Hello Code 2

JavaScript

Duane DuaneHaworth@gmail.com

Motivation

Fullstack.io, now https://www.newline.co, asked their readers “If your best friend asked you how to learn how to program, what would you tell them?” Instead of getting a list of URLs, people were giving great advice on how to **approach the problem of learning how to program**.

- Believe you can do it
- Have a project idea, what do you want to build
- What technology do you want to use
- Lots of resources
- Mentor
- Don’t need a CompSci degree

Why Learn JavaScript

- Language used by browsers
- Enhance User Experience
  - Add/remove/alter HTML
  - Add/remove/alter CSS
- Get and Submit Data
- Games
  - https://codepen.io/hellokatili/pen/xwKRmo
- Forgiving Language
  - Pros/Cons
- Syntax rules similar to others
  - Java, C, C++, C#

What We're Going to Use in this Course

- W3Schools
  - https://www.w3schools.com/js/
- Glitch
  - www.glitch.com
- GitHub
  - https://github.com/DSHaworth/HelloCode2

- Questions

Preliminaries

Syntax
* Language Rules

Statement
* Instruction executed by computer
* Executed in order they are written

Variables
* Hold data of various data types and can be changed at anytime

Data Types
* boolean
* number
* string
* object
* undefined
* null

Preliminaries continued

Conditions
* Execute block of code when condition true

Loops
* Execute block of code while condition true

Functions
* Block of code designed to perform a task

Reserved Words
* Words claimed by JavaScript. Off limits

Comments
* Not statements
* Not Executed
* Document Code
* Prevent Execution
* Single Line / /
* Multi-Line / * ... * /

Adding JavaScript to a Webpage

Internal Script

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Hello!</title>
  </head>
  <body>
    <script>
      alert("JavaScript says Hi");
    </script>
  </body>
</html>
```

External Script

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Hello!</title>
  </head>
  <body>
    <script src="script.js"></script>
  </body>
</html>
```

Functions

Functions are an important concept, so we cover it early

Functions

- Block of code that performs a task
- Executed when something “invokes” it.
- Can take zero or more arguments
- Can return a value.

Right now, we’re looking at functions provided by the JavaScript Window API.

- API
  - Application Programmer Interface
- Window Object
  - Global
- Some Window API functions we’ll use now
  - alert
  - prompt

CAUTION

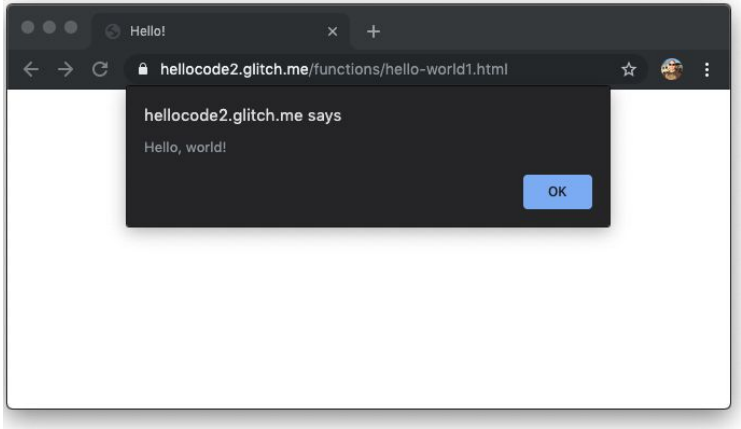
- Case Matters!!!!!!
  - `Alert` is not the same as `alert` is not the same as `ALERT`
- This is true for `variables` and `functions`
- Keep Code Blocks Aligned

Call JavaScript from HTML

- We’re going to ***invoke*** our JavaScript when we press an HTML button.
- The HTML button provides an **attribute** called **onclick** where we can assign our JavaScript function to execute.

alert

- Displays an alert box.
  - The developer determines the message to display



01-hello-world1-end.html

https://github.com/DSHaworth/HelloCode2/blob/master/01-functions/01-hello-world1-begin.html

- Button click event
- Executes function
- Displays alert - “Hello, World”

Variables and Data Types

Variables and Data Types

- Variables
  - Store any Data Type
  - Can change (hence the name)
- Declare variable using `var`

```
var state = “Nebraska”;
var age = 152;
var isState = true;
var cities = [“Omaha”, “Lincoln”];
```

- Data Types
  - string
  - number
  - boolean
  - object
  - array
  - undefined
  - null

Data Type: String

- Anything within quotes
- Type of quotes don't matter. (but it does)
  - "It's easier with double quotes"
  - 'It's easier with double quotes' (Error)
  - 'It\'s easier with double quotes'
- Rule of thumb for numbers
  - If you don't do math on it, it's a string
    - Phone number
    - Social Security Number
    - Date (though there is a date object)
- Examples
  - "Hello Code 2 Rocks"
  - 'Hello Code 2 Rocks'
  - '10/11/2019'
  - '800-555-1212'

Data Type: Number

- Any type of number
  - With decimal or no decimal
- Numbers in quotes are strings
- JavaScript will try to convert strings to numbers.
  - Safer to convert

- Examples
  - 1
  - 3.14
  - Numbers, but may not be what you expect
    - 10/11/2019
    - 800-555-1212

Data Type: Boolean

- One of two values
  - true
  - False
- Everything with a value is true
- Everything without a values is false

- True values
  - “Hi”
  - 3
  - “false”
    - This is a string
- False values
  - 0
  - “”
  - null
  - undefined

Data Type: Object

- Contain many values
  - Name:Value Pairs
- Can contain **functions**

JavaScript Object Types

- Date

- Example

```
var loc = {
  city: "Omaha",
  state: "NE",
  highTemp: 50,
  lowTemp: 30,
  statehood: "3/1/1867",
  inUS: true
};
```

Data Type: Array

- Store multiple values in a single variable
- Has an API

Example:

```
var cars = ["Saab", "Volvo", "BMW"];
```

Data Type: Undefined and Null

- Undefined
  - Variable declared, but no value assigned.
  - It doesn't know what it is yet
    - Number, string, boolean, etc..
- Null
  - Represents no value

Variables - Details, details, details

- Naming Rules (**syntax**)
  - Can contain letters, numbers, underscores, dollar signs
  - Must BEGIN with a letter, underscore, or dollar sign
  - ARE CASE SENSITIVE
  - Reserved words cannot be used as names
- Tips
  - Use a variable name that represents the data to be stored
    - Do: `var firstName = 'John';`
    - Don't: `var x = "7/4/1776";`

02-hello-world2-begin.html

https://github.com/DSHaworth/HelloCode2/blob/master/01-functions/02-hello-world2-begin.html

Getting value from HTML

- HTML elements can have an id attribute
- DOM - Document Object Model
  - Functions
  - Attributes
- input
- var val =
  document.getElementById(id).value

```
<input type="text" id="name" />
<div id="greetingsDiv"></div>

<script>
 var name = document
                    .getElementById("name").value;

 document
   .getElementById("greetingsDiv")
   .innerHTML = "Hello, " + name;
</script>
```

Concatenation

- Fancy word for putting strings together.
- In programming, there are multiple ways of doing the same thing.
- Examples
  - `var name = prompt("What is your name?");`
  - `var greetings = "Hello, " + name;`

    Other ways

  - `var greetings = "Hello, ".concat(name);`
  - `var greetings = `Hello, ${name}!`;  // back-ticks (under tilda ~)`

03-say-hi-begin.html

https://github.com/DSHaworth/HelloCode2/blob/master/01-functions/03-say-hi-begin.html

Create your own function

- function names follow the same restrictions as variable names
  - Letters, numbers, underscore, dollar signs
  - Must start with a letter, underscore, or dollar sign
  - Can take 0 or more arguments (parameters)
  - Can, but doesn't have to, return a value
  - Tip
    - Have function name represent what it does.
    - A function does something

04-get-name-function.html

https://github.com/DSHaworth/HelloCode2/blob/master/04-get-name-function-begin.html

get-value.html

https://github.com/DSHaworth/HelloCode2/blob/master/05-get-value-begin.html

getName() vs getValue()

You may have noticed, we're not doing anything different in `getName()` than what we can get from `getValue()`.

What's the point of `getName()`?

We're going to make `getName()` better by doing some **Data Validation**

Review

1) alert(“Hello, World”)
2) alert(greeting)
3) alert(prompt(“name”))
4) alert(prompt(question))

The idea is to see a progression from spitting out hard-coded string, to string determined by the user

What’s Next

Now we’re going to look at the User Input

1) Did user press Say Hi without entering anything?
    a) Return “”

We don’t want our app to respond with “Hello, !”

We want to ***validate*** user input

Challenge 10 Minutes

https://www.w3schools.com/js/js_htmldom.asp
https://www.w3schools.com/js/js_htmldom_methods.asp

Some liked the Age calculator. Write your own Age Calculator. :-)

DOM Object

Navigate to
https://github.com/DSHaworth/HelloCode2

Click on **calculate-age-begin.html**

In the function **askDob** is the ***algorithm*** for calculating age.

Fill in under the **Get** and **Display**

Decisions

Making decisions based on comparisons and logic

Decision Making

- Problems with our askQuestion function
  - Returns null on cancel
    - `Hello, null!` doesn't make sense
  - Returns "" when nothing entered
    - `Hello, !` doesn't make sense

- This is where things start to get intense.
- Several "moving parts" are being introduced here.

- Decisions consists of
  - Condition statements
    - Comparisons
    - Logic
    - Evaluated Left to Right

Comparison Operators

==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	great than or equal to
<=	less than or equal to

Remember

= assignment operator (assigns value to variable)

Logical Operators

`&&` AND - Both conditions need to be true for the whole thing to be true
`||` OR - Only one condition needs to be true for the whole thing to be true
`!` NOT - Reverses current value

Programmer Joke:
`!false - It's funny because it's true.`

Truth Tables

Truth Tables AND

A	B	A && B
T	T	T
T	F	F
F	T	F
F	F	F

Truth Tables OR

A	B	A || B
T	T	T
T	F	T
F	T	T
F	F	F

Decisions - Putting it all together

- Condition Statements are the test
  - `if else`
  - `switch`
- Comparison Operators are the evaluation
  - ==,===
  - !=, !==
  - >, >=, <, <=
- Logical Operators combine evaluations
  - &&
  - ||
  - !

condition01.html

- Hello Code 2 - Exercises
  - Condition null

condition02.html

- Hello Code 2 - Exercises
  - Condition empty string

logic01-bad-demo.html

- Hello Code 2 - Exercises
  - Test for the null and empty string
  - Logic Bad 1

logic01.html

- Hello Code 2 - Exercises
  - Putting the null test and empty test together
  - Logic Fixed 1

String API

Common String Methods

- toLowerCase()
- toUppserCase()
- ***trim()***
- replace(oldValue, newValue)
- indexOf(valueToFind)
- lastIndexOf(valueToFind)

logic02-bad-demo.html

- Hello Code 2 - Exercises
  - Propblem….User enters spaces
  - Logic Bad 2

logic02.html

- Hello Code 2 - Exercises
  - Conditions are tested left to right
  - Logic Fixed 2

Challenge.html

https://github.com/DSHaworth/HelloCode2/blob/master/02-conditionsLogic/logic02-assignment.html

- Put all the logic in askQuestion
  - If result = null or “” after trimming,
    - return null
  - return result trimmed
- If name
  - Display name
- Else
  - Display error

Review

Functions, Comparing, and Logic

Functions

Functions are commands.

You can send arguments (within parenthesis)

You can return a value

```
function multiplyTwoNumbers(num1, num2){
  return num1 * num2;
}

var result = multiplyTwoNumbers(5,6);
```

API Application Programmer's Interface

```
string.trim()
document.getElementById("id")
Date.parse("12/25/2019")
```

User Defined

```
multiplyTwoNumbers(3, 6)
```

Comparison and Logical Operators

Comparison Operators

== - Equal by value only
=== - Equal by value and by type
!= - Not equal by value only
!== - Not equal by value and type
> - Greater than
>= - Greater than or equal to
< - Less than
<== - Less than or equal to

Logical Operators

&& - AND - Both sides must be true to be true
|| - OR - If either side is true, it's all true
! - Reverse

Conditional Statements (if, loops)

Comparison Operators

== - Equal by value only
=== - Equal by value and by type
!= - Not equal by value only
!== - Not equal by value and type
> - Greater than
>= - Greater than or equal to
< - Less than
<== - Less than or equal to

Logical Operators

&& - AND - Both sides must be true to be true
|| - OR - If either side is true, it's all true
! - Reverse

Code Reviews

```
if (age < 5) {
  return "Free you Pay Nothing $0";
}
else if (age >= 5 && age < 12) {
  return "Your Price is $3";
}
else if (age >= 12 && age < 21) {
  return "Your Price is $5";
}
else if (age >=21 || age <=50) {  // Nothing more shall pass
  return "Sorry you drew the short straw your Price is $8";
}
else (age > 50) {
  return "Your Price is $5.  Have a great day."
}
```

```
if (age < 5) {
  return "Free you Pay Nothing $0";
}
else if ( age < 12) {
  return "Your Price is $3";
}
else if ( age < 21) {
  return "Your Price is $5";
}
else if ( age <=50) {
  return "Sorry you drew the short straw your Price is $8";
}
else {
  return "Your Price is $5.  Have a great day.";
}
```

Code Reviews

```
<input id = "tempInput" />
<button onclick="toCelsius(tempInput)">Try it</button>
<p id="demo"></p>

<script>
function toCelsius(f) {
  return (5/9) * (f-32);
}

document.getElementById("demo")
                .innerHTML = toCelsius(tempInput);
</script>
```

```
<input id = "tempInput" />
<button onclick="toCelsius(tempInput)">Try it</button>
<p id="demo"></p>

<script>
function toCelsius(f) {
  return (5/9) * (f-32);
}

var tempInput = 212; // Hard Coded

document.getElementById("demo")
                .innerHTML = toCelsius(tempInput);
</script>
```

Code Reviews

```
<input id = "tempInput" />
<button onclick="toCelsius(tempInput)">Try it</button>
<p id="demo"></p>

<script>
function toCelsius(f) {
  return (5/9) * (f-32);
}

document.getElementById("demo")
                .innerHTML = toCelsius(tempInput);
</script>
```

```
<input id = "tempInput" />
<button onclick="convert()">Try it</button>
<p id="demo"></p>

<script>
function convert(){
  var val = document.getElementById("tempInput").value;

  document.getElementById("demo")
                .innerHTML = toCelsius(val);
}
function toCelsius(f) {
  return (5/9) * (f-32);
}
</script>
```

Loops

Interacting with HTML Input Elements

Loops????

- When working with data, you're inevitably going to be working with arrays.
  - An Array is a series of data, strings, numbers, dates, and/or objects that contain any combination.
    - `[1,2,3]`
    - `["Peter", "Paul", "Mary"]`
    - `["1/1/2019", "11/11/2019", "12/25/2019"]`
    - `HTML Elements`
- Loops are used to ***iterate*** through an array
  - When looping through an Array, it always starts at **0**
  - Array is an object, it has properties and methods
    - .length
    - .sort()

Demo Array

- Go to **Developer Tools** on your browser
  - Go to **Console**
- Enter:
  - `var names = [ "Peter", "Paul", "Mary" ]`
  - `names[0]`
  - `names[2]`
  - `names[3]`
  - `names.length`
  - `names.sort()`

Loops???? Continued

- Loops need to know:
  - Where to start
  - Where to stop
    - test (**CONDITIONS**)!!!!!
  - How to move on
- Three Types of Loops
  - for
  - while
  - do while

Loops CAUTION

- Loops without a stopping point are called **ENDLESS LOOPS.**
  - **Endless Loops** are ***BAD***
  - Can bog down browser to the point of having to force browser to close
    - "Long running script, would you like to stop"
- It's going to happen.
  - It happened to me writing the demos for this.
  - I changed variable names to something more clear, but didn't change them all.

for loop

- Ugliest of the loops
  - Most common and most useful
  - Everything is on one line (where to start, stop, and how to move on)

```
for(start ; stop test ; move on){
  Everything between the {} is in the loop.
  BEST PRACTICE: ALWAYS USE {} //IF, ELSE, ELSE IF, SWITCH, ALL LOOPS
}
```

for loop

```
var start = 0;
var stop = 10

for(idx = start ; idx < stop ; idx++){
  console.log(idx);
}
```

`idx++`??????????

- ++ is an incrementer
  - Shortcut for `idx = idx + 1;`

for loop challenge 01 10 minutes

https://github.com/DSHaworth/HelloCode2/blob/master/03-loops/for-loop-challenge01-begin.html

```
for("start" ; "stop test" ; "move on")
```

- Replace “start” ; “stop test” ; “move on” with valid JavaScript
  - See previous slide
  - Use provided variables

“But Teach, you didn’t define `idx` in previous slide!!!”
Hoisting - JavaScript will create a definition for you.
"use strict" forces you to define all variables... Makes JavaScript more “Type A”

for loop challenge 01 final

https://github.com/DSHaworth/HelloCode2/blob/master/03-loops/for-loop-challenge01-final.html

```
for(var idx = start, stop = 10 ; idx < stop ; idx++){
    console.log(idx);
}
```

for loop challenge 02 10 minutes

https://github.com/DSHaworth/HelloCode2/blob/master/03-loops/for-loop-challenge02-begin.html

- Use for-loop to count by 2
  - Change the “Move On”
  - Increment by one
    - `idx++; //ADD 1 to idx`
    - `Idx += 1; //Add 1 to idx`
      - Equivalent to `idx = idx + 1`
  - Count by 2
    - `idx += 2; // Add two to idx`
      - Equivalent to `idx = idx + 2;`

demo-for-loop-change-colors.html

https://github.com/DSHaworth/HelloCode2/blob/master/03-loops/demo-for-loop-change-colors.html

demo-for-loop-leap-years.html

https://github.com/DSHaworth/HelloCode2/blob/master/03-loops/demo-for-loop-leap-years.html

while loop

- Next most common loop
  - Condition must be true to run first time

```
start
while( stop test ){
  Everything between the {} is in the loop.
  BEST PRACTICE: ALWAYS USE {}
  move on
}
```

while loop

```
var start = 0;
var stop = 10
var idx = start;

while(idx < stop){
  console.log(idx);
  idx++;
}
```

while loop challenge 01 10 minutes

https://github.com/DSHaworth/HelloCode2/blob/master/03-loops/while-loop-challenge01-begin.html

```
while("stop test"){
  console.log(idx);
  "move on"
}
```

- Replace “stop test” and “move on” with valid JavaScript
  - See previous slide
  - Use provided variables

Do while loop

https://github.com/DSHaworth/HelloCode2/blob/master/03-loops/do-while-loop-challenge01-begin.html

```
start
do {
  Everything between the {} is in the loop.
  BEST PRACTICE: ALWAYS USE {}
  move on
}while( stop test )
```

jQuery

Interacting with HTML Input Elements

jQuery

https://www.w3schools.com/jquery/default.asp

- jQuery is a JavaScript Library.
- jQuery greatly simplifies JavaScript programming.
- jQuery is easy to learn.

https://www.w3schools.com/jquery/jquery_intro.asp

- "write less, do more", JavaScript library.
- make it much easier to use JavaScript on your website.
  - HTML/DOM manipulation
  - CSS manipulation
  - HTML event methods
  - Effects and animations
  - AJAX
  - Utilities

JavaScript vs jQuery

JavaScript

```
document.getElementById("intro")
```

```
document.getElementsByTagName("p")
```

```
document.getElementsByClassName("intro")
```

```
document.querySelectorAll(".intro")
```

jQuery

```
$("#test")
```

```
$("p")
```

```
$(".test")
```

jQuery uses CSS selectors.

As you get better with your jQuery, your CSS Selector skills will improve and so will your CSS skills in general

https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-01-adding-jquery.html

Adding jQuery

- Two main ways to add jQuery
  - Download and reference
    - Work offline

    or

  - Point to CDN *(Glicth friendly)*
    - Content Delivery Network
    - Potentially Cached

https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-02-jquery-fun.html

JavaScript vs jQuery

```
<button id="hideParagraphs">Hide Paragraphs</button>
<button onclick="showParagraphs()">Show Paragraphs</button>

<script>
$(document).ready(function(){
  $("#hideParagraphs").click(function(){
    $("p").hide();
    });
});

function showParagraphs(){
  $("p").show();
}
</script>
```

jQuery school-of-thought is to avoid mixing html with actions.

Notice **`hideParagraphs`** doesn't use an **`onclick`** attribute, nor does it point to a function.

Challenge 01 (10 minutes)

04-jQuery

jquery-challenge01-begin.html

Objective:

- Use the buttons provided
- Capture **click** events
- **Hide** to hide the demoParagraph
- **Show** to show the demoParagraph

JavaScript vs jQuery

The idea behind $(document).ready, or its **shortcut** on the right, is to prevent any jQuery code from running before the document has finished loading. When the document is done loading, it executes this function as a ***callback***.

```
$(document).ready(function(){
  $("#hideParagraphs").click(function(){
    $("p").hide();
    });
});
```

```
$(function(){
  $("#hideParagraphs").click(function(){
    $("p").hide();
  });
});

//Just removes (document).ready
```

https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-03-callback.html
https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-04-callback-a.html

Callbacks

When you see **function(){....}** like below, it's proabably a callback.

```
$(document).ready(function(){
  $("#withCallback").click(function(){
    $("p").hide("slow", function(){
      alert("This is in a callback");
    });
  });
});
```

How many callbacks are there on the left???

3

Callbacks can be replaced with function names.

https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-05-callback-b.html

Callbacks as functions

```
$(documentLoadedCallback);

function documentLoadedCallback(){
  $("#withCallback")
    .click(withCallbackClicked);
}
function withCallbackClicked(){
  $("p").hide("slow", showDone);
}
function showDone(){
  alert("The paragraph is now hidden");
}
```

Why don't we have
`$(documentLoadedCallback());` ? //WRONG

Adding () would ***invoke*** the function

Adding just the function name, sends the name of the command we want to execute.

https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-06-callback-example.html

Callback behind the scene

```
$(function(){
  $("#sayHi").click(callSayHiFunction);
});
function callSayHiFunction(){
  sayHi("Hi", sayHiDone);
}
function sayHi(greeting, callback) {
  alert(greeting);
  callback();
}
function sayHiDone(){
  alert('Finished saying "Hi"');
}
```

`sayHiDone` is the function name sent to **`sayHi`**.
`sayHi` doesn't care what the function does, it just knows it's supposed to "callback" whatever is in the "callback" once the alert is done.
To **invoke** **`callback`**, we call **`callback()`**.

callback is just the **renamed reference** to `sayHiDone`.

Ideally, you would test to make sure **callback** is a function...but that's later. :-)

https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-07-common-input.html

Forms/Data Input

Input Controls are how the user and application interact.

Common Input Types:

- input(text, password)
- radio (only one can be chosen)
- checkbox (multiple can be selected)
- textarea (lots of text)
- select (dropdown list)
    - Can also display as list

https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-08-input-values.html

jQuery input values

Get Values

```
var firstName = $("#firstName").val();
```

Set Values

```
$("#firstName").val("Duane");
```

https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-08-input-values.html

Homework

Design a form that takes each of the inputs found in **demo-08-input-values.html**.

Output the values to console.log()

Set values using your own values.

https://github.com/DSHaworth/HelloCode2/blob/master/04-jQuery/demo-09-objects.html

Objects (container for variables)

Get Values

```
var firstName = $("#firstName").val();
```

Set Values

```
$("#firstName").val("Duane");
```

Final point

A one-line description of it

Weather

Interacting with HTML Input Elements

“This is a super-important quote”

- From an expert

This is the most important takeaway that everyone has to remember.

Thanks!

Contact us:

Your Company
123 Your Street
Your City, ST 12345

no_reply@example.com
www.example.com

https://quizlet.com/448732676/hellocode2-flash-cards/

Quizlet Link to help Learn