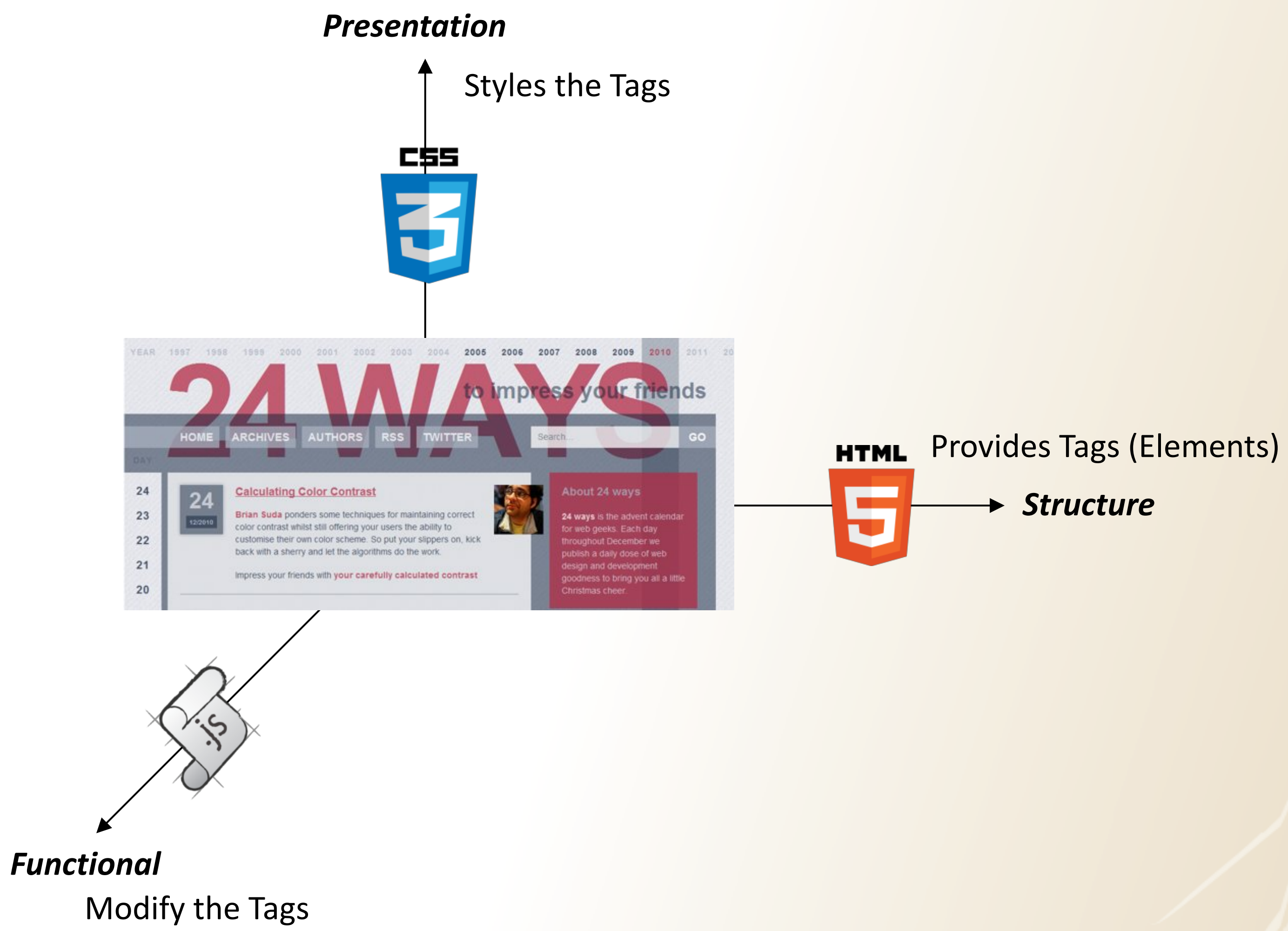
WEB-BASED SYSTEMS

IS 323

>> Introduction to JavaScript

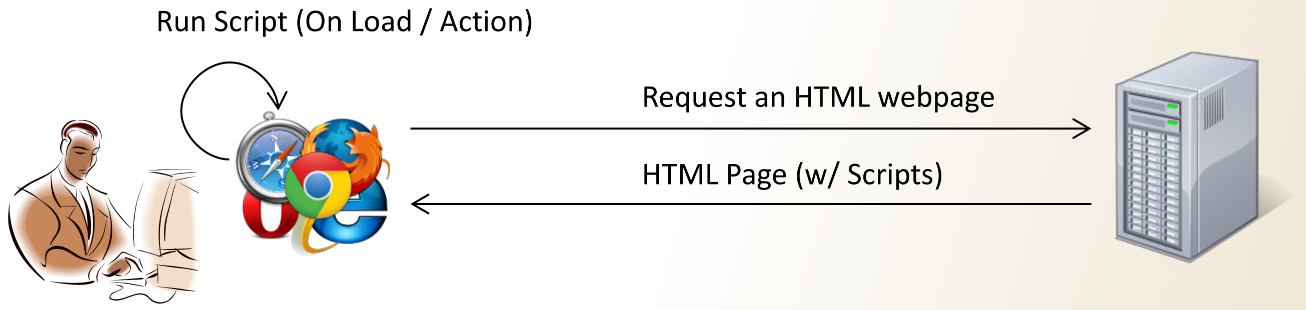
Separation of Concerns



JavaScript - Introduction

- **JavaScript**
  - Client-Side ***Scripting Language***
    - It tells the browser to go do the work
  - Makes Webpages more interactive
  - JavaScript is **not** the same as Java
    - But has various similarities with the programming language

- **Security**



Rules:

1. JS **cannot read/write** files from/to the computer file system
2. JS **cannot execute** any other programs
3. JS **cannot** establish any **connection** to other computer, except to download a new HTML page or to send mail

If No Internet Connection

- Open the browser (Chrome)
  - On a blank page
- Right Click on the page and select Inspect
- Go to the tab for Console

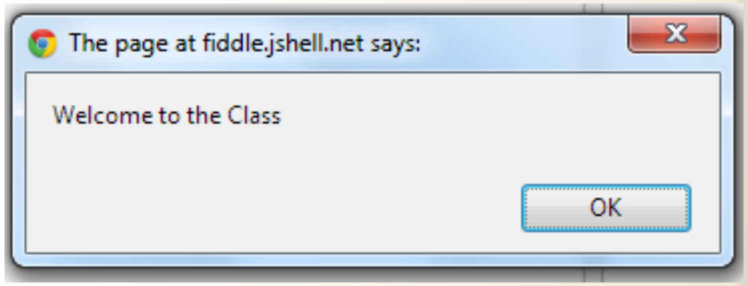
Two functions

- **Popups or Alert Boxes**
  - *alert("message")*
- **Write function**
  - *document.write()*
  - *Like the System.out.println() function in Java*

Pop-up Boxes in JS

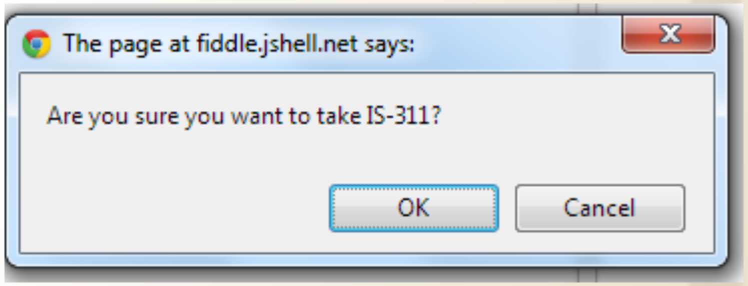
ALERT

alert("Welcome to the Class");


The page at fiddle.jshell.net says:
Welcome to the Class

CONFIRM

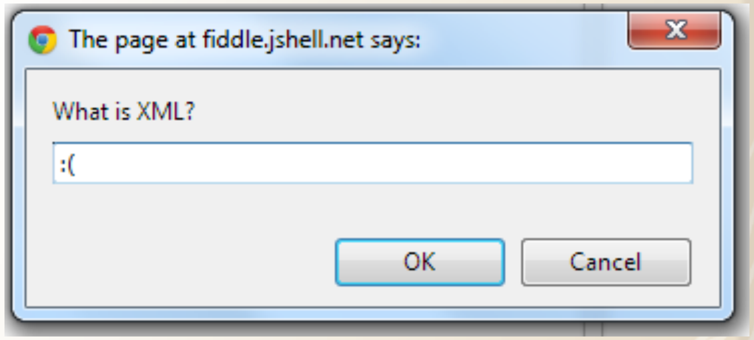
confirm("Are you sure you want to take IS-311?");


The page at fiddle.jshell.net says:
Are you sure you want to take IS-311?

PROMPT

prompt("What is XML?", ":(");

Returns a value also


The page at fiddle.jshell.net says:
What is XML?
:(

Variables

Variables or **Identifiers** are named memory locations that hold data to be used throughout the code

<u>Syntax</u>

$$\overbrace{\textbf{var}}^{\text{keyword}}\ \underbrace{\text{name}}_{\text{variable name}} = \overbrace{23}^{\text{value}};$$

Rules:

1. Case-sensitive
2. Cannot start with a number
3. Can contain letter, numbers & underscore

Note: *Must be declared before their use in the script*

TRY NOW

```
var x=23;
document.write(x);
```

Types of Variables

- **Numbers:** Integers, Decimal Numbers, Negative Numbers
- **Text / String:** “Use quotations for values”
- **Boolean:** true / false
- **No Value:** null *(Empty Variable)* – Not same as a zero

Strings in JavaScript

Quotes

You can use both **‘single quotes’** and **“double quotes”**
For eg: **var str = “This is a sample string”;**

Escape Characters

```
var x=“I said “Hi” ”;
document.write(x);
```

Use backslash (\) to escape

Concatenation

Use the “+” operator to join two strings

```
var x=“Web”;
var y= “Systems”;
document.write(x + “ “+y);
```

Strings Functions

- **.length** – Returns the length of the string

```
var test = “Hello World”;
document.write(test.length);          //Returns 11
```

- **.indexOf(substring)** – Will return the index of the substring passed in the parameter. If not found, will return (-1). It is case sensitive.

```
var test = “Hello World”;
document.write(test.indexOf(‘World’));          //Returns 6
document.write(test.indexOf(‘new’));            //Returns -1
document.write(test.indexOf(‘world’));          //Returns -1 (Case sensitive)
```

- **.charAt(index)** – Returns the character found at the index passed in the parameter. String indexes start from 0.

```
var test = “Hello World”;
document.write(test.charAt(4));          //Returns o
```

More Strings Functions

- **.substr(a, b)** – Returns the substring starting from a of length b

```
var test = “Hello World”;
document.write(test.substr(3, 2));          //Returns lo
```

- **.toLowerCase()** – converts the string to all lower case.

```
var test = “Hello World”;
document.write(test.toLowerCase());          //Returns hello world
```

- **.toUpperCase()** – converts the string to all upper case.

```
var test = “Hello World”;
document.write(test.toUpperCase());          //Returns HELLO WORLD
```

TRY NOW

```
var a = ‘Hello’;
var b = ‘World’;

document.write(a+” “+b);

document.write(“<br\>”);

document.write(a.length);

document.write(a.substring(2,4));
```

Operators

+	ADDITIONS CONCATENATION
-	SUBTRACTION
*	MULTIPLICATION
/	DIVISION
%	REMAINDER
++	INCREMENT
--	DECREMENT
=	ASSIGNMENT
==	COMPARISION
===	STRICT COMPARISION

Comparison Operators in JS

```
var a = 1;
var b = “1”;
if(a==b)    //true
```

Double equal (==) or weak comparison

- Check whether the two variables are equal
- If one is string and the other is a number, forcefully converts them both to the same type.

```
var a = 1;
var b = “1”;
if(a===b)  //false
```

Triple equal (===) or strong comparison

- Compares both the values and their data types

EXPRESSION

Statements

$$\overbrace{\text{test}}^{\text{Left Hand Side}} = \underbrace{\text{a + b}}_{\text{Right Hand Side}};$$

Evaluate the Right-Hand Side and store the value in the Left Hand Side

Assignment
Statements

```
var a = 23;
var b = 23;
test_var = a + b;
```

Conditional Statement

- **If Statement**
  - execute some code only if a specific condition is met.
- **Else If Statement**
  - Various conditions that are checked one after another until the script finds a true condition
- **Else Statement**
  - If none of the above conditions are met, this block of code is executed.

SYNTAX

$\overbrace{\textbf{if}}^{\textit{keyword}}$ (something is the case)

```
{
    more JavaScript commands
}
```

Larger than	>
Smaller than	<
Larger than or *equal* to	>=
Smaller than or *equal* to	<=
Equal to	==
Not equal to	!=

Conditional Statement

- **Switch Statement**
  - Select one of many blocks of code to be executed

SYNTAX

```
switch(test)
{
  case 1: execute code block 1
          break;
  case 2: execute code block 2
          break;
  default: default code
}
```

keywords

The condition for switch can be a "number" or a "string".

Boolean Conditions

- **Combine Multiple conditions in the IF statement**

AND (&&)	True when both elements are true
OR (||)	True when at least one of the elements is true
NOT (!)	Toggles a statement from true to false or from false to true

Looping Statement

- Initial Value; Test Condition; Update Value

- **For Statement**
  - execute some code repeatedly
- **While Statement**
  - Convenient when you want to loop until a condition changes
- **Do Statement**
  - Useful when you always want to execute the loop at least once

SYNTAX

keyword

$\underbrace{\textbf{for}}$ (initialize; condition; update)

```
{
        more JavaScript commands
}
```

Initialize outside

keyword

$\underbrace{\textbf{do}}$

```
{
        more JavaScript commands
      update inside
} while (condition);
```

Initialize outside

keyword

$\underbrace{\textbf{while}}$ (condition)

```
{
        more JavaScript commands
      update inside
}
```

JAVASCRIPT LOCATION IN HTML

JavaScript Location

```
<input type=button" onclick="alert('Hello');"/>
```

Inline

```
<script type="text/javascript">
        //Code goes here
</script>
```

Internal

```
<script type="text/javascript" src="jsfile.js"></script>
```

External

jsfile.js

JavaScript Location – Inline

<button onclick=“alert(‘Welcome’);" >Click Here</button>

Event (can be other events too like onblur ……)

Note: Cannot write longer JS statements / complete code

JavaScript Location – Internal

Optional in HTML5

```
<script type="text/javascript">

      alert('Welcome');

</script>
```

JavaScript Location inside HTML

```
<html>
   <head>
        <title>JavaScript Location</title>
         <script type="text/javascript">

         </script>
   </head>
   <body>



        <script type="text/javascript">

        </script>
    </body>
</html>
```

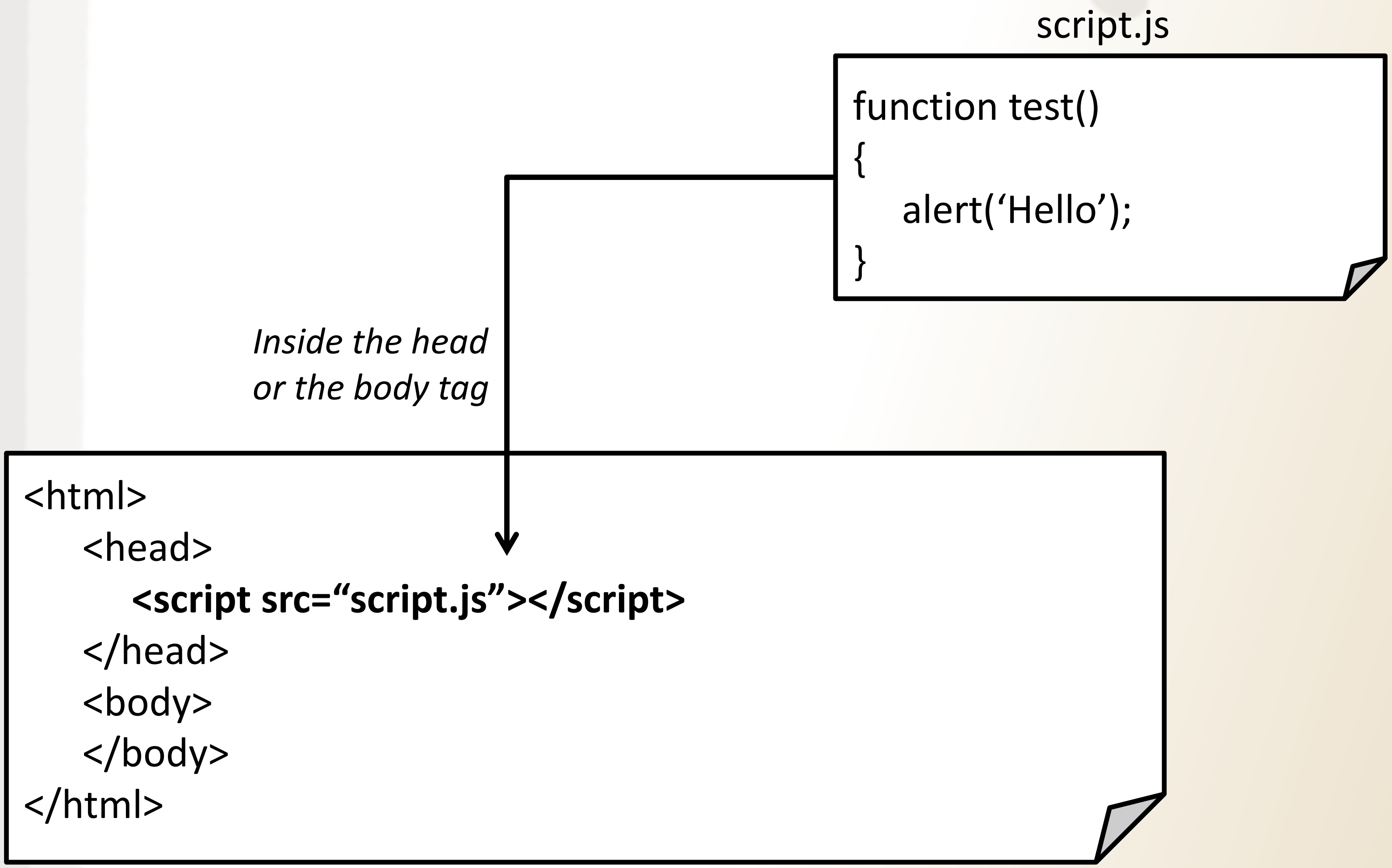
In the Head

Functions are loaded before the buttons, links or other things that call them are loaded

In the Body

Functions that needs running after the whole page (body) of the HTML is loaded http://www.kfu.edu.sa/ccsit/is/web-based-systems

JavaScript Location – External



```
function test()
{
    alert('Hello');
}
```

```
<html>
    <head>
        <script src="script.js"></script>
    </head>
    <body>
    </body>
</html>
```

Summary

- Variables
- Data Types
- String Functions
- Operators
- Statements
  - Assignment Statements
  - Conditional Statements (if, else, switch)
  - Looping Statements (for, while, do-while)
- JavaScript Location in HTML