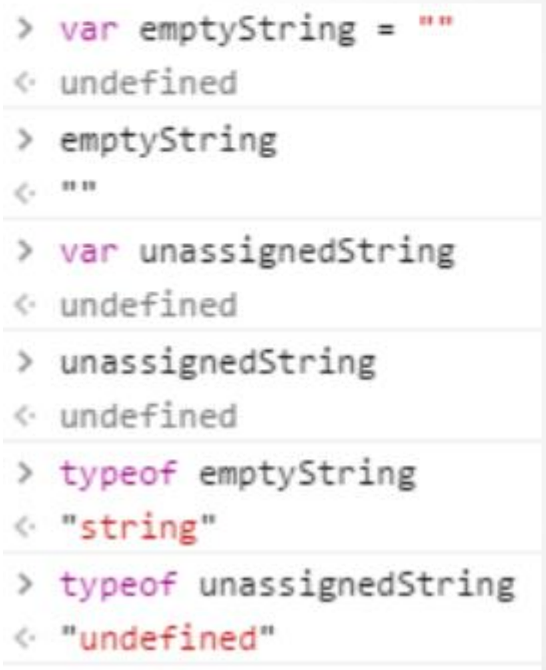
```
> "it's me"
<· "it's me"
> 'He said, "Hi!"'
<· "He said, "Hi!""
```



```
> 'it\'s me'
<· "it's me"
> "He said, \"Hi!\""
<· "He said, "Hi!""
>
```

Empty String

- A string can be assigned a value where the string content is empty
- An empty string **is** a string that has been assigned a value, and is of data type string
- Simply declaring a variable, even though you might intend the variable to be used as a string, does not automatically make the variable a string
- Unassigned variables are of data type undefined



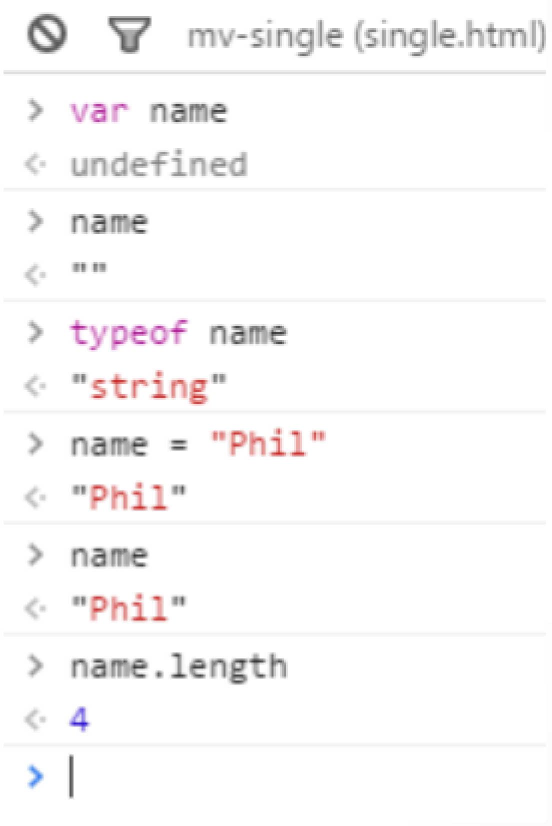
```
> var emptyString = ""
<· undefined
> emptyString
<· ""
> var unassignedString
<· undefined
> unassignedString
<· undefined
> typeof emptyString
<· "string"
> typeof unassignedString
<· "undefined"
```

Variables

- Simply typing in a string isn’t very useful if we want to remember user input, such as asking a user for their first name
- To work with data, we use a common programming element known as a variable
- A variable is simply a name given to a location in computer memory
- We use the variable name to access the information at the memory location

Variables

- In JavaScript, a variable is declared using the **var** keyword
- By default, declared variables are of data type undefined in other contexts
- Undefined means no data has been assigned, so JavaScript does not know the data type



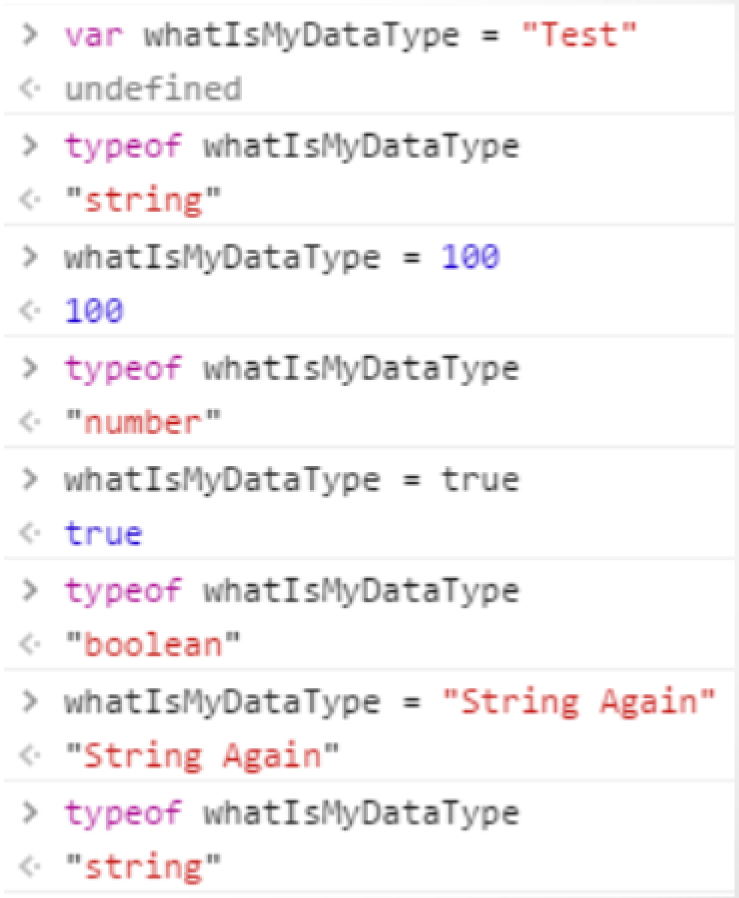
```
> var name
<· undefined
> name
<· ""
> typeof name
<· "string"
> name = "Phil"
<· "Phil"
> name
<· "Phil"
> name.length
<· 4
> |
```

Use Strict Directive

- You can declare a variable without using the var keyword, but this practice is **strongly** discouraged, and points will be deduced for missing var statements
- To force your JavaScript code to require the var keyword, we can add what's known as a directive to our JavaScript code
- We will revisit this requirement when we move out of the Console
- The directive is "use strict" (W3Schools Use Strict)

Changing Data Types

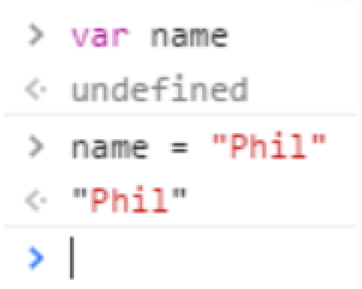
- We do need to note that JavaScript allows us to change the data type of the contents of a variable
- A string can become a number, and then a Boolean, back to a string, etc.
- This “feature” of the JavaScript language indicates that JavaScript is a “loosely typed” language
- You should avoid changing the data type of your JavaScript variables



```
> var whatIsMyDataType = "Test"
< undefined
> typeof whatIsMyDataType
< "string"
> whatIsMyDataType = 100
< 100
> typeof whatIsMyDataType
< "number"
> whatIsMyDataType = true
< true
> typeof whatIsMyDataType
< "boolean"
> whatIsMyDataType = "String Again"
< "String Again"
> typeof whatIsMyDataType
< "string"
```

Variable Assignment

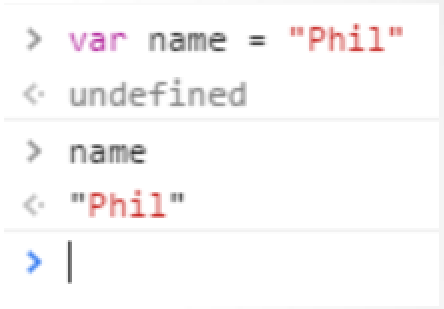
- Once declared, we use the variable name to access any information stored in the memory named by the variable
- To set, or change, the variable data, we use the assignment operator, =



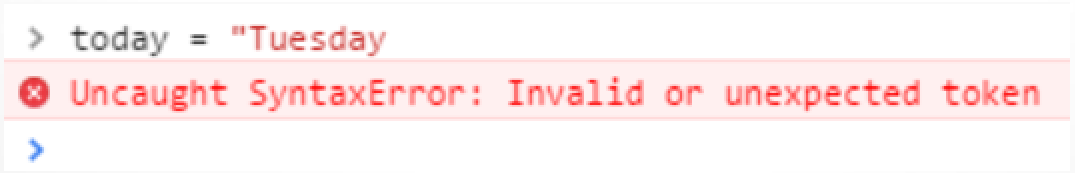
```
> var name
<· undefined
> name = "Phil"
<· "Phil"
> |
```

Variable Assignment

- We can declare a variable and assign a value at the same time
- The **var** keyword is technically not required in JavaScript, but omitting **var** can lead to later errors, so we will always use **var** when declaring a variable
- **var** is required when defining a variable in the Console



```
> var name = "Phil"
<· undefined
> name
<· "Phil"
> |
```



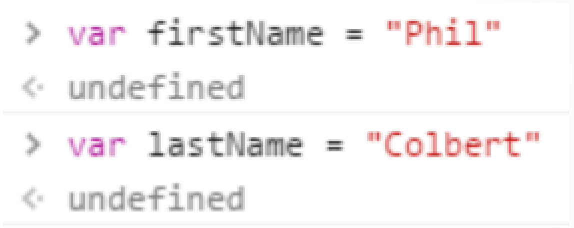
```
> today = "Tuesday
✖ Uncaught SyntaxError: Invalid or unexpected token
>
```

Variable Naming Requirements

- As with most programming elements, variables have naming requirements
  - May start with any uppercase or lowercase letter
  - May start with an underscore (_)
  - May start with a dollar sign ($)
  - May contain other uppercase or lowercase letters
  - May contain other underscores
  - May contain numbers
  - May not start with a number
  - Must NEVER contain a space
- Variables are case sensitive, so watch out for typos!

Variable Naming Conventions

- You should name your variables something meaningful
  - Avoid really long names
  - Avoid abbreviations that may not mean anything to someone else
- Use "camelcase"
  - Start with a lowercase letter, then use uppercase letters for the start of each word



```
> var firstName = "Phil"
<· undefined
> var lastName = "Colbert"
<· undefined
```

Remember to ignore the "undefined" responses from the Console

JavaScript Statements

- JavaScript is composed of statements
- So far, we've seen fragments of JavaScript code, and we'll continue to use fragments as we explore the basics of JavaScript
- You should begin to think in terms of more complete JavaScript syntax

Expressions and Statements

- Many of the JavaScript examples we've used so far are considered expressions, or pieces, of JavaScript code
- Even though these expressions will execute in the Console, only actual JavaScript statements will execute when we code complete JavaScript programs (JavaScript Statements)
- JavaScript statements are separated by a semi-colon (;)

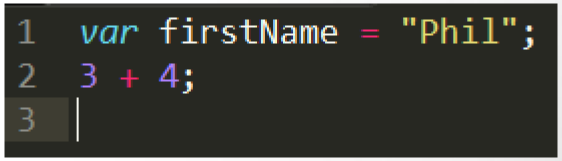
Expressions and Statements

- Below some of the JavaScript code we've seen so far is divided into expressions and statements
- Note that a semi-colon has been added to both lists, but even adding a semi-colon to a JavaScript expression does not make it a statement

Expressions	Statements
true;	var name;
"Phil";	name = "Phil";
3 * 4;	var result = 3 * 4;
typeof("test");	var datatype = typeof("test");
4 > 8;	alert("hi!");

Validating JavaScript Code

- As you learn JavaScript, you will undoubtedly run into coding errors
- The Browser Console will alert you to some errors, but as we've seen, the Console allows JavaScript expressions
- Using an editor such as Sublime to code JavaScript will help visualize JavaScript syntax through color highlighting, but will not catch errors
    - Note in the code snippet that the expression 3 + 4; appears to be valid JavaScript code

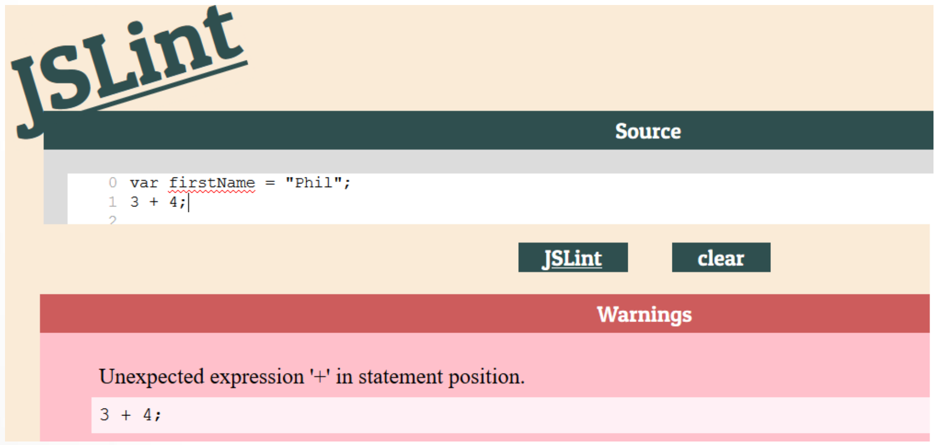


```
var firstName = "Phil";
3 + 4;

```

Validating JavaScript Code

- So how will we validate our JavaScript code (in other words, find errors)?
- Eventually we'll run our JavaScript programs in the browser, but you can also use an online JavaScript syntax checker program called JSLint (http://jslint.com/)



```
var firstName = "Phil";
3 + 4;
```

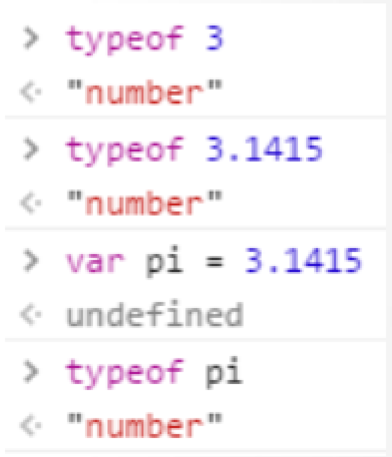
Unexpected expression '+' in statement position.

```
3 + 4;
```

- You can also add the JSHint Package to Lint directly within Sublime (see course Piazza for instructions)

Numbers

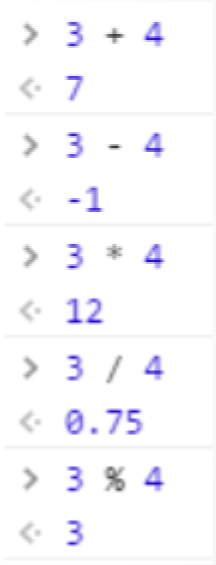
- Finally, we made it to the numbers data type!
- JavaScript supports both integer numbers (a whole number, or a number with no decimal component), and floating point numbers (a number with a decimal component)



```
> typeof 3
<· "number"
> typeof 3.1415
<· "number"
> var pi = 3.1415
<· undefined
> typeof pi
<· "number"
```

Arithmetic Operations

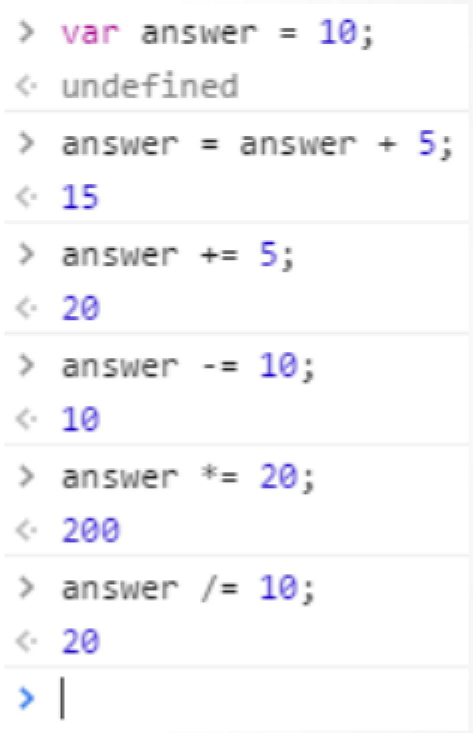
- Yes, you can do arithmetic using JavaScript
  - + : addition
  - - : subtraction
  - * : multiplication
  - / : division
- You can also get the remainder of a division(useful for loops and conditions)
  - % : modulo, or modulus operator
- Note: All of these math calculations are expressions, not statements



```
> 3 + 4
<· 7
> 3 - 4
<· -1
> 3 * 4
<· 12
> 3 / 4
<· 0.75
> 3 % 4
<· 3
```

Arithmetic Shortcuts

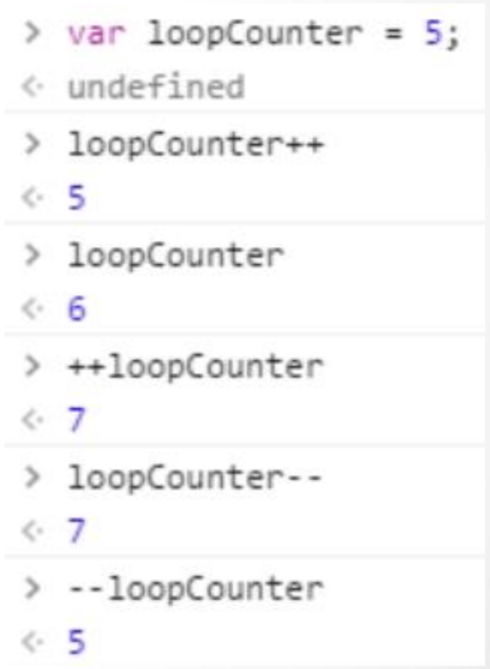
- Adding a number to a variable will require a complete JavaScript statement
- Common arithmetic operations that change a variable have shortcut techniques frequently used in JavaScript
  - += : "plus equals"
  - -= : "minus equals"
  - *= : "times equals"
  - /= : "divide equals"



```
> var answer = 10;
<· undefined
> answer = answer + 5;
<· 15
> answer += 5;
<· 20
> answer -= 10;
<· 10
> answer *= 20;
<· 200
> answer /= 10;
<· 20
> |
```

Increment Operator

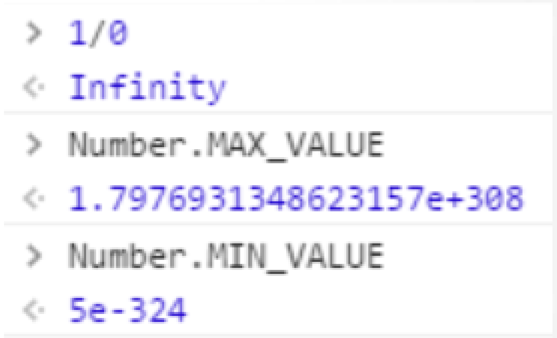
- When changing a variable by one, JavaScript has two special operators
  - ++ : increment operator
  - -- : decrement operator
- These operators can come before a variable, or after a variable
  - Before a variable is referred to as pre-increment or pre-decrement
  - After a variable is referred to as post-increment or post-decrement
- What's the difference between pre and post?
  - Pre: The calculation is performed, then the value is returned
  - Post: The value is returned, then the calculation is performed



```
> var loopCounter = 5;
<· undefined
> loopCounter++
<· 5
> loopCounter
<· 6
> ++loopCounter
<· 7
> loopCounter--
<· 7
> --loopCounter
<· 5
```

Infinity

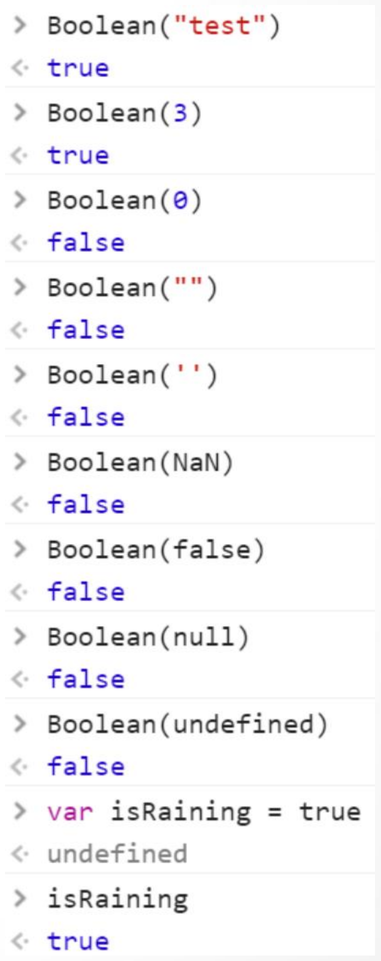
- Data types are stored in the computer as a series of bits
- The data type determines the number of bits used to store a value of a variable
- The number of bits determines the maximum size that may be represented
- If an attempt is made to store a number that exceeds the maximum storage size, positive or negative infinity will be returned, depending on the sign of the calculation
- Infinity can also be returned by dividing by zero
- We did not cover exponentials, but the number properties MIN_VALUE and MAX_VALUE indicate the minimum and maximum values of a number data type



```
> 1/0
< Infinity
> Number.MAX_VALUE
< 1.7976931348623157e+308
> Number.MIN_VALUE
< 5e-324
```

Booleans

- The Boolean data type has only two values
  - true
  - false
- Booleans are very use for asking questions
- You can convert to a Boolean using the Boolean JavaScript method



```
> Boolean("test")
<· true
> Boolean(3)
<· true
> Boolean(0)
<· false
> Boolean("")
<· false
> Boolean('')
<· false
> Boolean(NaN)
<· false
> Boolean(false)
<· false
> Boolean(null)
<· false
> Boolean(undefined)
<· false
> var isRaining = true
<· undefined
> isRaining
<· true
```

Comments

- You can (and should) comment your JavaScript code
- JavaScript comments are ignored, and come in two versions
  - Single line comment:

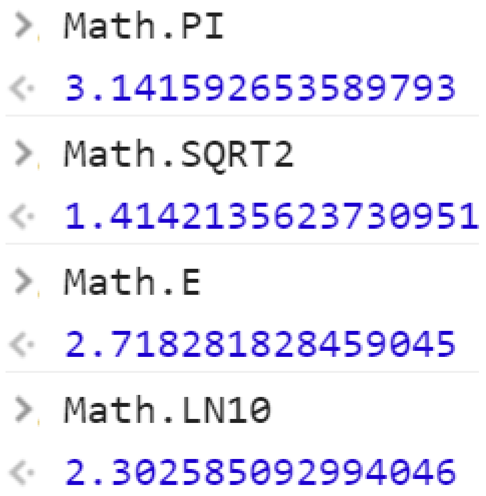
    ```
    // This is a single line JavaScript comment
    ```
  - Multi-line comment

    ```
    /* This is a multi-line
    JavaScript comment */
    ```
  - Comments may be used before you begin programming to outline the programming tasks

Math Object

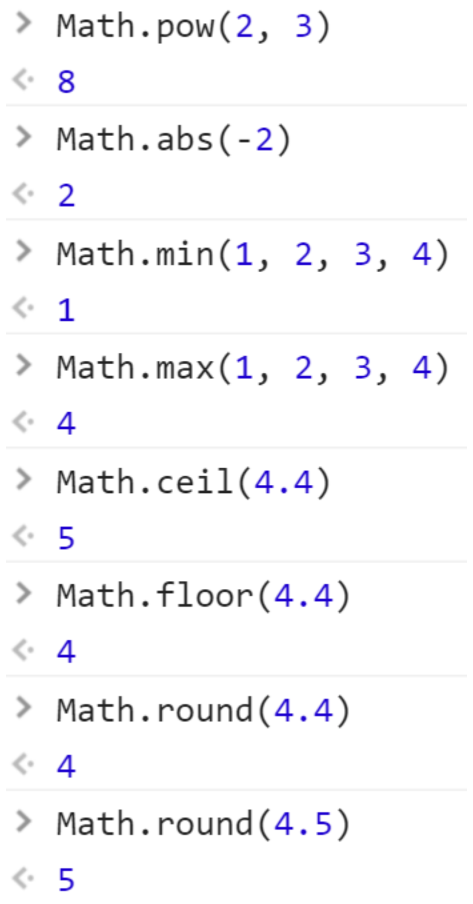
- The Math object has a number of mathematical properties that are essential property values that are constants
  - Eight mathematical constants
  - Numerous math operations
- Let's examine the Math constants and methods at the W3School's Javacript Math Object web page, and then let's see some examples of using the Math object

Math Constants



```
> Math.PI
< 3.141592653589793
> Math.SQRT2
< 1.4142135623730951
> Math.E
< 2.718281828459045
> Math.LN10
< 2.302585092994046
```

Math Methods



```
> Math.pow(2, 3)
<· 8
> Math.abs(-2)
<· 2
> Math.min(1, 2, 3, 4)
<· 1
> Math.max(1, 2, 3, 4)
<· 4
> Math.ceil(4.4)
<· 5
> Math.floor(4.4)
<· 4
> Math.round(4.4)
<· 4
> Math.round(4.5)
<· 5
```