

# JAVASCRIPT IV

IT1301 UX Design

Module Leader: Ms Grace Chan

# LEARNING OBJECTIVES

## What is Document Object Model?

- What is the usage of DOM?
- How does it relate to a web page?
- How does usage of Document Objects help in web applications development?

## What is Browser Object Model ?

- What are the available Browser objects?
- How does usage of Browser Objects help in web applications development?

# DOCUMENT OBJECT MODEL

When a browser loads a web page, it also remembers the structure of the web page.

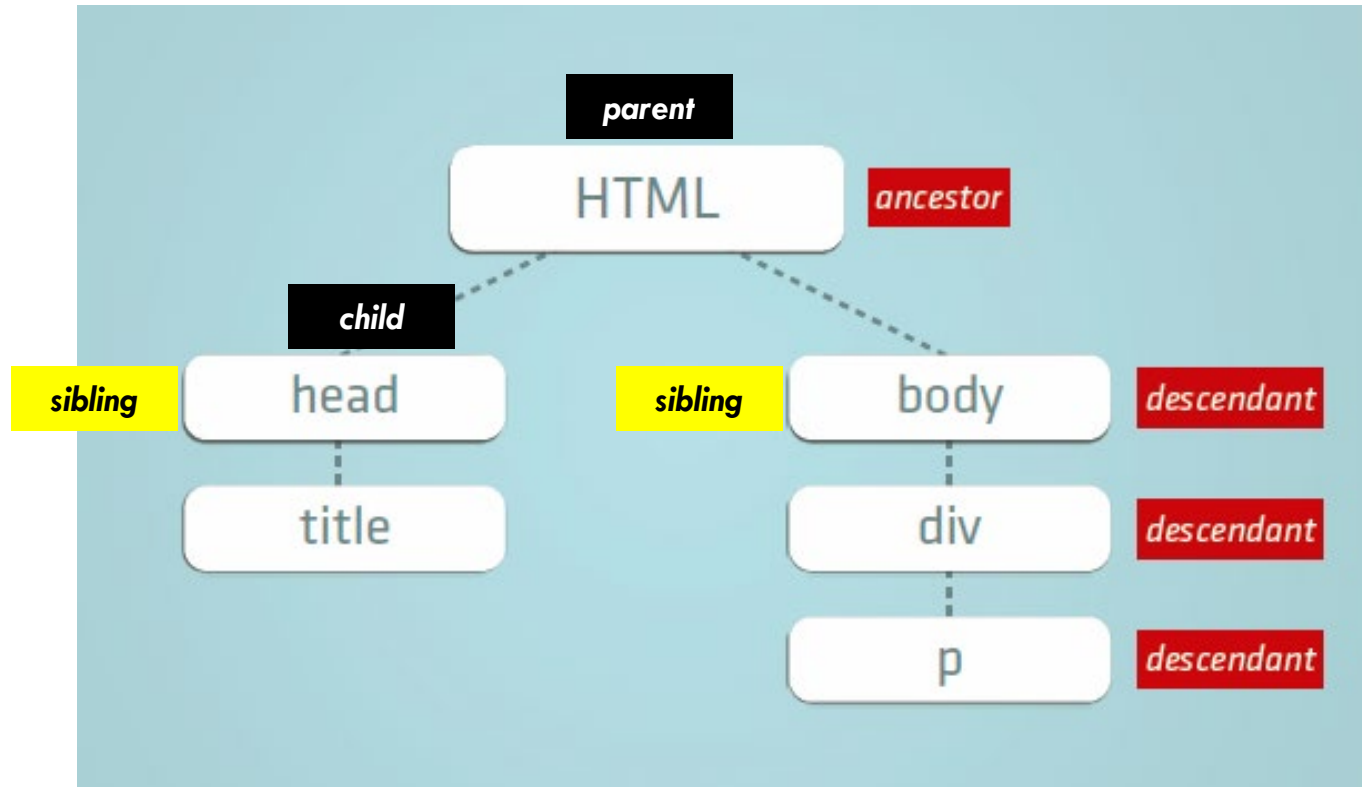
The structure contains information about the HTML tags, their attributes, and the order in which they appear in the file.

Such structure or model is called the Document Object Model (DOM).

Different documents are different in their structures.

The DOM lets JavaScript communicate with and change a web page.

# DOCUMENT OBJECT MODEL

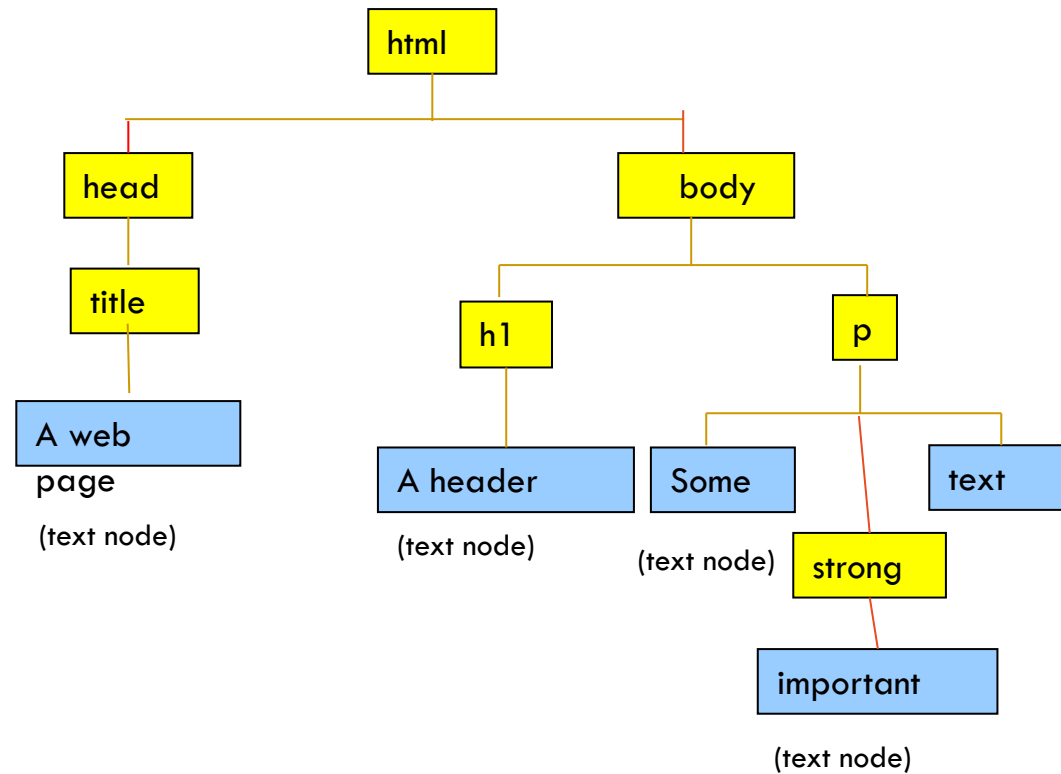


Each of the tags/elements and text are called **nodes**.

# DOCUMENT OBJECT MODEL

## Example: Basic structure of a HTML page

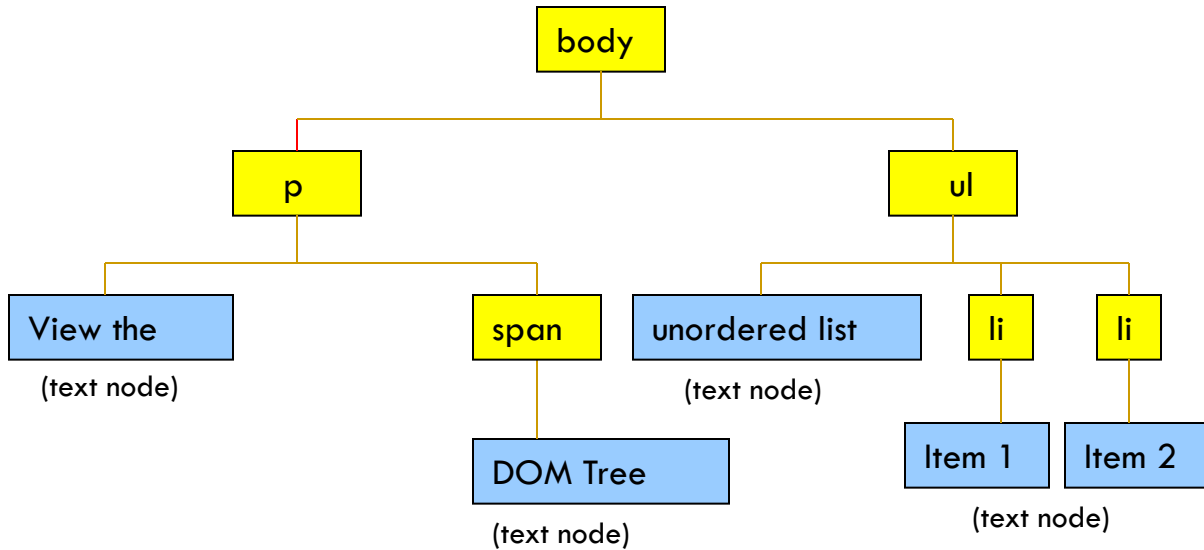
```
<html>
<head>
<title>A web page</title>
</head>
<body>
<h1>A header</h1>
<p>Some
<strong>important</strong>
text
</p>
</body>
</html>
```



# DOCUMENT OBJECT MODEL

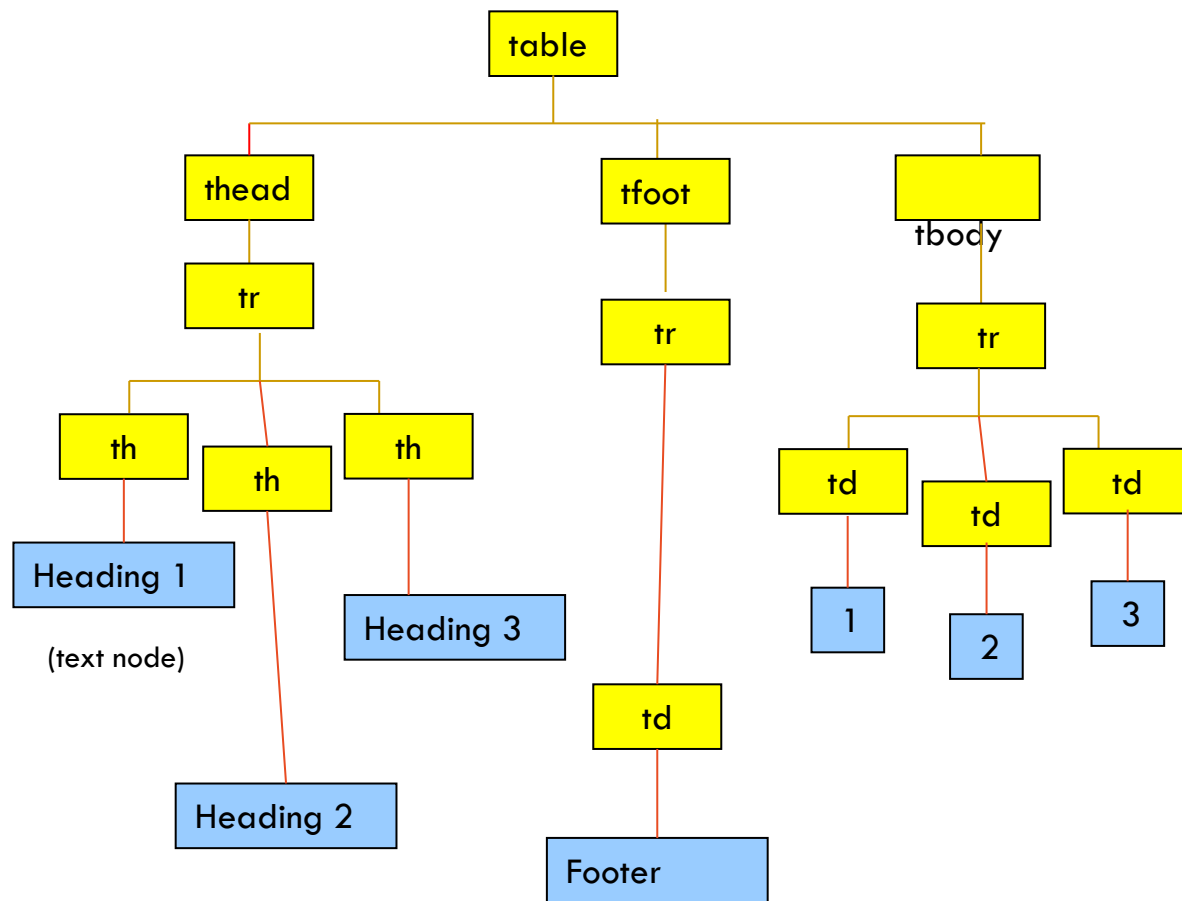
## Example (branch)

```
<body>
  <p>
    View the
    <span>DOM
    Tree</span>
  </p>
  <ul>
    Unordered list
    <li>item 1</li>
    <li>item 2</li>
  </ul>
</body>
```



# DOCUMENT OBJECT MODEL

```
<table border="1">
<thead>
<tr>
<th>Heading 1</th>
<th>Heading 2</th>
<th>Heading 3</th>
</tr>
</thead>
<tfoot>
<tr>
<td colspan="3">Footer</td>
</tr>
</tfoot>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
```



# DOM — SELECT AN ELEMENT/NODE

```
var special=document.getElementById("special");
```

or using CSS selector

```
var special=document.querySelector("#special");
```



# DOM — SELECT A GROUP OF ELEMENTS/NODES

```
var links =document.getElementsByTagNameName("a");
```

*links[0] is the first link in the group*

```
var sLinks =document.getElementsByClassName("sLinks");
```

*sLinks[1] is the second link in the group*

or using CSS selector

```
var allPhotos=document.querySelectorAll(".myPhotos");
```

*allPhotos[0] is the first photo in the group*

If use `document.querySelector(".myPhotos")` will only get the first element of myPhotos class.

# DYNAMICALLY CHANGE TEXT CONTENT OF ELEMENT

```
<h1 id="topic">How To Cook Rice</h1>
```

```
document.getElementById("topic").innerText="How to  
make coffee";
```

# DYNAMICALLY CHANGE HTML CONTENT OF ELEMENT

```
<div class="show">
  <ol>
    <li>Nasi Lemak</li>
    <li>Ice Cream</li>
  </ol>
</div>
```

Select and style every `<ol>` element where the parent is with `.show` class.

```
document.querySelector(".show > ol").innerHTML=
"<li>coffee</li><li>Milo</li>";
```

Content will be taken as html tags but not text only.

# DYNAMICALLY VALUE OF INPUT ELEMENT

Change the value of a input element:

```
<label>
```

```
First Number <input id="first"  
type="number"/>
```

```
</label>
```

```
var value1=document.getElementById("first").value;
```

# DYNAMICALLY CHANGE ATTRIBUTE OF ELEMENT

```
<a href="http://www.google.com">I am a link</a>  

```

```
var link = document.querySelector("a");  
var destSite= link.getAttribute("href"); // get the destination  
link.setAttribute("href", "http://www.dogs.com"); // get the destination  
  
var img = document.querySelector("img");  
img.setAttribute("src", "corgi.png");
```

# DYNAMICALLY CHANGE CLASS NAME

If only one class

```
divShow=document.querySelector(".show");  
divShow.setAttribute("class","hide");
```

The **setAttribute()** method sets a new value to an attribute.

If has element belongs to a list of classes

The **classList** property returns the CSS classnames of an element.

```
divShow=document.querySelector(".show");  
divShow.classList.remove("show");  
divShow.classList.add("hide");
```

Note: classList may not be supported by all browsers yet

# DYNAMICALLY CHANGE STYLE OF ELEMENTS

Style of element can be dynamically changed, for example:

```
document.getElementById("heading").style.color="red"
```

```
document.getElementById("heading").style.display="none"
```

*Note: Setting with the inline styles*

# SETTING CSS STYLE IN JAVASCRIPT

Properties that consists of more than one words that join with hyphen :

For example: **font-size**

The hyphen is to be taken away and replace the first character of the next word with capital letter, such as:

```
document.getElementById("p1").style.fontSize
```



# DYNAMICALLY ADDING NEW ELEMENTS

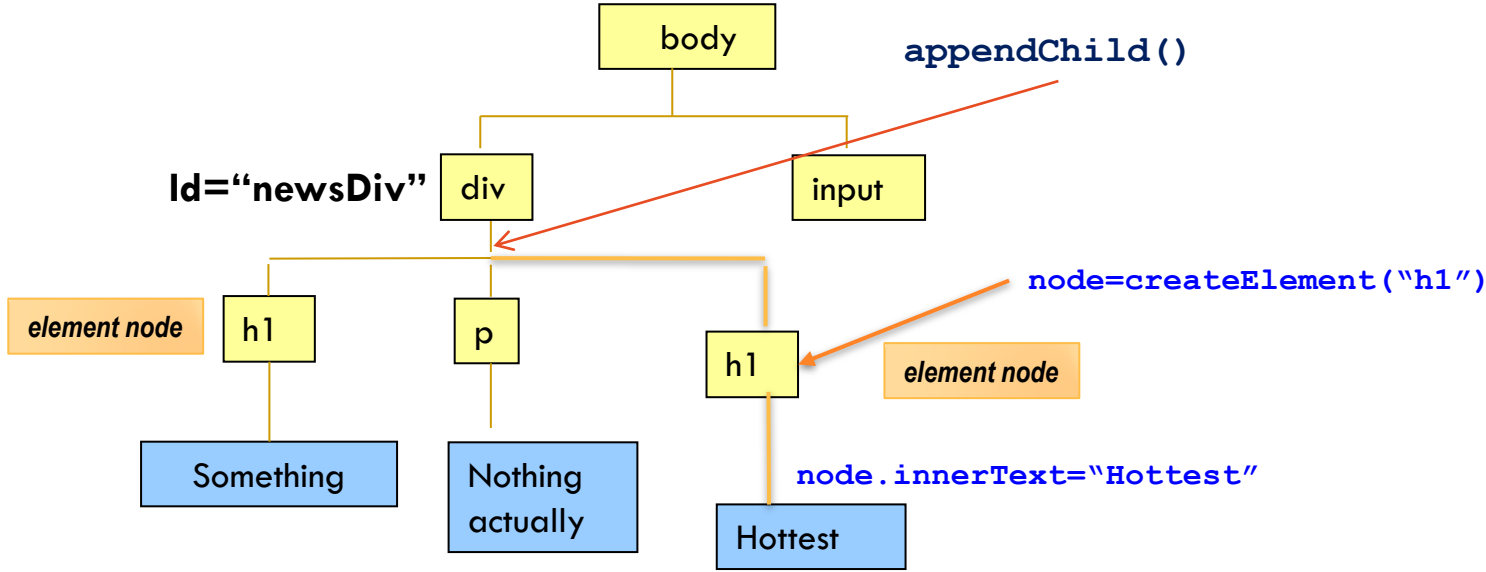
New elements in a document can be dynamically added by adding child nodes to existing nodes.

Example of methods to be use for nodes operations:

```
var node=document.createElement("h1");  
node.innerText="Hottest";  
document.getElementById("newsDiv").appendChild(node);
```

```
<body>
<div id="newsDiv">
<h1>Something</h1>
<p>Nothing actually</p>
</div>
```

```
var node=document.createElement("h1");
node.innerText="Hottest";
document.getElementById("newsDiv").appendChild(node);
```



Something

Nothing actually

Add Heading

Something

Nothing actually

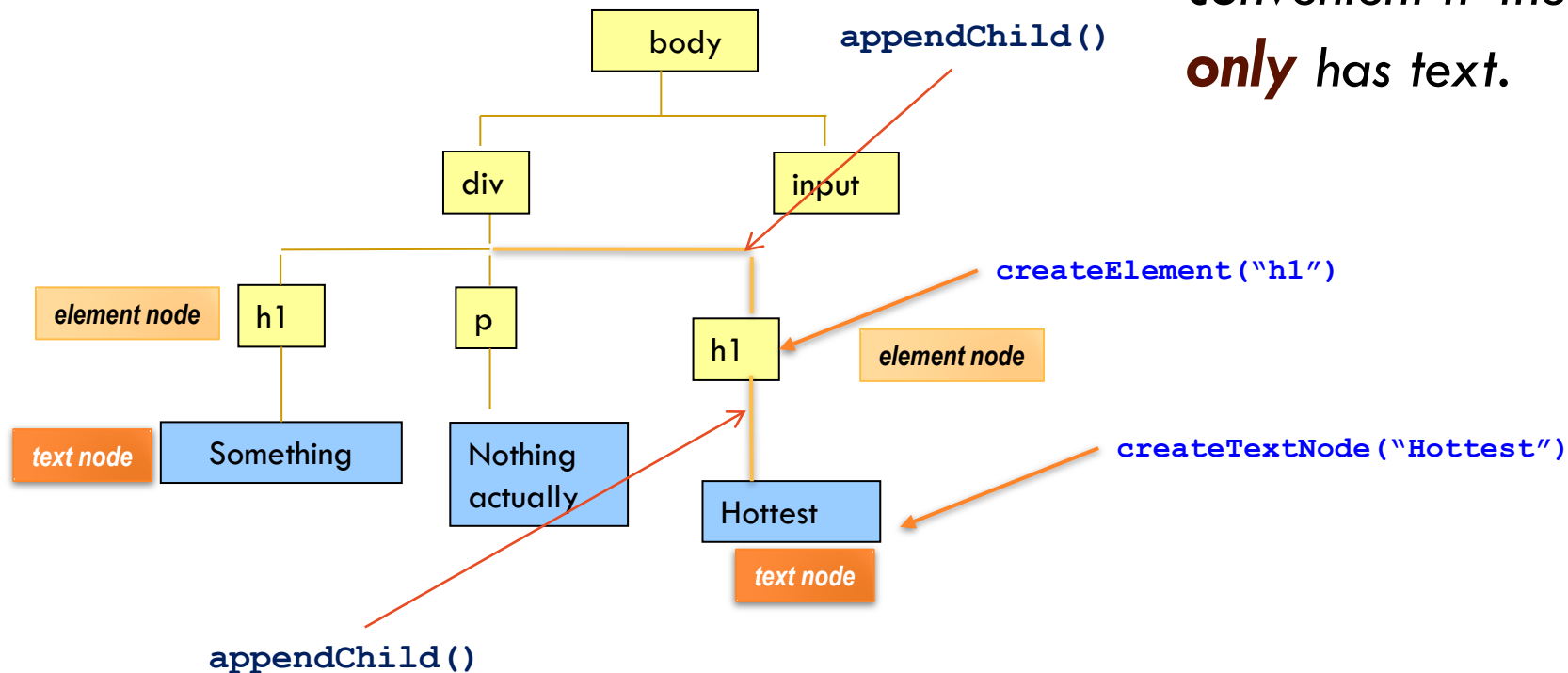
Hottest

Add Heading

# CREATE TEXTNODE VS SETTING INNERTEXT

```
var node=document.createElement("h1");  
var textnode=document.createTextNode("Hottest");  
node.appendChild(textnode);  
document.getElementById("newsDiv").appendChild(node)  
;
```

*Use `innerText` is more convenient if the node **only** has text.*



# DYNAMICALLY REMOVE ELEMENTS

```
<body>

<h2>JavaScript HTML DOM</h2>
<h3>Remove an HTML Element.</h3>

<div>
<p id="p1">This is a paragraph.</p>
<p id="p2">This is another paragraph.</p>
</div>
```

**`document.getElementById("p1").remove();`**

Refer to [https://www.w3schools.com/js/js\\_htmlDOM\\_nodes.asp](https://www.w3schools.com/js/js_htmlDOM_nodes.asp) for more info

# EVENTS

## Something happen

- Actions that users perform, e.g.
  - Moving the mouse
  - Click on a button
- Web browser operation, e.g.
  - Loading a document



# EVENTS (CONT.)

## Mouse Event

- click, dblclick, mousedown, mouseup, mouseover, mouseout, mousemove

## Form Event

- submit, reset, change, focus, blur, select

## Keyboard Event

- keypress, keydown, keyup


## Others

- load, unload, error, abort

View <https://developer.mozilla.org/en-US/docs/Web/Events> for reference of events

# HANDLING AN EVENT — ADD INLINE HANDLER

```
<style>
  span { font-size:2em; color:red }
</style>
<script>
function show() {
  document.getElementById("who").innerText="Interested to know who? Double click";
}
function tell() {
  document.write("<h1>I am the superman!!</h1>");
}
</script>
</head>
<body>
<p>Do you know
  <span id="who" onmouseover="show();" ondblclick="tell()">who</span>
  am I ?</p>
</body>
```



show

# HANDLING AN EVENT — REGISTER LISTENER

```
1 <!doctype html>
2 <html lang="en" >
3 <head>
4   <meta charset="utf-8"/>
5   <style>
6     span { font-size:2em; color:red }
7   </style>
8   <script>
9     function show() {
10       document.getElementById("who").innerText="Interested to know who? Double click";
11     }
12     function tell() {
13       document.write("<h1>I am superman lah!!</h1>");
14     }
15   </script>
16 </head>
17 <body>
18 <p>Do you know
19   <span id="who">who</span>
20   am I ?</p>
21 <script>
22   document.getElementById("who").addEventListener("mouseover",show);
23   document.getElementById("who").addEventListener("dblclick",tell);
24 </script>
25 </body>
26 </html>
```



# HANDLING AN EVENT — REGISTER LISTENER

```
<style>
  span { font-size:2em; color:red }
</style>
<script>
function show() {
  document.getElementById("who").innerText="Interested to know who? Double click";
}
function tell() {
  alert("Wait huh!!");
  document.getElementById("who").removeEventListener("mouseover",tell);
  document.getElementById("who").addEventListener("dblclick",function() {
    document.write("<h1>I am superman lah!!</h1>")
  })
}
</script>
</head>
<body>
<p>Do you know
  <span id="who">who</span>
  am I ?</p>
<script>
  document.getElementById("who").addEventListener("mouseover",show);
  document.getElementById("who").addEventListener("mouseover",tell);
</script>
```



The diagram consists of two red arrows. The first arrow originates from the `show` function definition in the script block and points to the `show` argument in the `addEventListener` call within the `<script>` block at the bottom. The second arrow originates from the `show` argument in the `addEventListener` call and points to the `show` function definition in the script block.

show

# WAYS TO HANDLE EVENTS

## Through inline handler

- Mixed with html tag (as an attribute of html element)
- Easier to add, harder to maintain
- One handler for one event

## Register listener

- Separated from html tag
- Easier to maintain
- Allow the queuing of multiple handlers for one event

# JAVASCRIPT TIMING EVENTS

**setTimeout()** - allows to run a function once after the interval of time.

**setInterval()** - allows to run a function regularly with the interval between the runs.

**clearTimeout()** - stop the execution of the function specified in the setTimeout() method.

**clearInterval()** - stop the execution of the function specified in the setInterval() method.

# EXAMPLE: USAGE OF SETTIMEOUT

```
<script type="text/javascript">  
function turnOff() {  
    document.getElementById("lightBulb").style.clip="rect(0px, 300px, 400px, 60px) ";  
    document.getElementById("lightBulb").style.left="250px";  
    document.getElementById("message").innerText="Go to Sleep!!";  
}  
window.onload=function() {  
    timer=setTimeout(turnOff,2000);  
}
```

2 secs later after the page load, turn off the light

[Show](#)

# TO SET TIMER AFTER DOCUMENT LOADED

```
window.onload=function() {  
    timer=setTimeout(turnOff,2000);  
}
```

**Can also be written as**

```
document.addEventListener("DOMContentLoaded",docIsReady);  
  
function docIsReady(){  
    timer=setTimeout(turnOff,2000);  
}
```

**Or set the timer at the end of the document**

```
timer=setTimeout(turnOff,2000);
```

# EXAMPLE: USAGE OF SETINTERVAL

```
1 <!doctype html>
2 <html>
3 <head>
4 <style>
5   span {font-size:2em; color:red}
6 </style>
7 <script>
8   var secs=10;
9   var timer;
10  function countDown() {
11    secs--;
12    document.getElementById("secsLeft").innerText=secs;
13    if(secs==0) {
14      clearInterval(timer);
15      alert("Time's Up");
16    }
17  }
18  window.onload=function() {
19    document.getElementById("secsLeft").innerText=secs;
20    timer=setInterval(countDown,1000);
21  }
22 </script>
23 </head>
24 <body>
25 <p>You left <span id="secsLeft"></span> secs</p>
26 </body>
27 </html>
```

1. Count Down in 10 secs
2. Every one second update the display
3. When reach 10 secs, clear the timer

[Show](#)

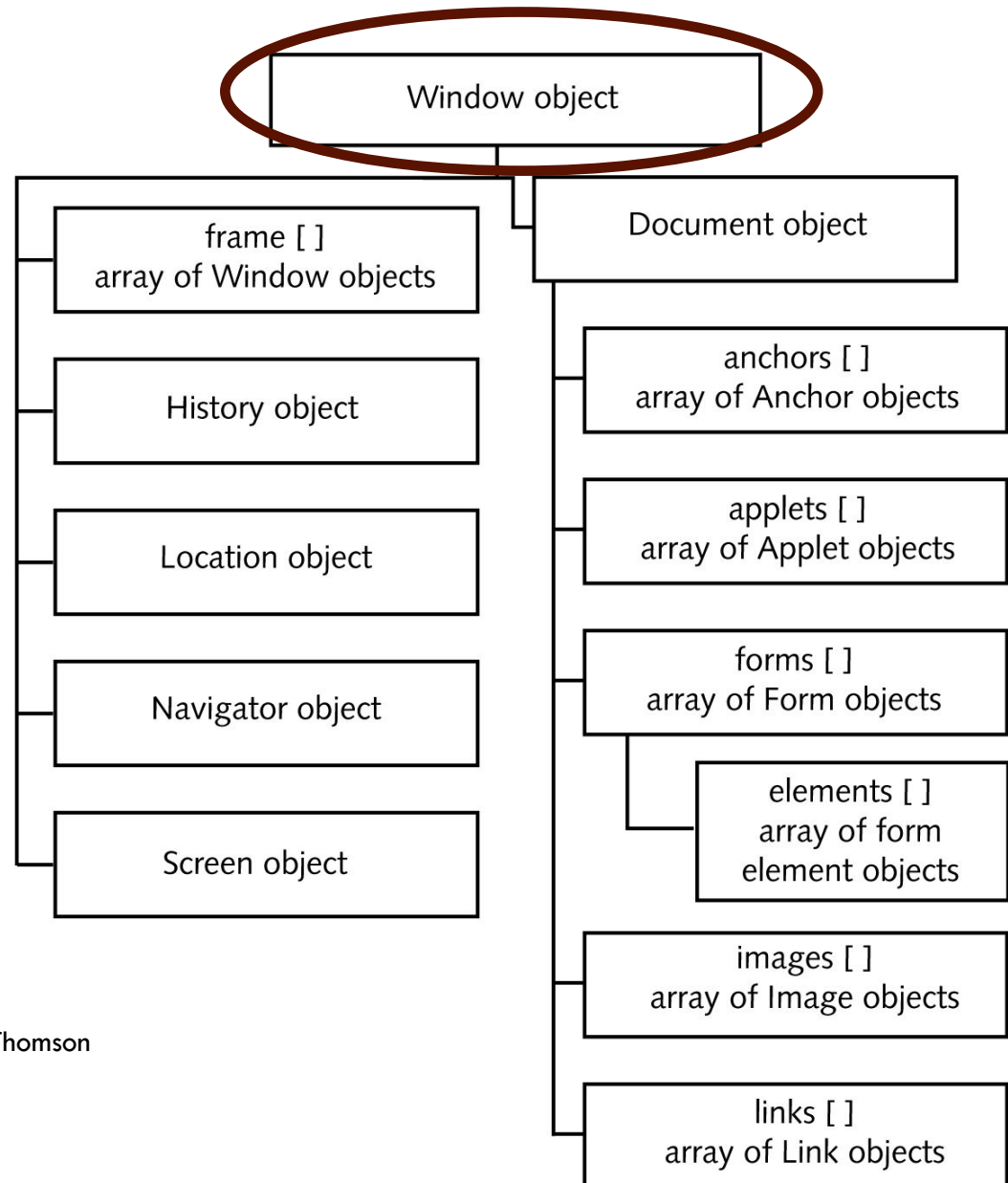
# BROWSER OBJECTS

Objects of browser, such as *window*, *document*, *navigator*, *screen*, *history*, *location*

There is *no official standard* across all browsers

Browser objects allow JavaScript to interface and interact with the browser itself

# BROWSER OBJECT MODEL (AN EXAMPLE)



Source from:JavaScript Third Edition by Don Gosselin, publisher :Thomson



# BOM OBJECT - WINDOW

```
function goNew(url){  
    var win1 = window.open(url, "win1",  
        "toolbar=no,resizable=no, width=900px,  
height=600px")  
}
```

```
<input type="button" value="go Google" onclick="goNew('http://www.google.com')"/>  
<input type="button" value="go W3Schools" onclick="goNew('http://www.w3schools.com')"/>
```

Example

# BOM OBJECT - LOCATION

```
function go() {  
    var index=document.getElementById("searchTool").selectedIndex;  
    location.href=document.getElementById("searchTool").options[index].value;  
}
```

```
<select id="searchTool" onchange="go()">  
  <option value="">select your search engine</option>  
  <option value="http://www.google.com">go google</option>  
  <option value="http://www.yahoo.com">go yahoo</option>  
  <option value="http://www.bing.com">go bing</option>  
</select>
```

Example

# BOM OBJECT - NAVIGATOR

```
<script type="text/javascript">  
  
    document.write("<b>Browser:</b> " + navigator.appName);  
    document.write("<br><b>Version:</b> " + navigator.appVersion);  
    document.write("<br><b>Platform:</b> " + navigator.platform);  
  
</script>
```

[Check browser](#)

# BROWSER COMPATIBILITY ISSUES

Different versions of browsers may be different in their support for such as

- font availability
- tags
- CSS styles & properties (not supported or appear differently)
- DOM objects, methods....

<http://www.netmechanic.com/products/Browser-Tutorial.shtml#1.2>

Try to use the common standard tags and features(w3c)

# SUMMARY

## What is Document Object Model?

- What is the usage of DOM?
- How does it relate to a web page?
- How does usage of Document Objects help in web applications development

## What is Browser Object Model ?

- What are the available Browser objects?
- How does usage of Browser Objects help in web applications development?