FEU INSTITUTE OF TECHNOLOGY
COLLEGE OF ENGINEERING • COLLEGE OF COMPUTER STUDIES

Web Design Fundamentals / Basic Web Design

JavaScript

FEU INSTITUTE OF TECHNOLOGY
COLLEGE OF ENGINEERING • COLLEGE OF COMPUTER STUDIES

JavaScript Intro

Advanced Web Design

What you should know

- HTML
- CSS

What you should know

You don't need to be a programmer

Though it's a great help if you are.

HTML

```
<html>
  <head>
     <title>Simple Page</title>
  </head>
  <body>
    <div id="main">
       <h1>Today's Headline</h1>
       <p>Regular paragraph text goes here.</p>
    </div>
  </body>
</html>
```

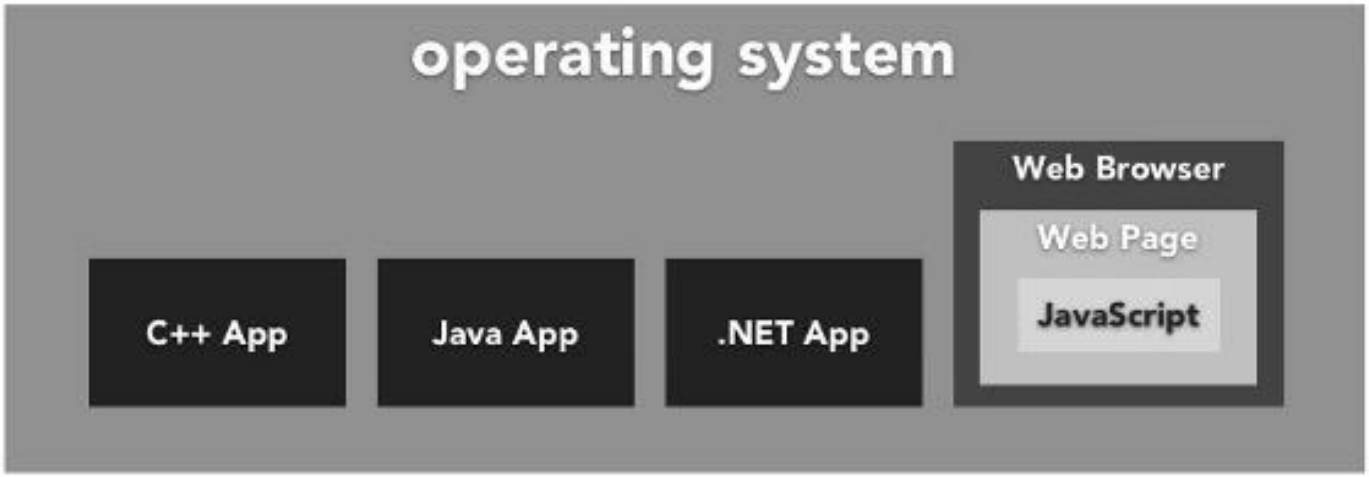
Connect

```
p {
    color:#333;
    font-size:1.2em;
}

#main {
    width:90%;
    background-color:#eee;
}
```

JavaScript: Introduction

What is scripting language?

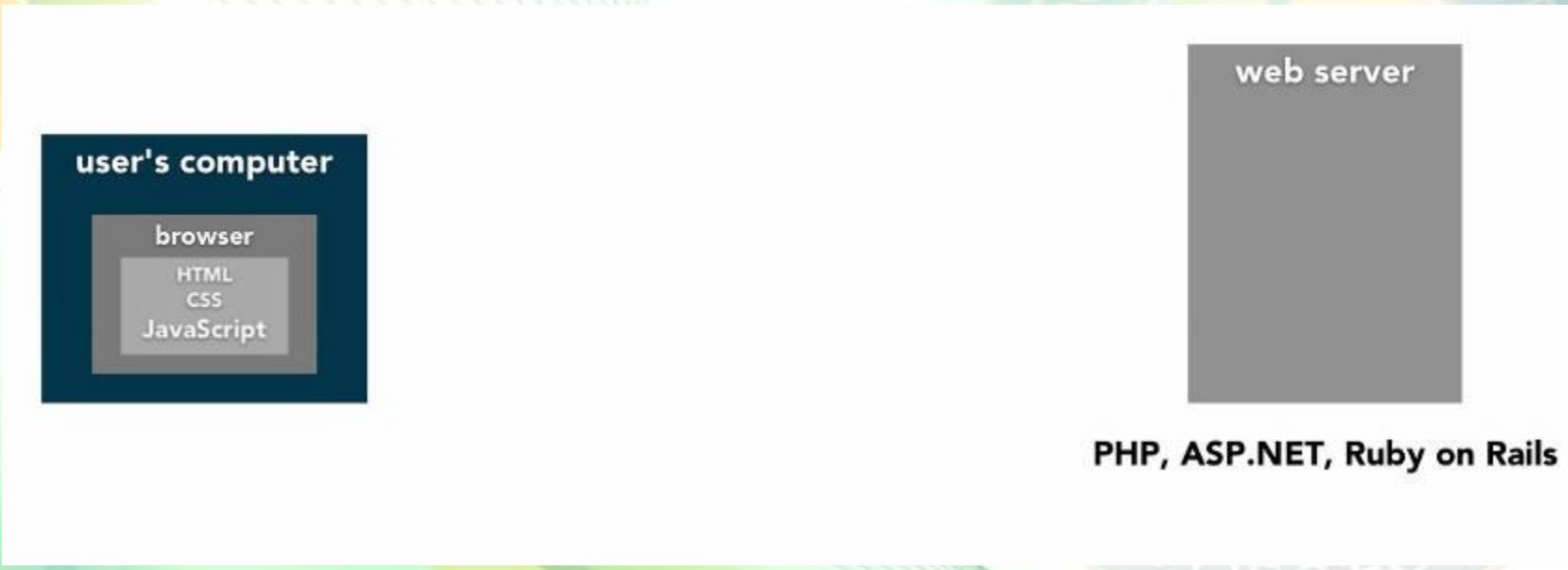


can't access local files

can't directly access database

can't access hardware (USB, etc)

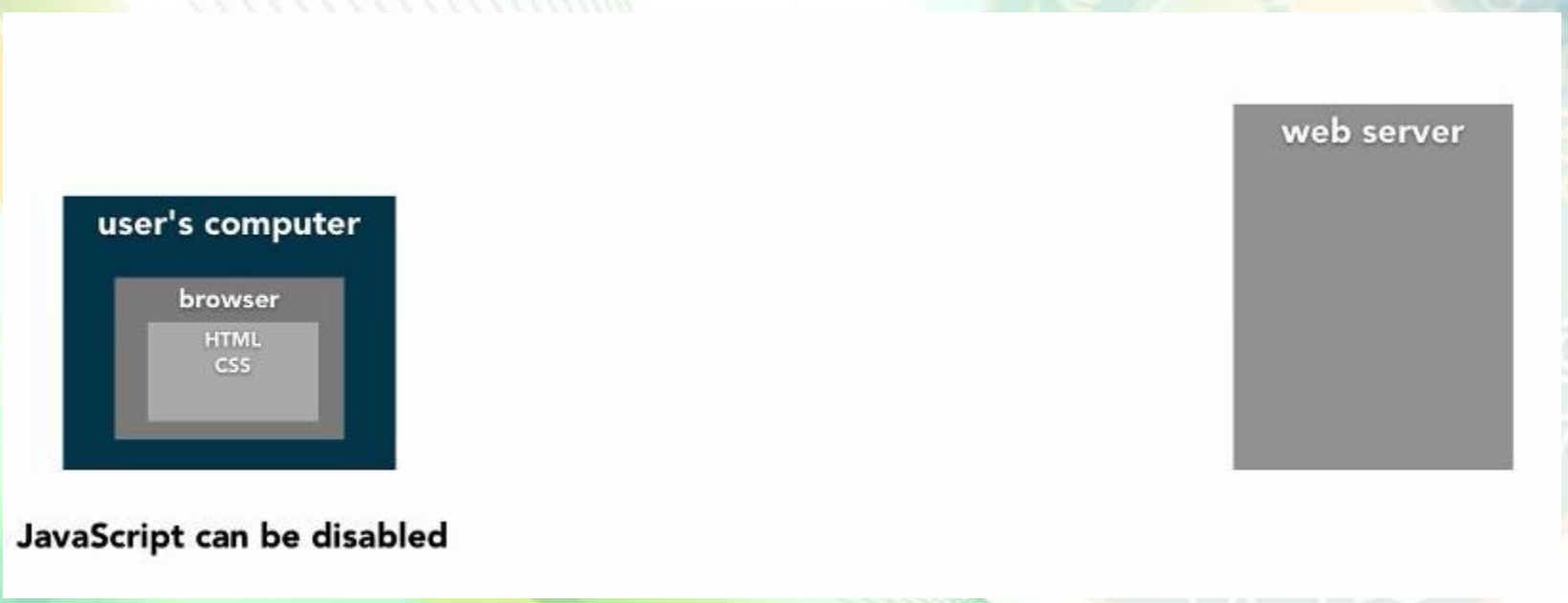
JS: Client-side language



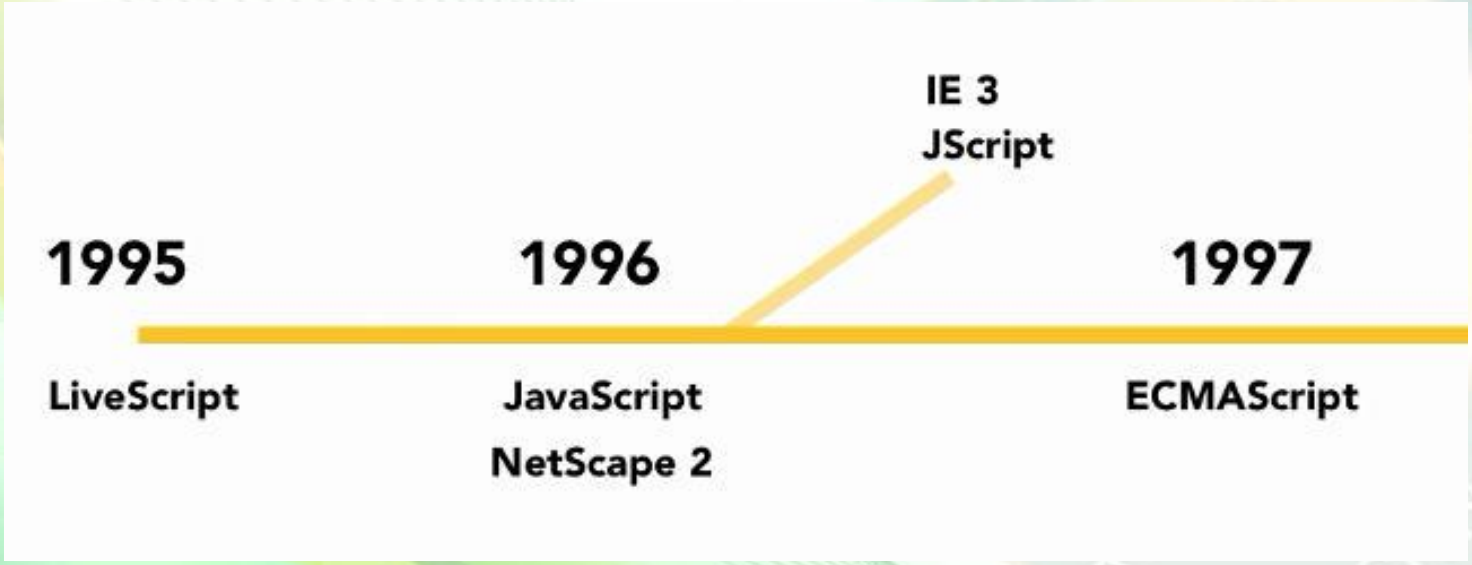
What is Javascript (JS)?

- One of the major web technology
- JavaScript was designed to add dynamic interactivity to HTML pages
- JavaScript is a scripting language
- A scripting language is a lightweight programming language
- Developed by **Brendan Eich,** co founder of Mozilla project

Issues with a client-side language



A bit History



A bit History

1999		2009
ECMAScript 3	Internet Explorer Firefox Chrome Opera Safari (etc.)	ECMAScript 5 published

What do you need to write JS?

You need text editor

Are Java and Javascript the same?

- NO!
- Java and JavaScript are two completely different languages in both concept and design!
- Java (developed by Sun Microsystems) is a powerful and much more complex programming language - in the same category as C and C++.

What can a Javascript do?

- **JavaScript gives HTML designers a programming tool** - HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax! Almost anyone can put small "snippets" of code into their HTML pages.

What can a Javascript do?

- **JavaScript can react to events -** A JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element

- **JavaScript can read and write HTML elements -** A JavaScript can read and change the content of an HTML element.

What can a Javascript do?

- JavaScript can be used to validate data
- JavaScript can be used to detect the visitor's browser

Javascript Structure 1

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Hello</title>
    </head>
    <body>
      <div>
          <p>regular body text</p>
      </div>
      <script>
          alert("hello");
      </script>
    </body>
</html>
```

Javascript Structure 2

```html
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>Untitled Document</title>
</head>

<body>
<h1></h1>
<script src="myscript.js"> </script>
</body>
</html>
```

index.html

```javascript
// JavaScript Document
var myHeading = document.querySelector('h1');
myHeading.innerHTML = 'Hello world!';
```

myscript.js

Javascript Where to!

- JavaScripts can be put in the body and in the head sections of an HTML page.

(observe the difference)

- JavaScripts in a page will be executed immediately while the page loads into the browser.

JS is interpreted not compiled

```
int main (int argc, const char * argv[])
{
    NSAutoreleasePool * pool =
       [[NSAutoreleasePool alloc] init];
    int foo = 99;
    foo++;
    for (int i =  1; i < foo; i++) {
        simpleFunction(i);
    }
    [pool drain];
    return 0;
}
```

C, C++, Objective-C, C#, etc.

JS is interpreted not compiled

```
<!DOCTYPE html>
<html lang="en">
      <head>
          <title>Hello</title>
      </head>
      <body>
          <div>
              <p>regular body text</p>
          </div>
          <script>
            alert("hello");
          </script>
      </body>
</html>
```

JavaScript is case SENSITIVE!

Javascript vs HTML

```
<html>
  <head>
     <title>Simple Page</title>
  </head>
  <body>
    <div id="main">
       <h1>Today's Headline</h1>
       <p>Regular paragraph text goes here.</P>
    </div>
  </body>
</html>
```

Not sensitive!

Javascript vs HTML

```
alert("Hello world");
Alert("Hello world");
```

Sensitive!

JS: Whitespace insensitive

```
alert("Hello world");
alert("Another message");
```

same

```
alert("Hello world");
alert  (  "Hello world"  )  ;
```

Best practices

```
alert("Hello world"); alert("Another message");
```

```
alert("Hello world");
alert("Another message");
```

JavaScript Comments

```
// this is a comment
alert("hello"); // can go here

/* this is a
   multiple
   line
   comment
*/
```

JavaScript Comments

```
// this is a comment
alert("hello"); // can go here

/* this is a
   multiple
   line
   comment
*/
```

Sample

1.) Embedding script in HTML

2.) Script Positioning

scriptPosition.html

```
<html>
<head>
<title>Simple Page</title>
    <script>
        alert("Hello, world!");
    </script>

</head>
<body>
    <h1>Simple HTML Page</h1>
    <p>
    This is a very simple HTML page.
    </p>

    <p>It's about as basic as they come. It has:<p>

    <ul>
        <li>An H1 Tag</li>
        <li>Two paragraphs</li>
        <li>An unordered list</li>
    </ul>


</body>
</html>
```

Sample

scriptPosition.html

```
<html>
<head>
<title>Simple Page</title>
    <script>
        alert("Hello, world!");
    </script>

</head>
<body>
    <h1>Simple HTML Page</h1>
    <p>
    This is a very simple HTML page.
    </p>

    <p>It's about as basic as they come. It has:<p>

    <ul>
        <li>An H1 Tag</li>
        <li>Two paragraphs</li>
        <li>An unordered list</li>
    </ul>


</body>
</html>
```

scriptPosition.html

```
<html>
<head>
<title>Simple Page</title>


</head>
<body>
    <h1>Simple HTML Page</h1>
    <p>
    This is a very simple HTML page.
    </p>

    <p>It's about as basic as they come. It has:<p>

    <ul>
        <li>An H1 Tag</li>
        <li>Two paragraphs</li>
        <li>An unordered list</li>
    </ul>

    <script>
        alert("Hello, world!");
    </script>
</body>
</html>
```

JS BASICS

variable

Variables are containers that you can store values in. You start by declaring a variable with the var keyword, followed by any name you want to call it

var myVariable;

After declaring a variable, you can give it a value:

myVariable = ”van”;

You can do both these operations on the same line if you wish:

var myVariable = ”van”;

The Lifetime of JavaScript Variables

- If you declare a variable within a function, the variable can only be accessed within that function. When you exit the function, the variable is destroyed. These variables are called local variables. You can have **local variables** with the same name in different functions, because each is recognized only by the function in which it is declared.

- If you declare a **variable outside a function**, all the functions on your page can access it. The lifetime of these variables starts when they are declared, and ends when the page is closed.

JavaScript Popup Boxes (Alert)

```
<!doctype html>
<html>
<head>
<script type="text/javascript">

 function show_alert()  {
    alert("I am an alert box!");
                          }

</script>
</head>
 <body> <input type="button" onclick="show_alert()" value="Show
alert box" /> </body>
</html>
```

JavaScript Popup Boxes (Confirm)

- A **confirm box** is often used if you want the user to verify or accept something.
- When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed.
- If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

JavaScript Popup Boxes (Confirm)

```
<html>
<body>
<p>Sample for confirm box</p>
<button onclick="myFunction()">Try it</button>
<script>
function myFunction() {
        if (confirm("Press a button!") == true) {
         alert("You pressed OK!");
    } else {
         alert("You pressed Cancel!");
    }
   }
</script>
</body>
</html>
```

@ FOR EMAIL

Note:

If you enter an email address without the @, you'll get an alert asking you to re-enter the data.

What is:x.indexOf(@)==-1? This is a method that JavaScript can search every character within a string and look for what we want. If it finds it will return the position of the char within the string.

If it doesn't, it will return -1. Therefore, x.indexOf("@")==-1 basically means: "if the string doesn't include @

@ FOR EMAIL

```
<html>
<head>
<script>
function emailchk()
{
var x=document.feedback.email.value
if (x.indexOf("@")==-1)
{
     alert("It seems you entered an invalid email address.")
     }
}
</script>
     </head>
<body>
<form name="feedback">
Email:
<input type="text" size="20" name="email" onBlur="emailchk()">
<br>
<input type="submit" value="Submit">
</form>
</body></html>
```

JavaScript Prompt (User Input)

```
var myInput = prompt(“Enter a number: ");
alert(myInput==“2” ? “Your choice is
   number two" : "Input again");
```

JavaScript Functions

- A **function** will be executed by an event or by a call to the function.
- To keep the browser from executing a script when the page loads, you can put your script into a function.

JavaScript Functions (Sample)

```
<html>
<head>
<script type="text/javascript">
function displaymessage()
{
alert("Hello World!");
}
</script>
</head>

<body>
<form>
<input type="button" value="Click me!" onclick="displaymessage()" />
</form>
</body>
</html>
```

The return Statement

- The return statement is used to specify the value that is returned from the function.
- So, functions that are going to return a value must use the return statement.
- **The example below returns the product of two numbers (a and b):**

The return Statement

```
<html>
<head>
<script type="text/javascript">
function product(a,b)
{
return a*b;
}
</script>
</head>

<body>
<script type="text/javascript">
document.write(product(4,3));
</script>

</body>
</html>
```

JavaScript For Loop

- Loops execute a block of code a specified number of times, or while a specified condition is true.

- In JavaScript, there are two different kind of loops:
  - **for** - loops through a block of code a specified number of times
  - **while** - loops through a block of code while a specified condition is true

Syntax

```
for (var=startvalue;var<=endvalue;var=var+increment)
{
code to be executed
}
```

For Loop

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>Untitled Document</title>
</head>

<body>

<script type="text/javascript">
var i=0; for (i=0;i<=5;i++) {
document.write("The number is " + i);
document.write("<br />"); }
</script>

</body>
 </html>
```

JavaScript While Loop

- The while loop loops through a block of code while a specified condition is true.

Syntax

```
while (var<=endvalue)
    {
    code to be executed
    }
```

While Loop Example

```
<html>
<body>
<script type="text/javascript">
var i=0;
while (i<=5)
  {
  document.write("The number is " + i);
  document.write("<br />");
  i++;
  }
</script>
</body>
</html>
```

The do...while Loop

- The do...while loop is a variant of the while loop. This loop will execute the block of code ONCE, and then it will repeat the loop as long as the specified condition is true.

Syntax

```
do
    {
    code to be executed
    }
  while (var<=endvalue);
```

The do...while Loop

```
<html>
<body>
<script type="text/javascript">
var i=0;
do
 {
 document.write("The number is " + i);
 document.write("<br />");
 i++;
 }
while (i<=5);
</script>
</body>
</html>
```

The break Statement

The break statement will break the loop and continue executing the code that follows after the loop (if any).

```
 <html>
<body>
<script type="text/javascript">
var i=0;
for (i=0;i<=10;i++)
 {
 if (i==3)
   {
   break;
   }
 document.write("The number is " + i);
 document.write("<br />");
 }
</script>
</body>
</html>
```

The continue Statement

The continue statement will break the current loop and continue with the next value. The break statement will break the loop and continue executing the code that follows after the loop (if any).

```
<html>
<body>
<script type="text/javascript">
var i=0
for (i=0;i<=10;i++)
 {
 if (i==3)
  {
  continue;
  }
 document.write("The number is " + i);
 document.write("<br />");
 }
</script>
</body>
</html>
```