Programming for Digital Media EE1707

JavaScript

By: A. Mousavi & P. Broomhead
SERG, School of Engineering Design, Brunel University, UK

References and Sources

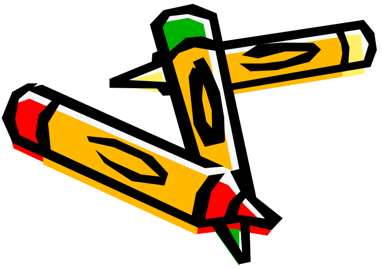
1. Javascript & JQuery: interactive front-end Web development - Jon Duckett, Gilles Ruppert, Jack Moore c2014
2. Understanding ECMAScript 6: The Definitive Guide for JavaScript Developers 30 Sept. 2016

3. DOM scripting: web design with JavaScript and the Document Object Model - Jeremy Keith c2010

4. Lecture notes and other supporting material on

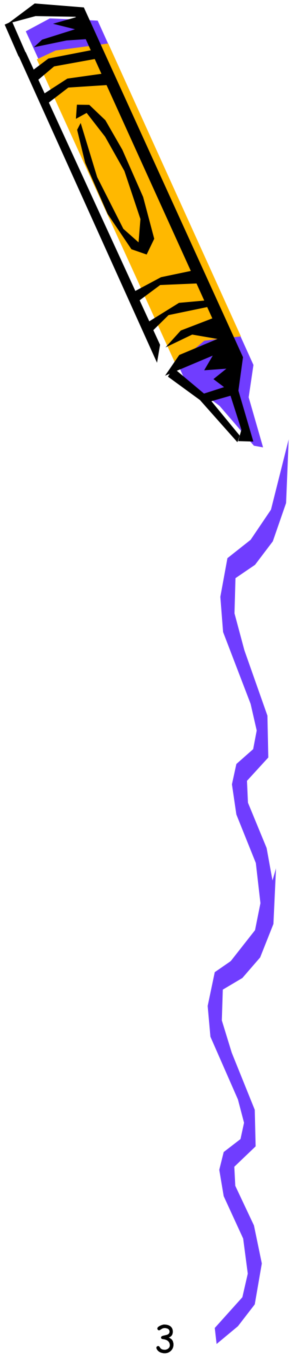
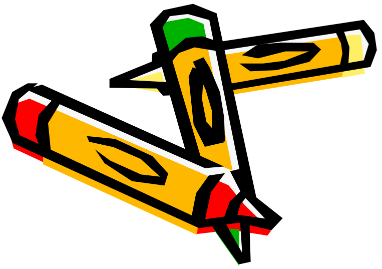
   http://people.brunel.ac.uk/~emstaam/

   The Blackboard

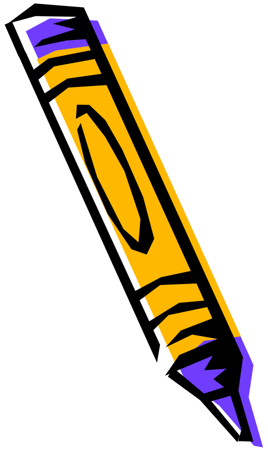


Content of lectures

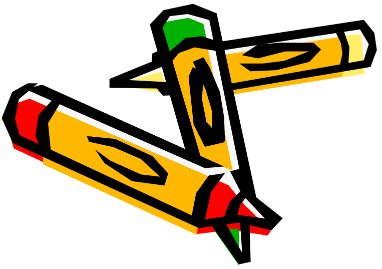
1. **Introduction to JavaScript:**
   - Brief History
   - Client-Side and Server-Side JavaScript
   - What is Data Object Model (DOM)
   - Dynamic HTML and Browsers
   - Some elementary concepts and JavaScript Syntax
2. **Java Script Syntax**
   - Statements
   - Variables and arrays
   - Operators
   - Conditional and looping statements
   - Functions and objects

Content of lectures continued



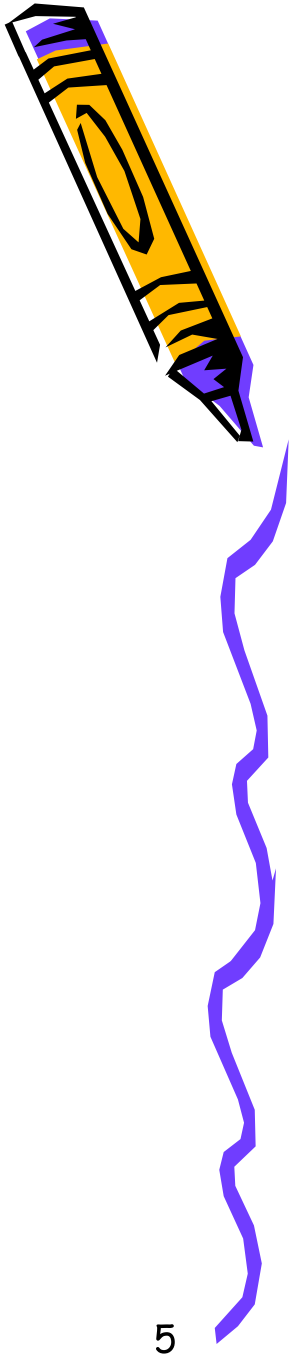
3. **Objects and DOM**
   - Built-in objects, browser object models
   - Document Object Model (DOM)
   - Client-Side validation of input
   - Programming with JavaScript – What can we do to date?
4. **Event Handling**
   - Event Handlers
   - Image objects
5. **Forms**
   - Manipulating fields
   - Interactive Forms

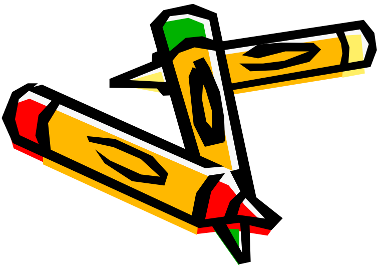


Content of lectures continued

7. Objects/JSON

8. AJAX

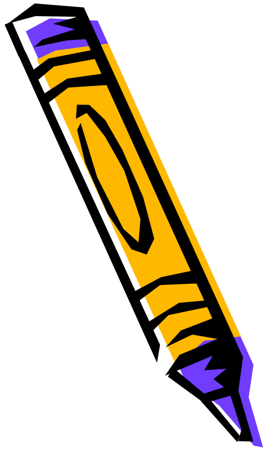


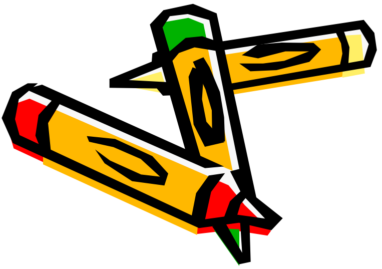


Introduction to JavaScript

- **Today's topics:**

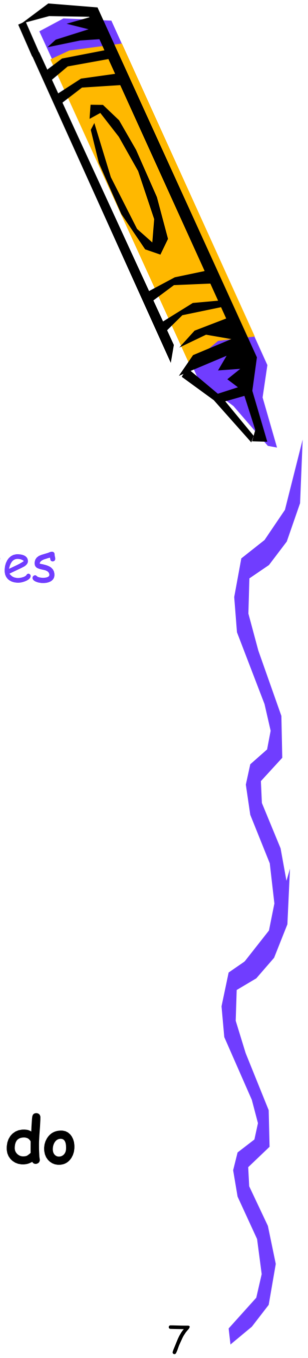
1. Brief History
2. Client-Side and Server-Side JavaScript
3. What is Document Object Model (DOM)
4. Dynamic HTML and Browsers
5. Some elementary concepts and JavaScript Syntax

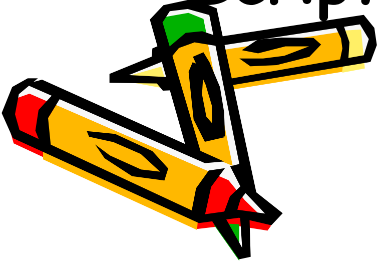


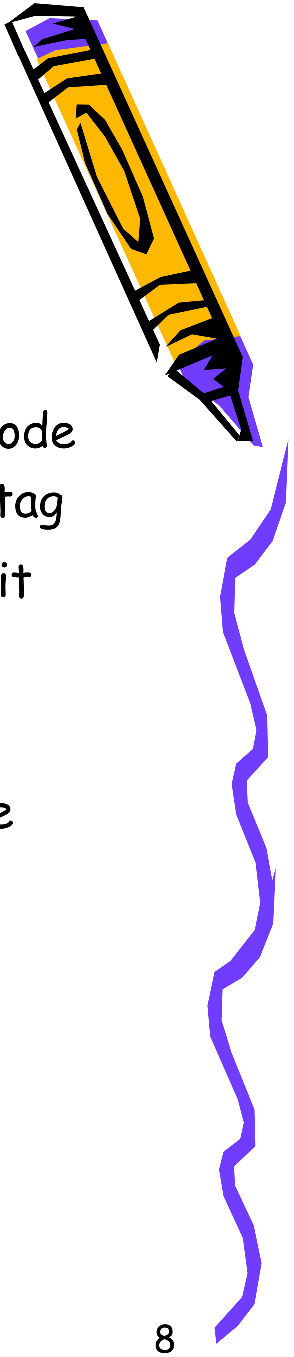


JavaScript History

- **Was developed by Netscape and Sun Microsystems**
  - Not *Java*
  - To make websites more interesting and dynamic
  - To interact with server side applications and Databases
  - VBScript and Jscript by Microsoft
- **Collaboration with European Computer Manufacturers Association (ECMA) – *ECMAScript***
- **Scripting language tells the browser what to do**

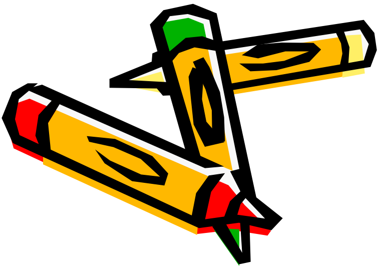




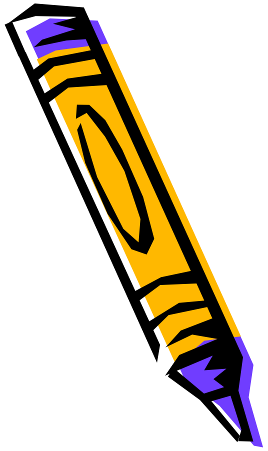


Facts

- Can be created using any text editor
- Case sensitive
- The code can be included in an HTML document using <script></script> tags but it is better to put your JavaScript code in a separate file *filename.js* and then call it with the <script> tag
- Note that the target Browser supports JavaScript otherwise it will be ignored. Ex: <script language="JavaScript1.6">
- Use the body to trigger events that calls specified functions
- JavaScript used to be included in the <head> - but not anymore
- The Head section always loads first and then the <body> can safely refer to it
- Interpreted codes and compiled languages



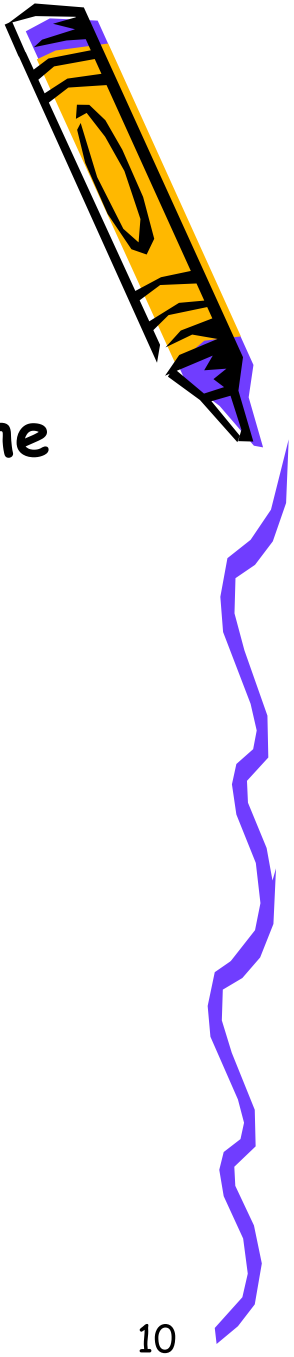
Browser Object Model



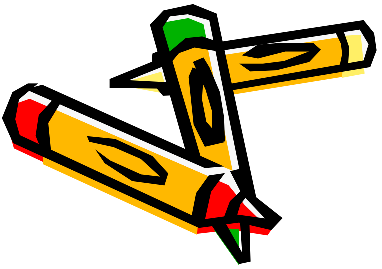
- JavaScript allows you to control how data is presented in a browser

- Things like:
  - The properties of browser window can be manipulated (e.g. height, width, position)

Document Object Model (DOM)



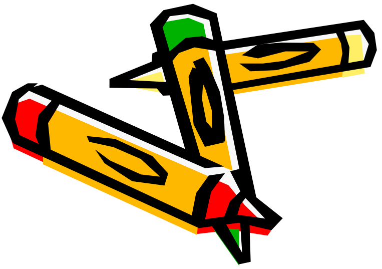
- By using DOM we can manage and describe the contents of a document
- Objects contain *properties and methods*
- It allows you to define “What it is” – image, form etc.



Dynamic HTML

- **Integration of HTML, Cascading Style Sheets (CSS) and JavaScript**
- **Principles of DHTML:**
  - use HTML to mark up web page into elements
  - use CSS to arrange and manage those elements
  - use JavaScript to dynamically change the appearance and styles
- **Using DHTML you could suddenly create complex animation effects**

Clash of Browsers!!!



Example

Define an element:

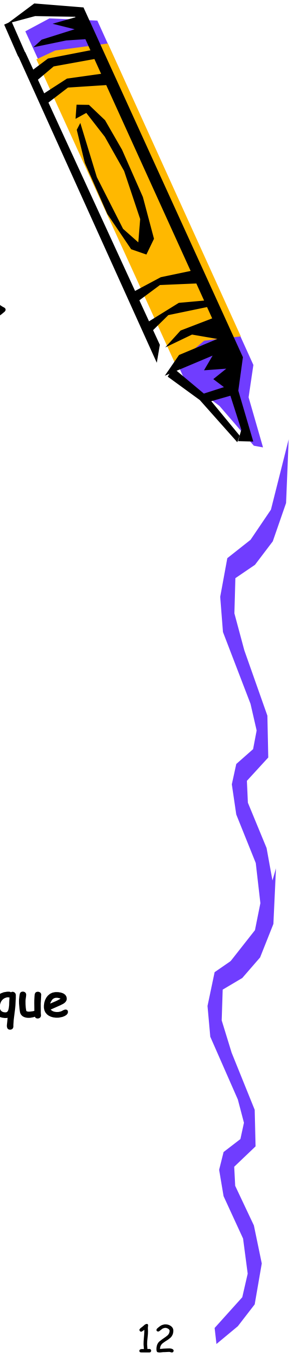
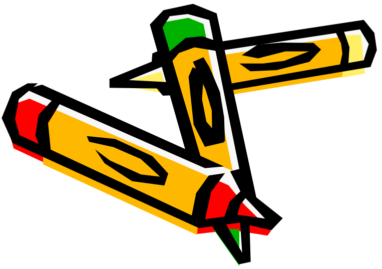
```
<div id= "anelement"> This is an element </div>
```

Then use the CSS to apply positioning:

```
#anelement   {
               position: absolute;
               left: 50px
                top: 100px
               }
```

Use DOM property element called *layers*. The *layers* having unique ID. And then using JavaScript

```
document.layers['anelement']
```

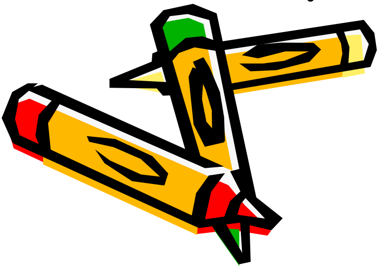
Cascade Style Sheets and JavaScript

How a content of a page is represented can be defined by style sheets. The contents of a web page is influenced by style sheets through tags and elements.

Cascading Style Sheets (CSS) & CSS-DOM

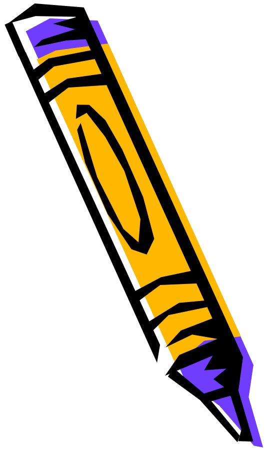
Is a style sheet mechanism that has been specifically developed to meet the needs of Web designers and users. A CSS file can be created and edited manually with a text editor, but one can also write a program in a scripting language i.e. JavaScript that manipulates a style sheet.

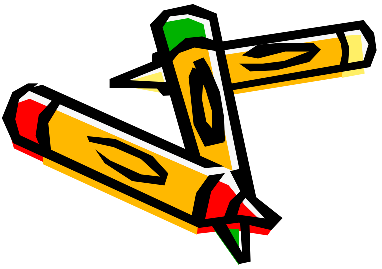
The CSS Document Object Model is an API for manipulating CSS from within the program.



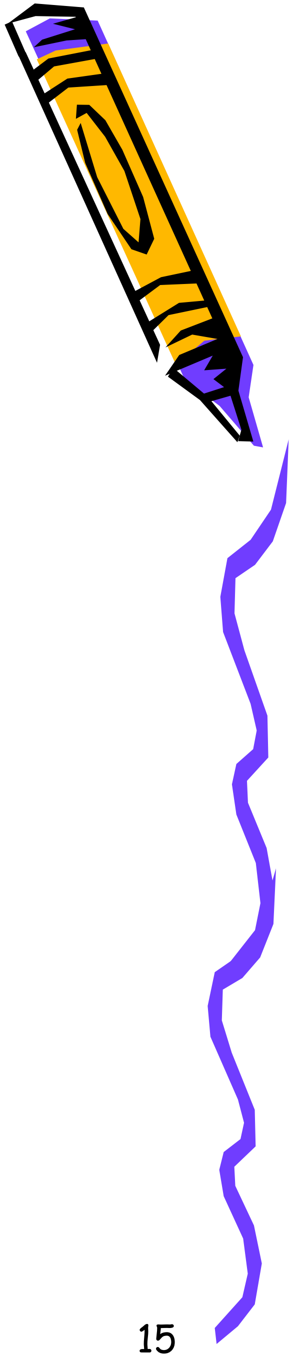
Clash of Browsers

- A struggle between various browsers
- W3C involvement
- DOM standardisation in-built in the latest browsers

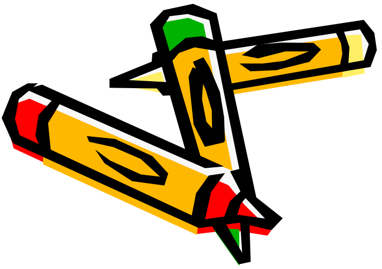




Client-Side and Serve-Side apps



- **Client-Side**
  - What happens on your browser
- **Server-Side**
  - What happens on the remote/local server

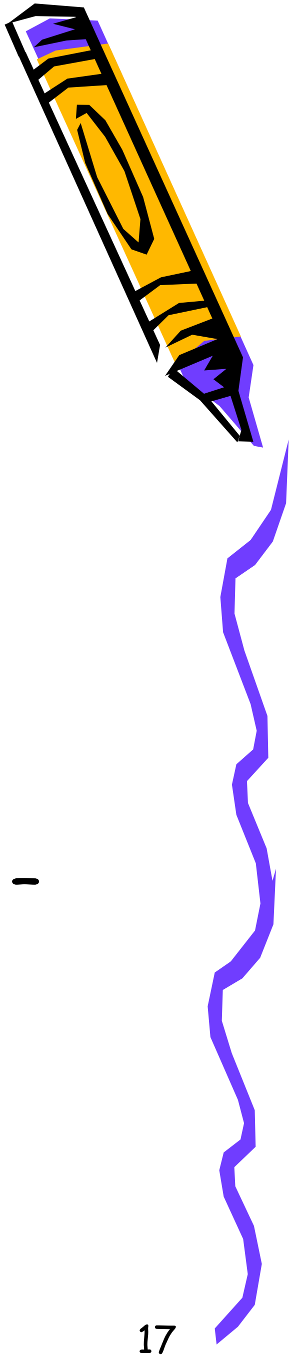


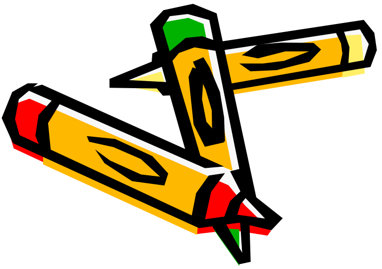
Client-Side JavaScript

1. **Generate and modify HTML content on-fly**
2. **Lively interaction with the user:**
   - Multimedia, images, animation (image manipulation), links
   - Pop-up Prompts, alerts and dialog boxes
   - Interactive forms with field validation on client-side
3. **Deal with Cookies – reset and store**
4. **Simple computations**
5. **Manage embedded Java applets and plug-ins**

Limits of Client-Side JavaScript

1. Limitations in directly drawing graphics
2. Does not really interact with the underlying file systems and OS
3. Cannot support multithreading
4. Unable to use arbitrary network connections – but able to load documents from arbitrary URLs





Server-Side JavaScript

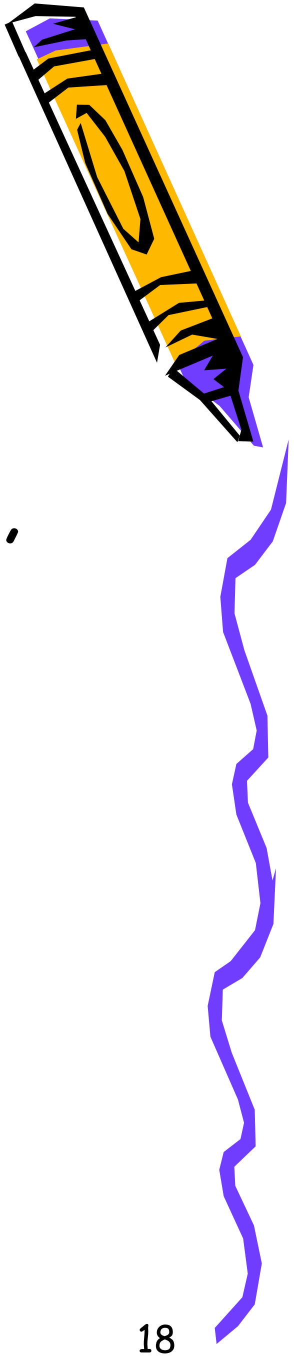
1. Extends functionality for customised web applications

2. Interaction with database systems via ODBC, JDBC,…

3. Read and write server files

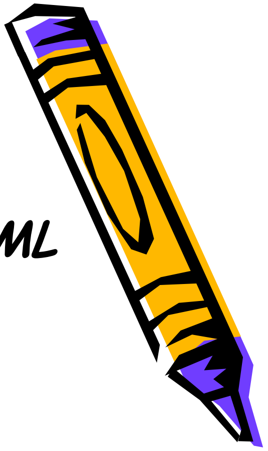
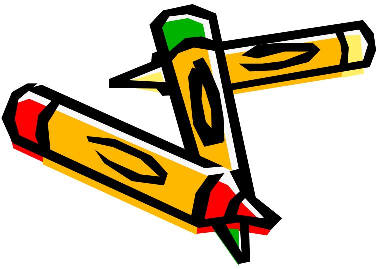
4. Invoke programs residing in the server

5. Establish arbitrary network connections



Short example 1

```
<!-- Document : Test 1 Author : emstaam --> <!DOCTYPE HTML
   PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title> Test 1 </title>
<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>
```

Short example 2

Create a file in the same directory as your html file. Call it alertbox.js.

Type in:

```
function show_alert( ) {
  alert (" I am an Alert Box!");
  }
```

and save.

In your html file write your code:

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

What is the difference between example 1 and 2?