CS 146: Intro to Web Programming and Project Development

Instructor: Iraklis Tsekourakis

Lieb 213

Email: itsekour@stevens.edu

JavaScript

JavaScript

- JavaScript is the programming language of HTML and the Web
- Programming makes computers do what you want them to do
- JavaScript is easy to learn
- Why learn JS?
  - JavaScript is one of the **3 languages** all web developers **must** learn:
    1. **HTML** to define the content of web pages
    2. **CSS** to specify the layout of web pages
    3. **JavaScript** to program the behavior of web pages
- We will talk about JavaScript, and how JavaScript works with HTML and CSS

JavaScript

- JS and Java are completely different languages, both in concept and design
- JS is a client-side web programming language
  - This means that the code is downloaded onto the user's computer and processed by the browser
  - The code is easily readable with a "View Source" of the page
- JS is object based
  - While JS has classes that can be instantiated into objects, as well as pre-defined objects, you are not expected to write classes in JS
  - In fact writing a class in JS can be pretty ugly

What can we do with JS?

- Can change HTML Content
- Can change HTML Elements
- Can change HMTL Styles (CSS)
- Can hide HTML Elements
- Can show HTML Elements

JS: Where to?

- JavaScript can be placed in the <body> and the <head> sections of an HTML page
- External JavaScript
    - External scripts are practical when the same code is used in many different web pages
    - JavaScript files have the file extension **.js**
- Finally you can play around with JS directly on the console
    - Most browsers come with developer tools that will allow you to open the console and run commands from it

The <Script> Tag

- In HTML, JavaScript code must be inserted between <script> and </script> tags

```
<script>
document.getElementById("demo").innerHTML = "My First JavaScript";
</script>
```

- Older examples may use a type attribute: <script type="text/javascript">
- This type attribute is not required; JavaScript is the *default* scripting language in HTML

JavaScript Functions and Events

- A JavaScript **function** is a block of JavaScript code, that can be executed when "asked" for
- For example, a function can be executed when an **event** occurs, like when the user clicks a button
- You can place any number of scripts in an HTML document, BUT
- **Keeping all code in one place, is always a good habit**

JavaScript in <head>

- In this example, a JavaScript function is placed in the <head> section of an HTML page
- The function is invoked (called) when a button is clicked:

```
<!DOCTYPE html>
<html>

<head>
<script>
function myFunction() {
    document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>
</head>

<body>

<h1>My Web Page</h1>

<p id="demo">A Paragraph</p>

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

</body>
</html>
```

JavaScript in <body>

- In this example, a JavaScript function is placed in the <body> section of an HTML page
- The function is invoked (called) when a button is clicked:

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

<h1>My Web Page</h1>

<p id="demo">A Paragraph</p>

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

<script>
function myFunction() {
   document.getElementById("demo").innerHTML = "Paragraph changed.";
}
</script>

</body>
</html>
```

It is a good idea to place scripts at the bottom of the <body> element. This can improve page load, because script compilation can slow down the display.

External JavaScript

myScript.js

```
function myFunction() {
   document.getElementById("demo").innerHTML = "Paragraph changed.";
}
```

- External scripts cannot contain <script> tags

External JavaScript

- To use an external script, put the name of the script file in the src (source) attribute of a <script> tag:

```
<!DOCTYPE html>
<html>
<body>
<script src="myScript.js"></script>
</body>
</html>
```

- You can place an external script reference in <head> or <body> as you like
- The script will behave as if it was located exactly where the <script> tag is located

External JS Advantages

- Placing JavaScripts in external files has some advantages:
  - It separates HTML and code
  - It makes HTML and JavaScript easier to read and maintain
  - Cached JavaScript files can speed up page loads

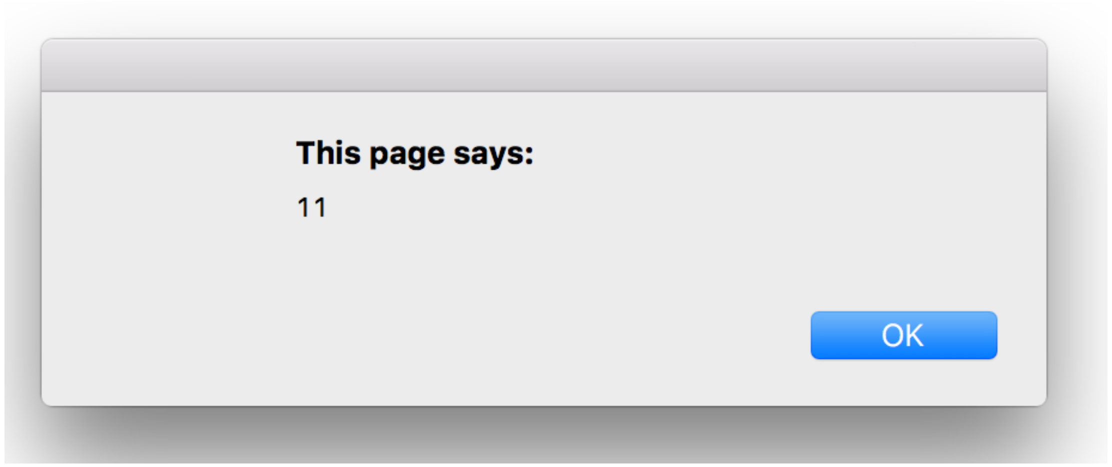
JavaScript Output

- JavaScript does NOT have any built-in print or display functions
- JavaScript can "display" data in different ways:
  - Writing into an alert box, using **window.alert()**
  - Writing into the HTML output using **document.write()**
  - Writing into an HTML element, using **innerHTML**
  - Writing into the browser console, using **console.log()**

Using window.alert()

- You can use an alert box to display data

```
<script>
window.alert(5 + 6);
</script>
```


This page says:

11

Using document.write() *(testing only)*

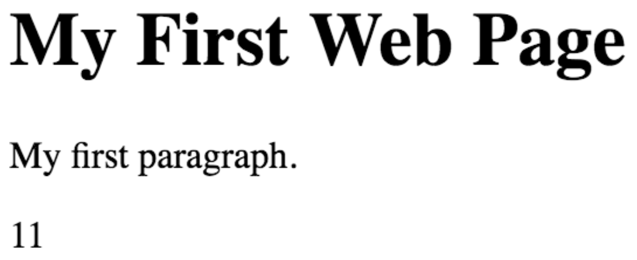
- For testing purposes, it is convenient to use **document.write()**:

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

<h1>My First Web Page</h1>
<p>My first paragraph.</p>

<script>
document.write(5 + 6);
</script>

</body>
</html>
```



My First Web Page

My first paragraph.

11

- Using document.write() after an HTML document is fully loaded, will **delete all existing HTML!**

Using InnerHTML

- To access an HTML element, JavaScript can use the **document.getElementById(id)** method.
- The **id** attribute defines the HTML element; the **innerHTML** property defines the HTML content:

```
<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = 5 + 6;
</script>
```

- To "display data" in HTML, (in most cases) you will set the value of an innerHTML property

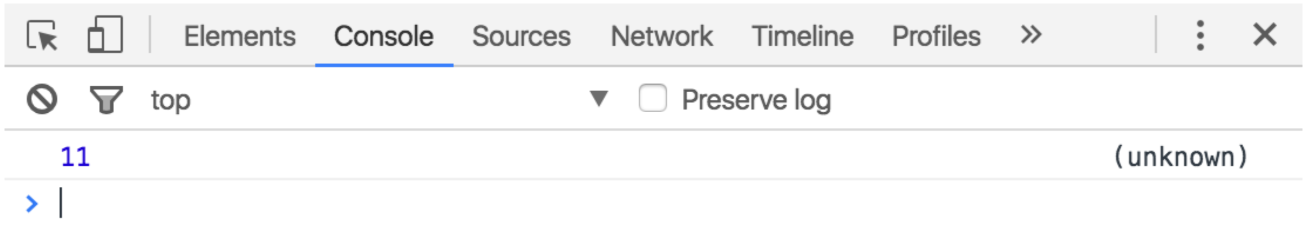
Exercise: Change HTML Content

Broboken!

Using console.log()

- In your browser, you can use the **console.log()** method to display data

```
<script>
console.log(5 + 6);
</script>
```



```
11                                                        (unknown)
> |
```

- In your browser you should be able to find the console under Developer Tools
- If you wish to distinguish different categories of output in the console, you can also use ***console.error(), console.info(), or console.warn()*** (among others)
- These will have nice little icons in front of them!

JavaScript Programs

- A **computer program** is a list of "instructions" to be "executed" by the computer
- In a programming language, these program instructions are called **statements**
- JavaScript is a **programming language**
- JavaScript statements are separated by **semicolons**

```
var x = 5;
var y = 6;
var z = x + y;
```

JavaScript Syntax

- JS statements are composed of Values, Operators, Expressions, Keywords, and Comments
- JS has two types of values: fixed values and variable values
- Fixed values are called **literals**
- Variable values are called **variables**
- Literals:
  - Numbers
  - Strings

```
10.50

1001
```

```
"John Doe"

'John Doe'
```

JavaScript Identifiers

- Identifiers are names
- In JS, identifiers are used to name variables (and keywords, and functions, and labels)
- The rules for legal names are pretty much the same in most programming languages
- In JS, the first character must be a letter, an underscore (_), or a dollar sign ($)
- Subsequent characters may be letters, digits, underscores, or dollar signs
- Numbers are not allowed as the first character; this way JS can easily distinguish identifiers from numbers

JS is Case Sensitive!

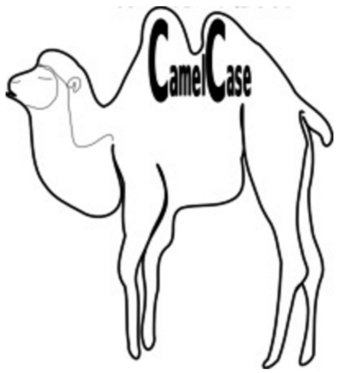
- All JavaScript identifiers are **case sensitive**
- The variables **lastName** and **lastname**, are two different variables
- Open the console and try the following:

```
lastName = "Doe";
lastname = "Peterson";
```

- JavaScript does not interpret **VAR** or **Var** as the keyword **var**

JavaScript and Camel Case

- Historically, programmers have used three ways of joining multiple words into one variable name
  - **Hyphens**: test-case
  - **Underscore**: test_case
  - **Camel Case**: TestCase



- JS programmers tend to use camel case that starts with a lowercase letter:
  - firstName, lastName, masterCard, interCity