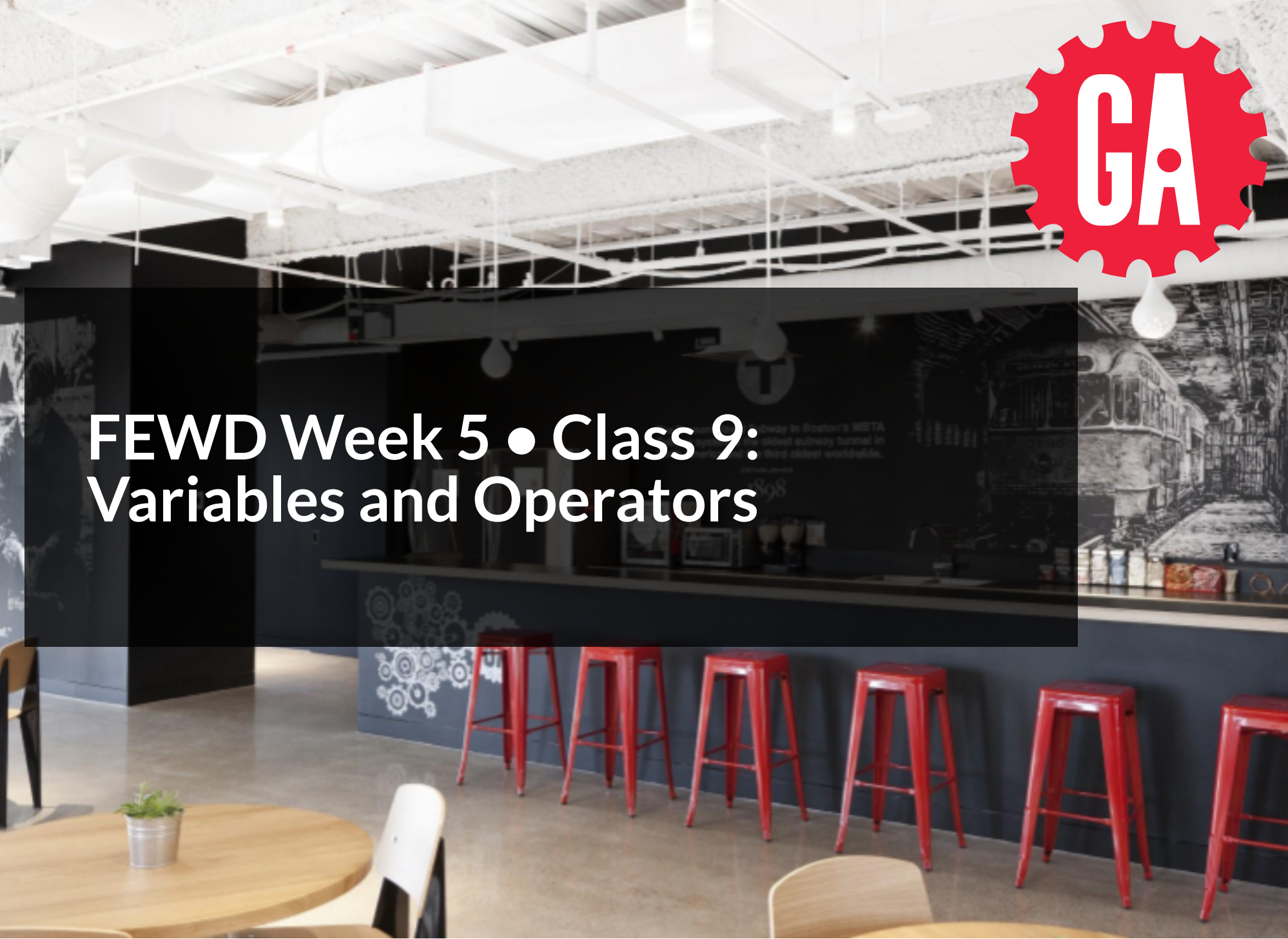

FEWD Week 5 • Class 9: Variables and Operators

Quick Review

- What does the $ mean in jQuery?
- Which tag do we use to add Javascript?
- Where do I put the Javascript on the page?
- Can someone give me an example of an event in Javascript?
- How do I write an event listener?
- Can you give me the name of a method and tell me what it does?

What We'll Cover

- What variables are and how to declare and assign them in Javascript.
- What operator are and how they are used.
- How to use the jQuery text() method to modify text in the DOM.

Variables

Objectives: Variables

- Understand what variables are
- Define and name variables
- Assign values to variables

What is a Variable?

Variables in programming are like containers used for storing pieces of data. Variables have names so that we can ***access*** them in order to add data to and retrieve data from them.

Creating Variables

- Variables are ***declared*** with `var`, `let` or `const`
- Variables names can contain: letters, numbers, the underscore (_) and the dollar sign ($), but **cannot** begin with a number.
- By convention, variables are named with lower camelCase.

```
var homeTeamScore;

var firstName;
```

Declaring Variables

- `var`: General variable, should prefer let or const
- `let`: Tells Javascript that the content of the variable is meant to be changed (called reassigned)
- `const`: Tells Javascript that the variable is a *constant*. It won't be changed! This kind of variable needs to be declared and given its value in one statement.

BEST PRACTICE: Use const unless your value is going to change. If it will change, use let.

Assigning a Value

The action of storing a piece of data in a variable is referred to as ***assigning a value*** or simply ***assignment.*** Assignment is done with an equals sign (=) in Javascript.

```
var lastName; /* Declaration */

lastName = 'Meade'; /* Assignment */

var age = 21; /* Declaration and assignment together */
```

Re-assigning Variables

The values stored in variables declared with `var` can be reassigned.

```
var age = 21; /* Declaration and assignment */

age = 'not 21'; /* Reassigned */

console.log('My age is ' + age); /* outputs: My age is not 21. */
```

Javascript Data Types

What can go in Variables?

```
age = 21;              /* Number (integers and floating point) */
height = 65.25;        /* Number */
balance = -20.66;      /* Number */

name = 'Jen';          /* String (any combination of characters) */
dogs = '2';            /* String */

tired = true;          /* Boolean (true or false) */

var book;              /* undefined (declared but not assigned)*/

tickets = null;        /* null (empty but not undefined) */
```

Creating Variables

jQuery .text() Method

The text method lets us replace the ***text*** inside of a tag.

```
var name = 'John Doe';

$('#output').text(name);
```

Score Keeper

Operators

Objectives: Operators

- Understand how data type affects how operators behave
- Be able to recognize and use various assignment, arithmetic, string and comparison operators

What are Operators?

Operators are special symbols that tell Javascript to perform specific operations.

=	*"Yo, Javascript, assign this value to this variable."*
*	*"Hey, Javascript, multiply these things!"*
>	*"Ummmm, Javascript, compare these and tell me if this one is larger than the other."*

Arithmetic Operators

With numbers, the **+**, **-**, *, and **/** operators act as expected.

```
var width = 20;
var height = 30;
var area = width * height; /* 600 */
```

HEADS UP: Area is **not a function**. It is assigned the number 600. If width or height is reassigned, area doesn't change!

Remainder Operator

The **%** does **not** mean percent. It's called the ***remainder*** operator. It gives us the remainder (as an integer) after dividing the first number by the second.

```
2 % 2;      /* 0 */
3 % 2;      /* 1 */
4 % 2;      /* 0 */
5 % 2;      /* 1 */

19 % 4;     /* 3 */
```

How could this be useful?

HEADS UP: Beware of negative numbers and cases where the first value is smaller than the second.

String Operator

When working with strings, the **+** concatenates.

```
var firstName = 'Jennifer';

var lastName = 'Meade';

var fullName = firstName + ' ' + lastName; /* "Jennifer Meade" */
```

Numbers + Strings

- If one value is a string and the other a number, the + operator concatenates them:

```
var age = 21;
var yearsOld = age + ' years old'; /* output: "21 years old" */

var a = 2;     /* number */
var b = '5';   /* string */

var c = a + b; /* "25" */
```

FYI: The term for this is coercion. Javascript is ***coercing*** the number value data type into a string.

Arithmetic on Strings

- If a string value could be a number, Javascript will coerce it into a number when performing other arithmetic operations:

```
var a = 2;     /* number */
var b = '5';   /* string */

var c = a * b; /* 10 */
```

FYI: If Javascript cannot coerce the string to a number, it returns a special value of `NaN` which stands for Not a Number.

Unary Operators

Unary operators have only one operand. The increment (++) and decrement (--) operators are unary operators you'll see a lot.

```
var a = 2;
++a;          /* the same as a = a + 1 */
              /* (2 + 1) output is: 3 */
--a;          /* the same as a = a - 1 */
              /* (3 - 1) output is: 2 */
a++ * 4;      /* run this operation with a and also increment a */
++a + 1;      /* increment a and then run this operation */
```

Converting Data Types

You can convert a string that looks like a number to a number and numbers to strings.

```
var temp = '65.67777';     /* "65.67777" */
temp = parseFloat(temp);   /* 65.67777 */
temp = parseInt(temp);     /* 65 */

var month = parseInt('12-2018'); /* 12 */

var name = 'Jen';    /* NaN */

var number = 4;
number.toString();  /* "4" */
```

Grace Hopper

Takeaways

1. Declare variables with `var`
2. Assign variables with `=`
3. Strings must be surrounded in straight quotes
4. Arithmetic operators act normally with numbers
5. The `+` concatenates values that include a string
6. Javascript will do what it can to obey you, but coercion can lead to unexpected results

Go Build Awesome Things!