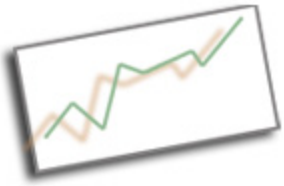

Coding and Data Skills
Dr. Cindy Royal
Texas State University
School of Journalism and Mass Communication

Introduction to Programming for Communicators

- JavaScript is a scripting language commonly implemented as part of a web browser in order to create enhanced user interfaces and dynamic websites.
- JavaScript is not the same as Java.
- Can be run client side or server-side (Node.js – a JavaScript environment).
- Foundation for code libraries like JQuery.

For the first series of exercises, we'll be practicing in Chrome's console. This is an environment that let's you run code and see the result. Later, we will be implementing JavaScript in html pages.

In Chrome, choose View, Developer Tools and click on the Console tab. Or Ctrl-click on the screen and choose Inspect Element.

Normally, these commands would be in a .js file or encapsulated in the <script> tag in an html document. But we are just practicing with concepts in this demonstration.

Your First Program
Write the string "Hello World!"

```
console.log("Hello World!");
document.write("Hello World!");
```

Don't worry if you see "undefined" in the console after the code returns. The console is attempting to determine the type that the code returns, and in most cases in our exercises that is "undefined." console.log logs to the console. document.write writes to the document. Try them both.

Note: the Chrome console doesn't require the console.log() function. You can just start typing your commands. But I wanted you to get the feel of the object.method format.

Objects, Properties and Methods
For this exercise, we will mostly be using the console object to write code to the console.

```
console.log("info goes here");
```

Later we will use the document.write method to write to the browser window of the document. We do this by calling the document object and the write method.

Basic JavaScript Syntax
End lines with semicolon ;
Put arguments/parameters in parentheses ( )
Period (.) between object and method or property.
Strings in quotation marks "

Unlike other languages, you do not have to declare a type when instantiating a variable. JavaScript can discern whether something is a text string or a number.

Some objects are built-in (i.e. document, window), some can be defined.

Data Types
String, Number, Boolean, Array, Object, Null, Undefined - programming languages have syntax and functions that work with a variety of data types.

We will mainly be concerned with strings and numbers, later adding booleans and arrays.

Length
Length is a string property that returns the length of the string.

```
Hello World!".length;
```

This returns a number that is the length of that string, including the space.

Question: *In what situations would knowing the length of a string be necessary?*

Concatenation
Use concatenation to join strings. The empty space between the quotation marks (" ") represents a space. The plus sign is the concatenation operator.

```
Hello" + " " + "Cindy!";
```

Variables
Use variables to store numbers and strings. The var prefix is optional.

```
var name = "Cindy";
Hello" + " " + name;
```

and

```
var firstname = "Cindy";
```

```
var lastname = "Royal";
firstname + " " + lastname;
```

Math and numbers
Operators and Math

Within your script tag, try the following individually. You will see the different operators for math functions.

```
3+4;
3-4;
3*4;
3/4;
3%4;
```

The "%" operator is the modulo that shows the remainder of division calculation.

Substrings
I'm using these scripts to see what my pop star name would be (like JLo), finding the first character of my first name and the first two characters of my last name.

```
var firstname = "Cindy";
var lastname = "Royal";
firstname.substring(0,1) + lastname.substring(0,2));
```

or

```
var first = "Cindy".substring(0,1);
var last = "Royal".substring(0,2);
first + last;
```

Notice that we used a combination of variables and the substring method in two different ways to achieve the same result. Often in programming, there are multiple ways to solve a problem.

Alerts and Prompts

Alert - a general warning

```
alert("Danger");
```

Confirm - answer yes or no before you can continue.

```
confirm("Are you sure?");
```

Prompt - answer anything - creates variable

```
prompt("What is your name?");

var name = prompt("What is your name?");
"Hello " + name;
```

Booleans and if statements

Use booleans to test if something is true or false. Combine with an if/else statement to define different outcomes. Notice that the "if statements" actions are delineated with curly braces. You can also test for mathematical statements like greater than >, less than < or equal to ==.

```
pw="Ilikecats63";
if(pw.length>=8){
"Password accepted";
}
else {
"Password rejected";
}
```

or with a prompt

```
pw = prompt("What is your name?");
if(pw.length>=8){
"Password accepted";
}
else {
"Password rejected";
}
```

Comparisons (= or ==)

A single equal sign (=) indicates assignment, usually used to assign a value to a variable. If you are doing a test for comparison, then you use 2 equal signs (==).

```
2+2==4;

var name="Cindy";
if(name=="Cindy")...
```

Functions
Functions allow you to define an operation and then use it later. Notice the statements in the function are enclosed in curly braces. Then we call the function using its name to execute it. There are two different ways to write a function.

```
var hello = function () {
var name = prompt("What is your name?");
"Hello " + name;
}

hello();
```

or

```
function hello() {
var name = prompt("What is your name?");
"Hello " + name;

}

hello();
```

Arguments
You can add arguments in the parentheses of a function to pass information into it.

```
var hello = function (a) {
"Hello " + a;
}

hello("Cindy");
```

or

```
function hello(a) {
"Hello " + a;
}
hello("Cindy");
```

Loops
Loops allow for iteration through a cycle. We will switch to document.write command to see the result in the browser window.

The **while loop** is a statement that uses a variable as initializer, condition and iterator.

```
var i=0;  // initializer
while (i<5) { //condition
document.write("I am writing this five times");
i++; // iterator
}
```

The **for loop** reduces some of the coding by having three statements in the parentheses, separated by a semicolon; - an initializer, a condition to test and an iterator.

```
for (var i=0; i<4; i++)
{
document.write("I am writing this 4 times<br />");
}
```

Or if you want to get fancy, you can add the iterator to have a different number in each line.

```
for (var i=0; i<4; i++)
{
document.write("This is time " + i + "<br />");
}
```

Question: *What is a situation in which a loop would be necessary?*

Arrays

Arrays allow you to store a collection of information in a variable and then access it via its index number. Index numbers start with 0. Enclose elements of an array in square brackets.

```
var favGroups = ["Old 97s", "Spoon", "Wilco"];
document.write(favGroups[1]);
```

This accesses the item in the array in location 1. Notice that it isn't the first item in the array. Arrays start with 0, so to access the first item, use favGroups[0].

You can use a loop to iterate through an array. Concatenate a break so the items appear on a different line. Your condition uses the length property of an array to determine how many items are in it.

```
var i=0;
var favGroups = ["Old 97s", "Spoon", "Wilco"];
while(i<favGroups.length){
document.write(favGroups[i] + "<br />");
i++;
```

```
}
```

Notice how we can add html items as a string. We are beginning to integrate html and JavaScript for interactivity.

Objects

Objects are similar to arrays except they store items in key/value pairs rather than accessing them by their index number. Instead of using square brackets, you use curly braces for objects.

```
var bands;

bands = {
name: "Old 97s",
city: "Dallas",
genre: "alt-country"
};

document.write(bands.name + ", " + bands.city + ", " +
bands.genre);
```

We can use an array in conjunction with an object to add multiple elements to the object.

```
var bands;

bands = [
{
name: "Old 97s",
city: "Dallas",
genre: "alt-country"
},

{
name: "Spoon",
city: "Austin",
genre: "Indie Rock"
},
{
name: "Wilco",
city: "Chicago",
genre: "alt-country"
}

];
```

```
document.write(bands[1].name + ", " + bands[1].city + ", "
+ bands[1].genre);
```

And we can loop through an object just like we did with an array. Your condition uses the length property of an object to determine how many items are in it.

```
var bands;

bands = [
{
name: "Old 97s",
city: "Dallas",
genre: "alt-country"
},
{
name: "Spoon",
city: "Austin",
genre: "Indie Rock"
},
{
name: "Wilco",
city: "Chicago",
genre: "alt-country"
}
];

for(var i=0; i<bands.length; i++) {
document.write(bands[i].name + ", " + bands[i].city + ", "
+ bands[i].genre + "<br />");

}
```

You will sometimes see data embedded in objects in JavaScript referred to as JSON (JavaScript Object Notation).

Using JavaScript in an html document

getElementById(); is a magical method that you can use to access parts of a Web page. We need to work with a web page, so that this function can manipulate the Document Object Model (DOM). The function below, when called, changes the innerHTML of the element with the id "first". This is very powerful when combined with forms to ask a user for input to change something on a page.

Let's properly structure an html document now to include the HTML5 doctype declaration, too.

```
<!DOCTYPE html>
<html>
<head>
<title>My JavaScript</title>
</head>
<body>

<p>Hello <span id="first"></span></p>

<script>

  var firstname = prompt("What is your name?");

  document.getElementById('first').innerHTML = firstname;


</script>

</body>
</html>
```

Events
An advanced use of JavaScript in an html document has to do with mouse events. An event is an interaction – onclick, onload, onmouseover, onmouseout. We'll be using an onsubmit event for our form.

Simple:

```
<!DOCTYPE html>
<html>
<head>
<title>My JavaScript</title>
</head>
<body>
<h1 onclick="this.innerHTML='Good work!'">Click on this
text!</h1>
</body>
</html>
```

The "this" operator tells the code to operate on the current element.

You will use events combined with getElementById() to create interactivity on your site.

Comments
Developers use comments to leave notes in the code, provide instructions for other developers or to provide clues to certain techniques. It helps to organize code. Comments can also be used to temporarily remove and test code without having to delete it.

With the script tag.

```
// This is a comment
/* This is a multi-line
comment */
```

Note: basic html comments are handled by <!-- -->. This would be used in html documents outside the <script> tag.

Now you have a basic understanding of programming concepts. There are many ways to go from here. You can learn more about coding Web pages with HTML/CSS, so that you can integrate interactive concepts. Or you can move into the world of JQuery to learn how to take advantage of coding libraries.

We'll move on now to make our quiz interactive.