GENERAL ASSEMBLY

JAVASCRIPT DEVELOPMENT

Sasha Vodnik, Instructor

JAVASCRIPT DEVELOPMENT

DATA TYPES & LOOPS

LEARNING OBJECTIVES

At the end of this class, you will be able to

- Describe the concept of a "data type" and how it relates to variables.
- Declare, assign to, and manipulate data stored in a variable.
- Create arrays and access values in them.
- Build iterative loops using `while`, `do/while`, `for`, and `forEach` statements.
- Iterate over and manipulate values in an array.

AGENDA

Timing	Topic
15 min	Opening
25 min	Data types
15 min	Variables
5 min	Break
60 min	Arrays
5 min	Break
20 min	Loops
15 min	Independent Practice
5 min	Forking and cloning the homework repo
15 min	Final Questions & Exit Tickets

Checkin and questions

- The **most significant thing I learned** about using the command line is ___________.
- My **biggest outstanding question** about using the command line is ___________.

Suppose a friend moved and was giving you new contact information. How would you detect an error in any of the following? (What kind of data should each contain?)

- Street address
- City
- State
- Zip
- Phone

THE DATA TYPE IDENTIFIES THE KIND OF DATA

"I just pushed my changes to the repo."

`string`

"red", "orange", "yellow", "green", "blue", "violet"

`array`

42

`number`

STRINGS

"a"

"satisfied"

"none of the above"

"Touch my hair. It’s real. (Donald Trump, June 18, 2015)"

NUMBERS

1.5

3.1415926535

27.36

45

525600

SOME LANGUAGES TREAT INTEGERS AND FLOATS AS SEPARATE TYPES, BUT NOT JAVASCRIPT

~~integers~~	~~floats~~
0	1.5
45	3.1415926535
2016	27.36
525600	

numbers

TYPEOF()

- Returns a string with the data type of the data you pass to it

ARITHMETIC OPERATORS

+	add (also concatenates strings)
–	subtract
*	multiply
/	divide
%	modulus

SPECIAL NUMBER OPERATORS

The `Math` object provides methods for additional operations

`Math.pow(m,n)`	Returns `m` to the power of `n`
`Math.sqrt(n)`	Returns the square root of `n`
`Math.random()`	Returns a random number between 0 (inclusive) and 1 (exclusive)
`Math.floor(n)`	Returns largest integer less than or equal to `n`
`Math.ceil(n)`	Returns smallest integer greater than or equal to `n`

VARIABLES

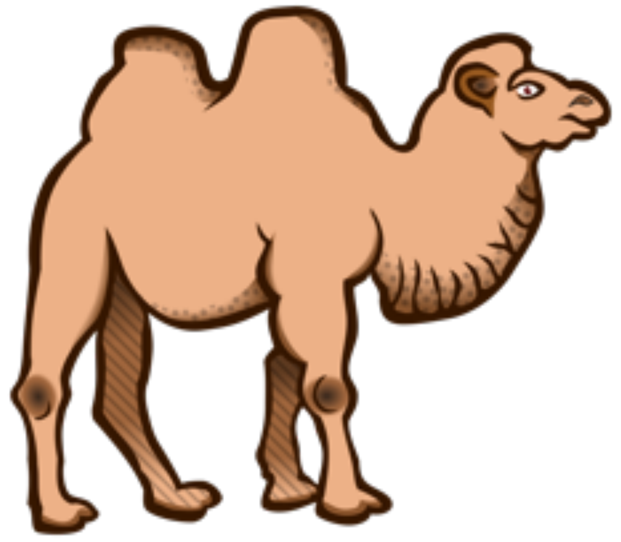
CAMEL CASE

- Use when creating a name based on multiple words
- Remove spaces, then capitalize the first letter of the second and subsequent words

camel humps

first name → `firstName`

sanitized form data → `sanitizedFormData`



KNOW YOUR EQUAL SIGNS

=	assigns value on right to object on left
===	evaluates whether values on left and right are the same

KEYWORDS FOR DECLARING VARIABLES

keyword	**when will we learn it?**
`var`	we will use `var` today
`let`	we will learn about `let` and `const` next week
`const`	

INCREMENTING AND DECREMENTING

+=	adds a number to a variable and assigns the new value to the same variable
−=	subtracts a number from a variable and assigns the new value to the same variable
++	adds 1 to a value
−−	subtracts 1 from a value

TRANSFORMING A VALUE INTO A STRING

```
toString()
```

BREAK (5 MINUTES)

ARRAYS

ARRAYS

- An **array** is a collection of data that you can use efficiently
- Similar in concept to a list
- Good for storing, enumerating, and quickly reordering data
- Each item in an array is called an **element**

ARRAY INDEX

- Each array element has a number used to reference it
- Index starts at 0
- Index ends at `length – 1`

LENGTH PROPERTY

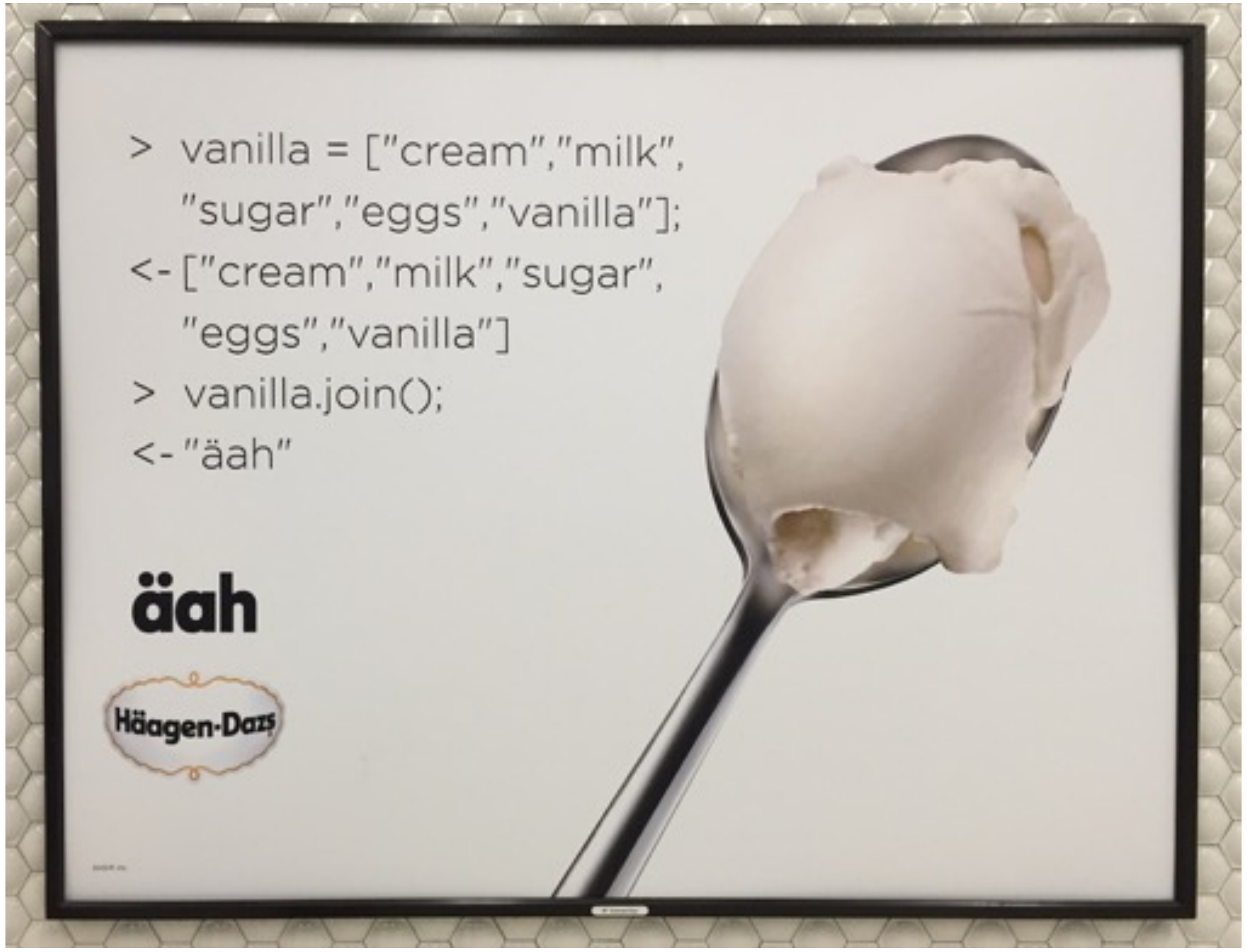
- `length` property is a number 1 greater than the final index number
- `length !==` number of elements in the array

ARRAY HELPER METHODS

ARRAY HELPER METHODS

`toString()`	Returns a single string consisting of the array elements converted to strings and separated by commas
`join()`	Same as `toString()`, but allows you to pass a custom separator as an argument
`pop()`	Removes and returns the last item
`push(item1, …, itemN)`	Adds one or more items to the end of the array
`reverse()`	Reverses the array
`shift()`	Removes and returns the first item
`unshift(item1, …, itemN)`	Adds one or more items to the start of the array

WHY IS THIS AD FUNNY?



```
>  vanilla = ["cream","milk",
   "sugar","eggs","vanilla"];
<- ["cream","milk","sugar",
   "eggs","vanilla"]
>  vanilla.join();
<- "äah"
```

BREAK (5 MINUTES)

LOOPS

`while` STATEMENT

- A loop statement that will run while a condition is `true`

```
var input = 0;

while (input < 10) {
   input++;
   console.log(input);
}
```

`do while` STATEMENT

- A loop statement similar to `while`, but that ensures that the code block is executed at least once

while

```
var input = 0;

while (input < 10) {
   input++;
   console.log(input);
}
```

do while

```
var input = 0;

do {
   input++;
   console.log(input);
} while (input < 10);
```

ITERATING

Going through the same process with a bunch of items, one at a time

`for` STATEMENT

```
var fruits = [apples, oranges, bananas];

for (var i = 0; i < fruits.length; i++) {
  console.log(i);
}
```

result in console:

```
< apples
< oranges
< bananas
```

`forEach()`

- Method specific to arrays, but similar to the `for` statement
- Lets you specify a function to execute for each array element
- We will learn all about functions in the next class
- ECMAScript 5 and later, so not supported by older browsers (IE8!)

ARRAY ITERATOR METHODS

`forEach()`	Executes a provided function once per array element
`every()`	Tests whether all elements in the array pass the test implemented by the provided function
`some()`	Tests whether some element in the array passes the text implemented by the provided function
`filter()`	Creates a new array with all elements that pass the test implemented by the provided function
`map()`	Creates a new array with the results of calling a provided function on every element in this array

QUESTIONS ON ARRAYS?

ARRAYS LAB

Next class preview: Conditionals & Functions

- Use `if/else` conditionals to control program flow based on Boolean tests.
- Use Boolean logic to combine and manipulate conditional tests.
- Differentiate among true, false, truthy, and falsy.
- Describe how parameters and arguments relate to functions
- Create and call a function that accepts parameters to solve a problem
- Define and call functions defined in terms of other functions
- Return a value from a function using the `return` keyword
- Define and call functions with argument-dependent return values

Exit Tickets!

Q&A