Functions

Functions - Why?

Functions are re-usable pieces of code that perform common tasks.
Make sure the function is loaded when it is called, often put into the <head> section.

Examples:

```
alert(message); //built-in
sayHello(name); //self-made
```

Divide and conquer problems.
Re-use and organise code.

Define a Function

Syntax:

```
function <name> (<argument[, argument]>)
{
  //JavaScript statements
}
```

Example:

```
function sayHello()
{
   alert( "Hello, I have been clicked !!" )
}
```

Always include brackets, even if you don't need an argument.

```
<script type="text/javascript">
<!--
function checkYourNumber()
{
   var n = prompt("Check your number", "");
   n = parseInt(n);
  if (isNaN(n))
   { alert("The input is not a number"); }
  else
  {
      alert("The number is " + n );
  }
}
//-->
</script>
</head>
<body>
<h1>'Nan' Example</h1>
<script type="text/javascript">
```

```
   checkYourNumber();
</script>
</body>
```

Parameters and Arguments

Parameters provide placeholders for values in the declaration of functions. Arguments are actual values that are provided when the function is called.

Examples of arguments:

- Variables, e.g. name
- Literals, e.g. 20 or “Hello, user!”

The values are “accessed” within the body of the function.
Example:

```
<script type="text/javascript">
<!--
function checkYourNumber( num ) //num is the parameter
{
  var in_num = parseInt(num);

 if (isNaN(in_num))
  { alert("The input cannot be parsed to a number"); }
 else
 {
    alert("The number is " + in_num );
 }
}
 //-->
</script>
</head>
<body>
<script type="text/javascript">

var n = prompt("Check your number", "");
checkYourNumber( n ); // n is the argument

</script>
```

Where to Put Functions?

Put Functions inside script tags in the head.

```
<head>
<script type=”text/javascript”>
function sayHello() {...}
</script>
</head>
```

Or into an external JavaScript file that is imported into the <head> section.

```
<head>
<script type=”text/javascript” src=”sayHello.js”>
</script>
</head>
```

Returning Values from Functions (1)

Sometimes functions “produce” values.

- we have already seen this …. The prompt box.

If the user clicks "OK" the box returns the input value.
If the user clicks "Cancel" the box returns null.

```
var name=prompt("Whats your name?","Your name");
```

Returning Values from Functions (2)

In the calling statement the return value of the function replaces the call to the function.

Inside the function:

- Use the return keyword …
- followed by the data / value to be returned.

A function can only return a single piece of data.
return exits the function.

Example:

```
<script type="text/javascript“>
function add_nums(a,b)
{
   var sum = a +b;
   return sum;
}
</script>
</head>
<body>
```

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

   document.write( "The answer is " + add_nums(30,12))

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

JavaScript Buttons

Button Object

The Button object represents a push button.
normally used to trigger an event such as ‘OnClick’ event
the <input> element should be used to create buttons in an HTML form.

Standard Attributes include :

- type – button/reset/submit
- name/id
- value – button text
- event – usually ‘onclick’

```
<script type="text/javascript">
<!--
function showtext(txt)
{
  alert(txt);
}
// -->
</script>
</head>

<body>
<input type="button" id="button1" onclick="showtext('Hello')" value="Click Me">

</body>
</html>
```

id attribute

The id attribute specifies a unique id for an HTML element.
The id must be unique within the HTML document.
The id attribute can be used by a JavaScript (via the HTML DOM) or by CSS to make changes or style the element with the specified id.

- We have already used the id attribute within CSS

document.getElementById

The getElementById() method returns a reference to the object with the specified ID.

```
<script type="text/javascript">
<!--
function changeSize()
{
document.getElementById("smileyface").height="150       ";
document.getElementById("smileyface").width="120";
}
// -->
</script>
</head>
<body>
<img id="smileyface" src="smiley1.jpg" width="80" height="60" />

<input type="button" id="buttonchange" onclick="changeSize()" value="Change
height and width of image">
</body>
```

InnerHTML

innerHTML Property

http://www.w3schools.com/js/js_htmldom.asp

This is an easy way to get or modify the content of an HTML element

It can also be used to view the source of a page that has been dynamically modified.

Each HTML element has an innerHTML property that defines both the HTML code and the text that occurs between that element's opening and closing tag.

By changing an element's innerHTML after some user interaction, you can make much more interactive pages.

- For example can change , <p> etc

However, using innerHTML requires some preparation if you want to be able to use it easily and reliably.

- You must give the element you wish to change an ‘Id’ and use the getElementById function

```
<script type="text/javascript">
<!--
function changeText()
{
  document.getElementById('mainText').innerHTML = 'Hello World !!!! ';
```

```
}
//-->

</script>

</head>
<body>

<p id='mainText'>Welcome All !!!!</p>

<input type='button' onclick='changeText()' value='Change Text'/>
</body>
```

use a button to change innerHTML of the txt in a <p> tag

```
<script type="text/javascript">
function CopyValues()
{
  document.getElementById("displayText").innerHTML =
document.getElementById('field1').value
}

function CopyValuesBig()
{
  document.getElementById("displayText").innerHTML = "<h1>" +
document.getElementById('field1').value + "</h1>"
}
</script>
</head>

<body>
  Field1: <input type="text" id="field1" value="Hello World!" />
<br />
Click the button to copy the text
<br />
 <input type="button" value= "Copy Text"  onclick="CopyValues()“>
<br/>
 <input type="button" value= "Copy Text - Large "  onclick="CopyValuesBig()">
<br/>
  <div id ="displayText"> </div>
</body>
```

Change the innerHTML of a field

displays relevant messages

- Including error message

uses TWO fields

This example also uses a ‘Clear’ button

- to clear each specific items

```
<script type="text/javascript">
function CheckValues()
{
 var fieldval = parseInt(document.getElementById('field1').value);
 if (isNaN(fieldval) )
 {
   //set error message to red font - use 'errorText’ div
   document.getElementById("errorText").innerHTML = "<span
style='color:#FF0000'>" + "Error  -  Not a Number" + "</span>";
 }
 else
 {
   // display message that Number is valid - use displayText' div
   document.getElementById("displayText").innerHTML = "The Number is ...." +
document.getElementById('field1').value
 }
}

function ClearValues()
{
  // reset all fields
  document.getElementById('field1').value = "";
  document.getElementById("errorText").innerHTML =  ""
  document.getElementById("displayText").innerHTML = ""
}
</script>
<!-- A Table has been used to keep web page tidy -->
<table>
<tr>
<td> Enter a Number : </td>
<td> <input type="text" id="field1" value=" " /> </td>
<td> <div id ="errorText"> </div> </td>
</tr>
<tr>
<td>
<input type="button" value= "Check Number "  onclick="CheckValues()">
</td>
<td>
<input type="button" value= "Clear "  onclick="ClearValues()">
</td>
<td>
</td>
</tr>
<tr>
```

```
<td colspan =3>
<div id ="displayText"></div>
</td>
</tr>
</table>
```

display relevant messages – uses TWO fields

```
<script type="text/javascript">
function checkAnswer()
{
 var Q1_txt = ""
 var Q2_txt = ""
 // check if purple button has been selected
 if (document.quiz.colour[3].checked)
  {
   Q1_txt = "Q1  - Correct Choice"
  }
 else
  {
   Q1_txt = "Q1  - Wrong Choice";
  }

  // check if Washington button has been selected
  if (document.quiz.city[1].checked)
  {
   Q2_txt = "Q2  - Correct Choice"
  }
 else
  {
   Q2_txt = "Q2  - Wrong Choice"
  }
 document.getElementById ("Q1_ans").innerHTML =  Q1_txt
 document.getElementById ("Q2_ans").innerHTML =  Q2_txt
}
</script>
<form name="quiz">

Question 1. What is the Odd One Out  ? <br />

<input type="radio" name="colour" value="Glasgow" />Glasgow ? <br />
<input type="radio" name="colour" value="London" />London ? <br />
<input type="radio" name="colour" value="Edinburgh" />Edinburgh ? <br />
<input type="radio" name="colour" value="Purple" />Purple ? <br />

Question 2. What city is NOT in Europe?    <br />

<input type="radio" name="city" value="Paris" />Paris ? <br />
```

```
<input type="radio" name="city" value="Washington" />Washington ? <br />
<input type="radio" name="city" value="Barcelona" />Barcelona ? <br />
<input type="radio" name="city" value="Dublin" />Dublin ? <br />

<input type="button" value= "Check Your Answer"  onclick="checkAnswer()">
</form>

<div id = "Q1_ans" > </div>
<div id = "Q2_ans" > </div>
```

a basic Quiz using Radio Buttons and inner HTML

HTML5 - Range

```
<script type="text/javascript">
function ShowRangeValue()
{
document.getElementById('RANGE_field').value=document.getElementById('slider1'
).value;
}

function updateval_innerhtml (vl)
{
document.getElementById('val').innerHTML = vl;
}
</script>
<body>
Range (1-50) :
<br/>
1
<input id="slider1" type="range" min="1" max="50"
oninput="updateval_innerhtml(value) " />
50
<br />
<input type="button" value= "Show Range Value"  onclick="ShowRangeValue() ">
<br />
<em>
<div id="val"> 50 </div>
( This will be explained at next Lecture - uses innerHTML )
</em>
Range Output Value :  <input type="text" size = "5" id="RANGE_field"
disabled="disabled" />
```

Document Object Model (DOM)

The Document Object Model (DOM) is an application programming interface (API) for HTML and XML documents.

The DOM represents a document as a hierarchical tree of nodes, allowing developers to add, remove, and modify individual parts of the page.

The DOM is a cross-platform, language-independent way of representing and manipulating pages for markup.

Defines the logical structure of documents and the way a document is accessed and manipulated

Has been designed to be used with any programming language.

To "make it easy for programmers to access components and to delete, add, or edit their content, attributes and style."

JavaScript

Is used by JavaScript to inspect or modify a web page dynamically.

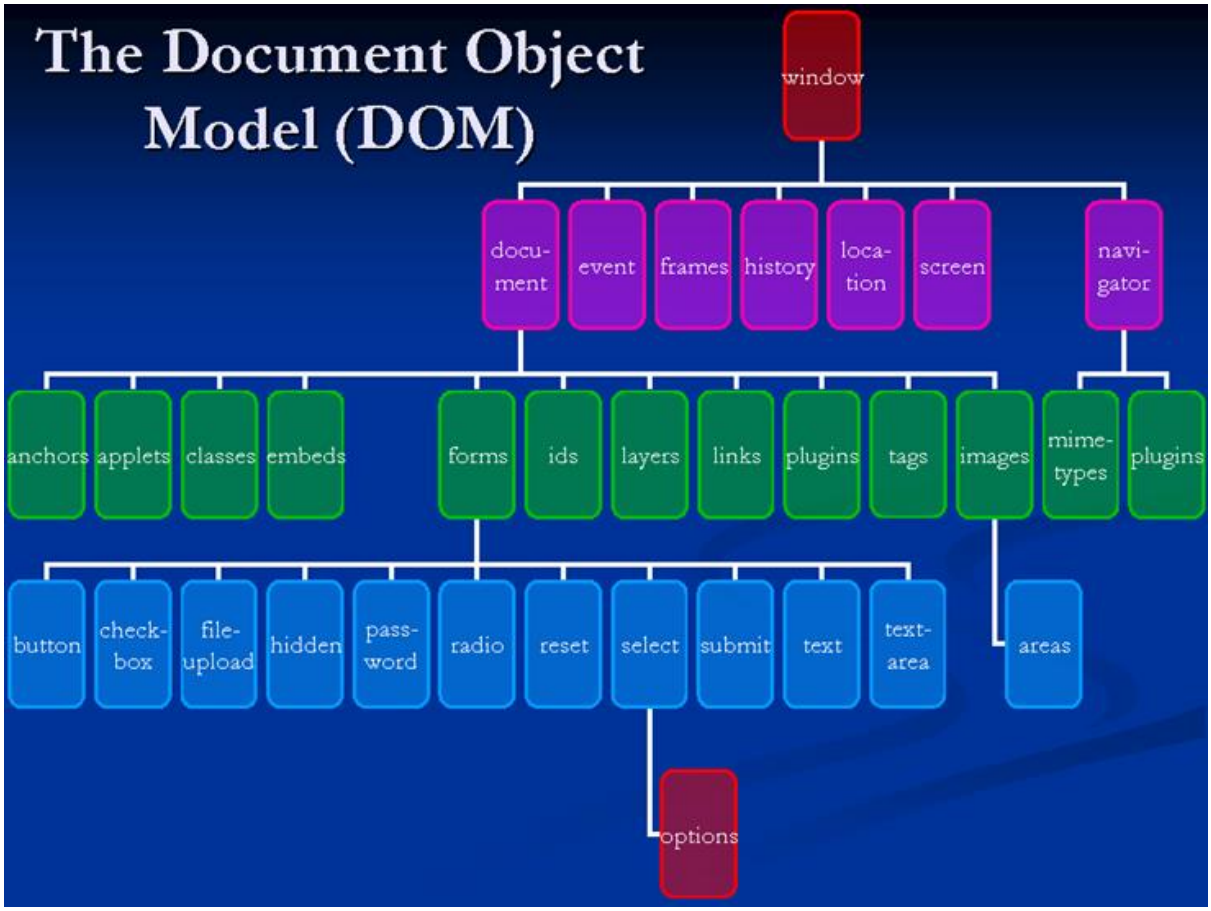
Through DOM JavaScript can build documents, navigate their structure, and add, modify, or delete elements and content.

Anything found in an HTML or XML document can be accessed, changed, deleted, or added using the Document Object Model, with a few exceptions

JavaScript Browser Objects

They are automatically created by the JavaScript engine at run time.

- Window: represents a browser window.
- Location: contains information about the current URL.
- History: consists of an array of URLs. These URLs are the URLs the user has visited within a browser window.
- Screen: contains information about the client's display screen.



Example used earlier ... Note the use of the DOM :

- document -> form -> textBox -> value

```
<script type="text/javascript">
function Test()
{
// access form value
var val1 = document.exampleform.T1_txt.value
var val2 = document.exampleform.T2_txt.value
....
}
</script>
<form name="exampleform">
  <input type="text" id="T1_txt" name="T1_txt" size="15"  maxlength="10">
  <textarea  name="T2_txt" id="T2_txt"  rows="5" cols="20">This is a
textarea</textarea>
  <p><input type="button" value="Click Here" id="B1" onClick="Test()"></p>
</form>
```

The document Object

Represents the entire HTML document.

Can be used to access all elements in a page.

Is part of the window object.

http://www.w3schools.com/jsref/dom_obj_document.asp
http://www.w3schools.com/JS/js_examples.asp

Node Types

Document
- represents the HTML document

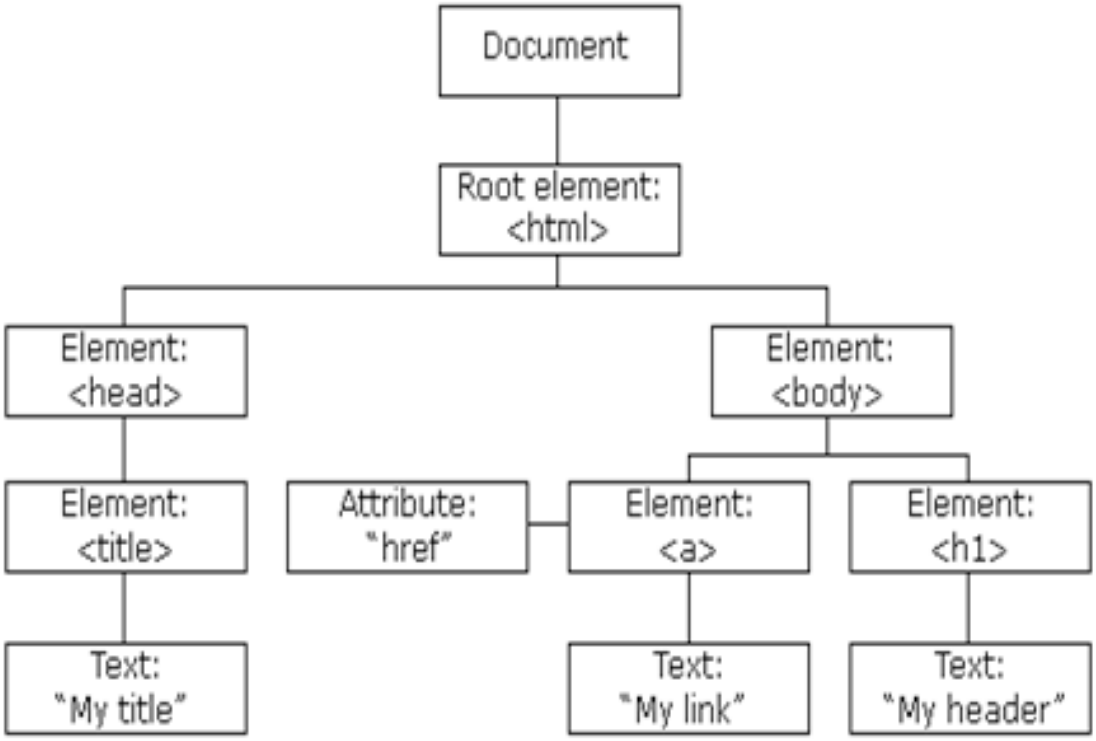
Element
- an HTML element that corresponds to an HTML tag

Text
- text content for an element

Attribute
- attribute of an element node, not present directly in the DOM tree

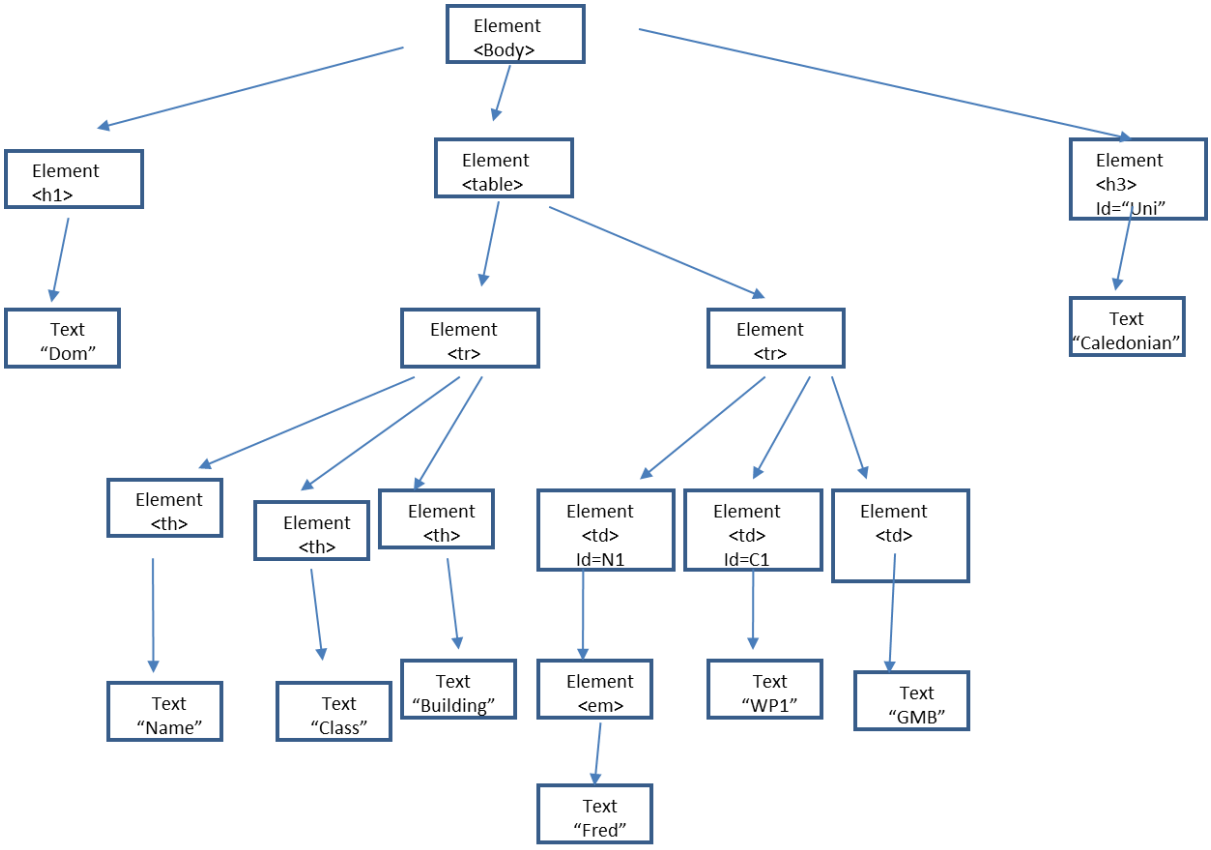
DOM Trees

```
<html>
<head>
<title>My Title</title>
</head>
<body>

<a href="target.html"> My link </a>

<h1>My header</h1>

</body>
</html>
```



DOM Example

```
<body>
<h1>DOM</h1>
<table>
  <tr>
    <th>Name</th>
    <th>Class</th>
    <th>Building</th>
  </tr>
  <tr>
    <td id="N1"><em>Fred</em></td>
    <td id="C1">WP1</td>
    <td>GMB</td>
  </tr>
</table>
<h3 id="Uni">Caledonian</h3>
</body>
```



DOM Properties

nodeType

- The type of a node, e.g. TEXT, ELEMENT

nodeValue

- The value stored in a node.
- Only for text and attribute nodes.

childNodes

- Array containing all child nodes under a node.
- In the order in which they appear in the HTML document.

parentNode

- The parent node of the current node.

firstChild

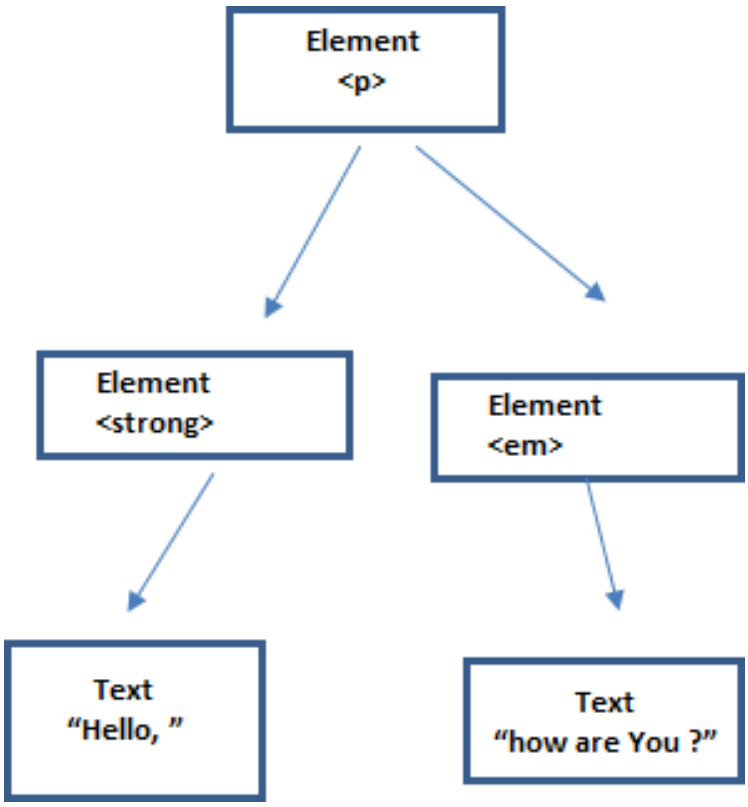
- The first child node under a node.

lastChild

- The last child node under a node.

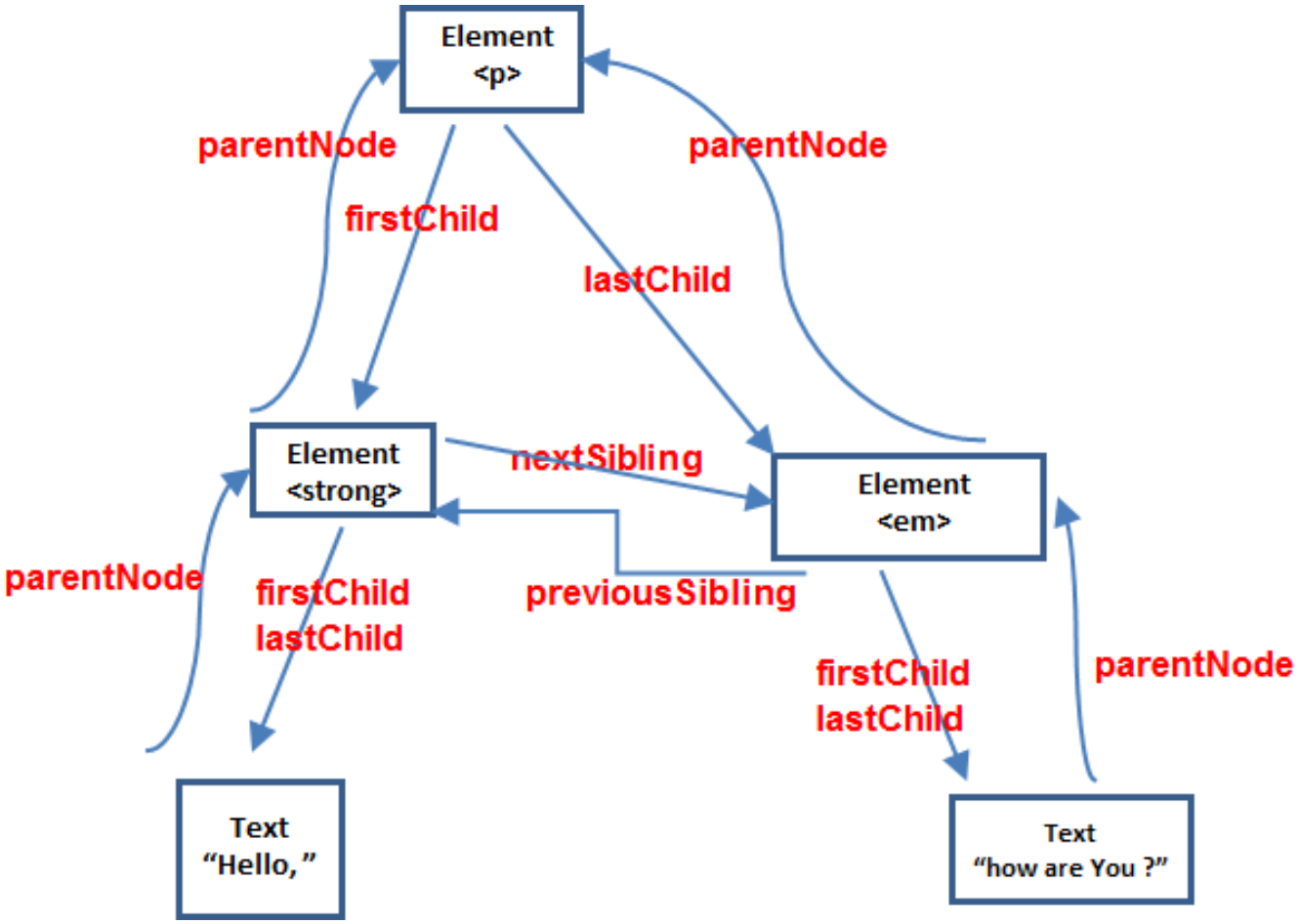
nextSibling/previousSibling

- Sibling nodes


Element
<p>
Element

Element

Text
“Hello, ”
Text
“how are You ?”


Element
<p>
parentNode
parentNode
firstChild
lastChild
Element

nextSibling
Element

parentNode
firstChild
lastChild
previousSibling
firstChild
lastChild
parentNode
Text
“Hello, ”
Text
“how are You ?”

DOM Methods

appendChild(<node>)

- Appends a child node onto the current node.

removeChild(<node>)

- Removes the specified child node from the current node.

createTextNode(<message>)

- Creates a text node with the specified text.

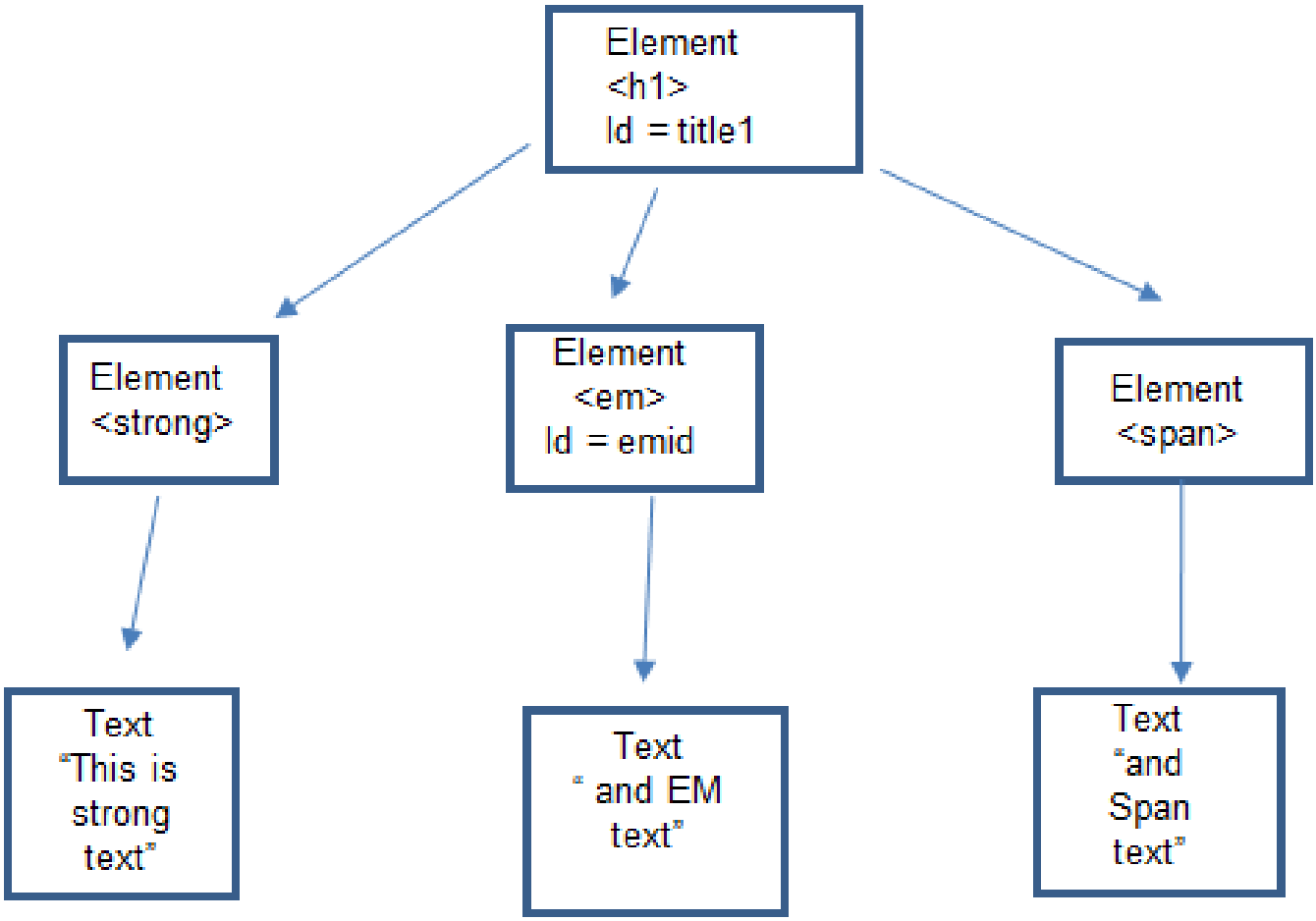
createElement(<element>)

- Creates the specified element.

Dom Examples

```
<h1 id="title1">
<strong>This is strong text </strong><em id = "emid">and EM text</em><span>and
Span text</span>
</h1>
```

Output is: **This is strong text***and EM text* and Span text

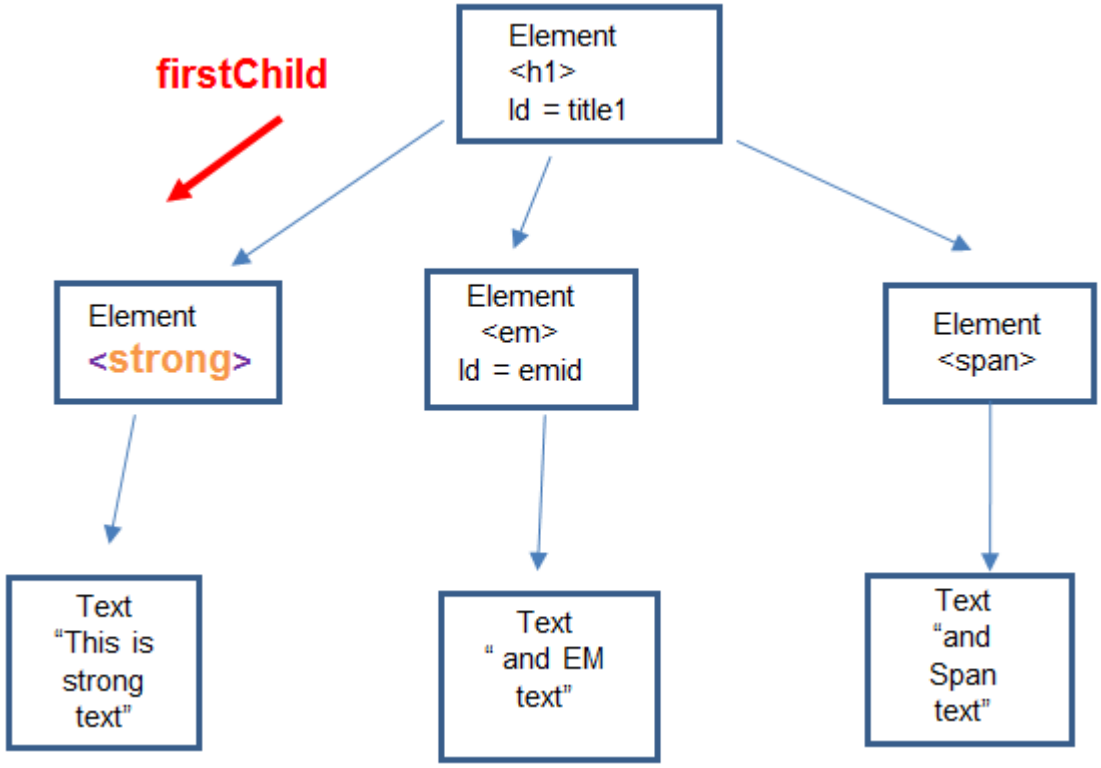


a basic Quiz using Radio Buttons and inner HTML

Example 1

Get Node Name of first child

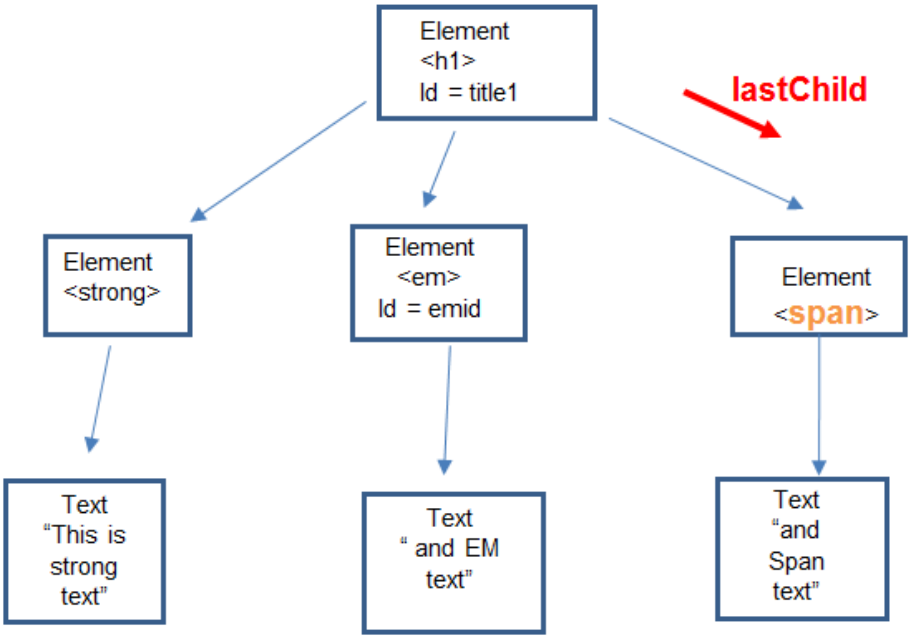
document.getElementById('title1').firstChild.nodeName



Example 2

Get Node Name of last child

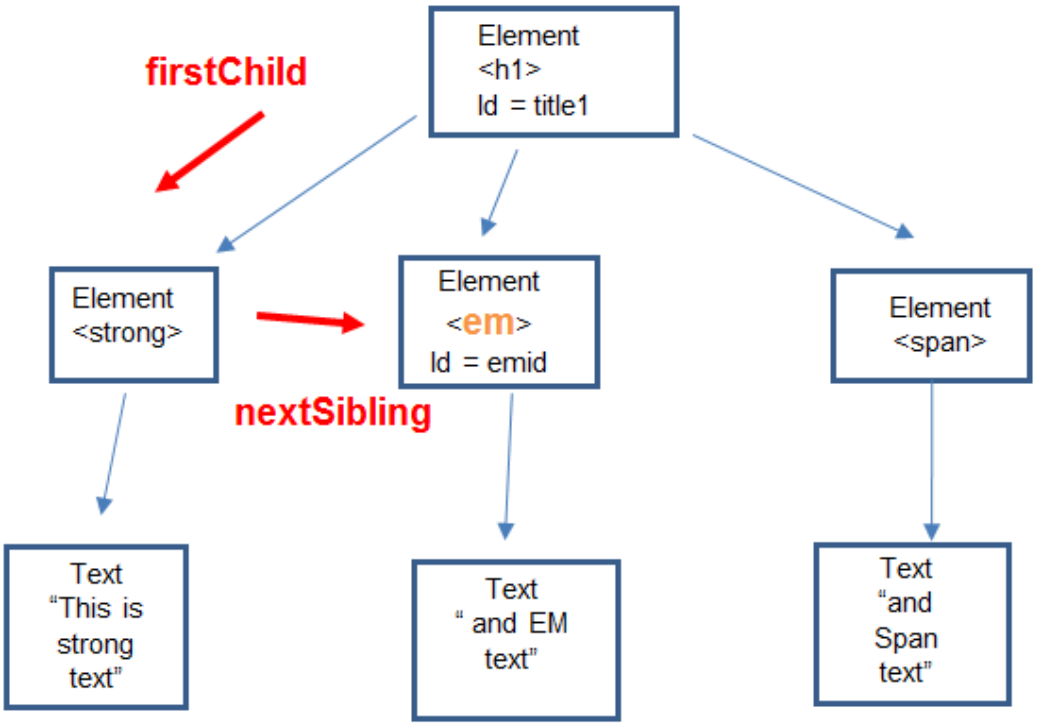
document.getElementById ('title1').lastChild.nodeName



Example 3

Get next sibling of first child

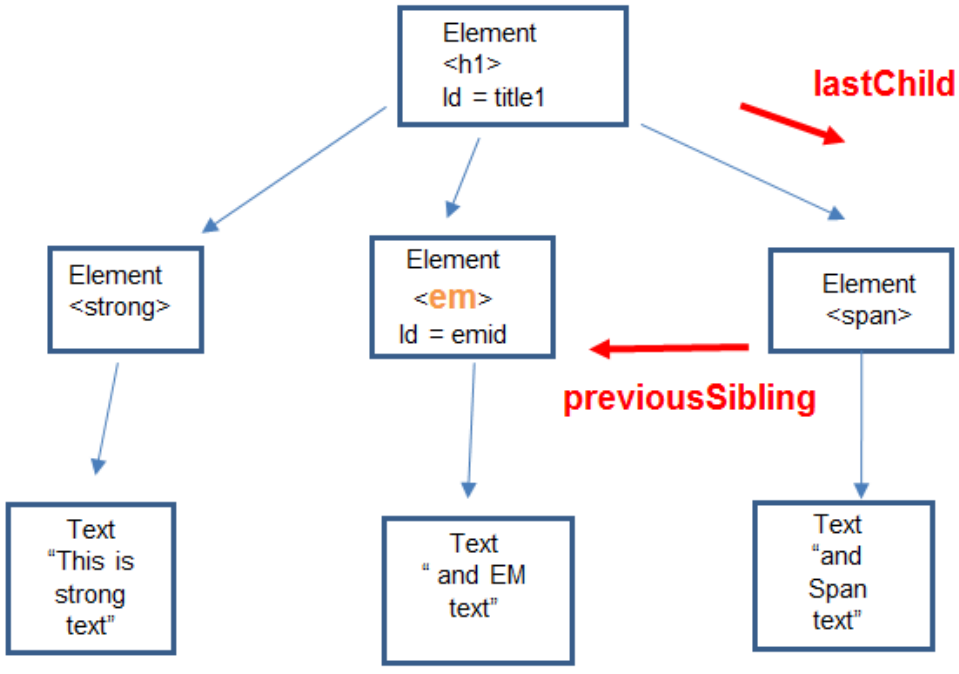
```
document.getElementById('title1').firstChild.nextSibling.nodeName;
```



Example 4

Get previous sibling of last child

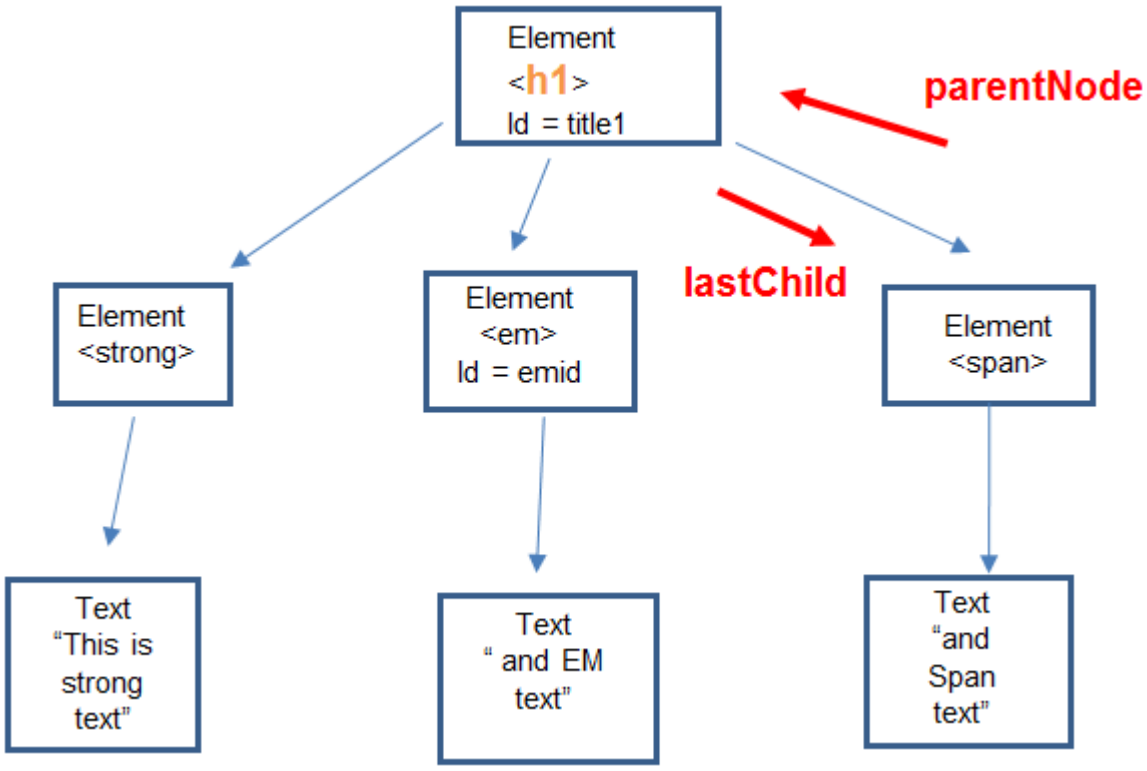
```
document.getElementById('title1').lastChild.previousSibling.nodeName
```



Example 5

Get parent node name of last child

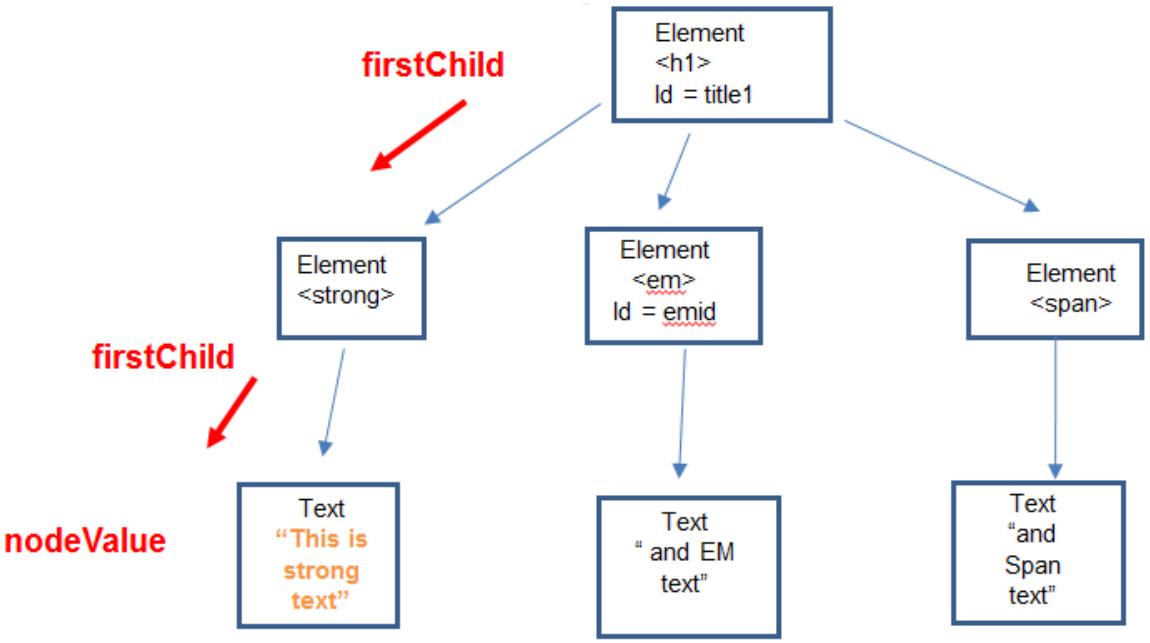
```
document.getElementById('title1').lastChild.parentNode.nodeName;
```



Example 6

Get number of child nodes

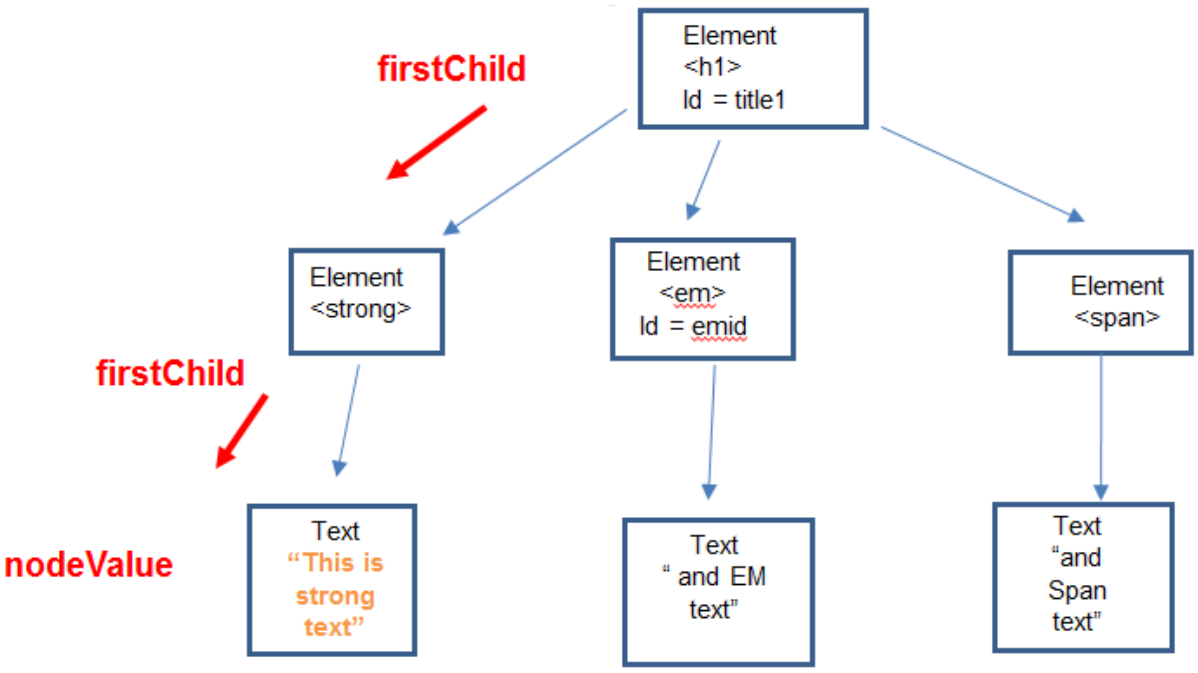
```
document.getElementById('title1').childNodes.length;
```



Example 7

Get value of text node under first child – need to go down another level – so need an extra firstChild

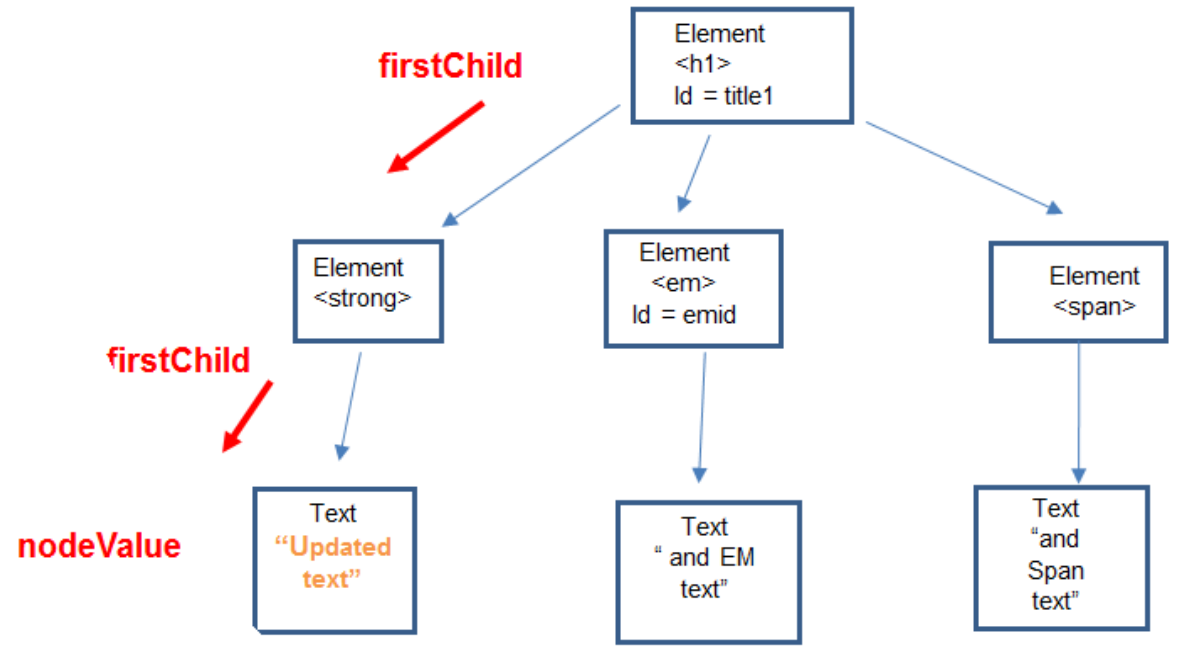
```
document.getElementById('title1').firstChild.firstChild.nodeValue;
```



Example 8

Update the text node in Ex 7 to now be "Updated Text"

```
document.getElementById('title1').firstChild.firstChild.nodeValue = "Updated Text ";
```

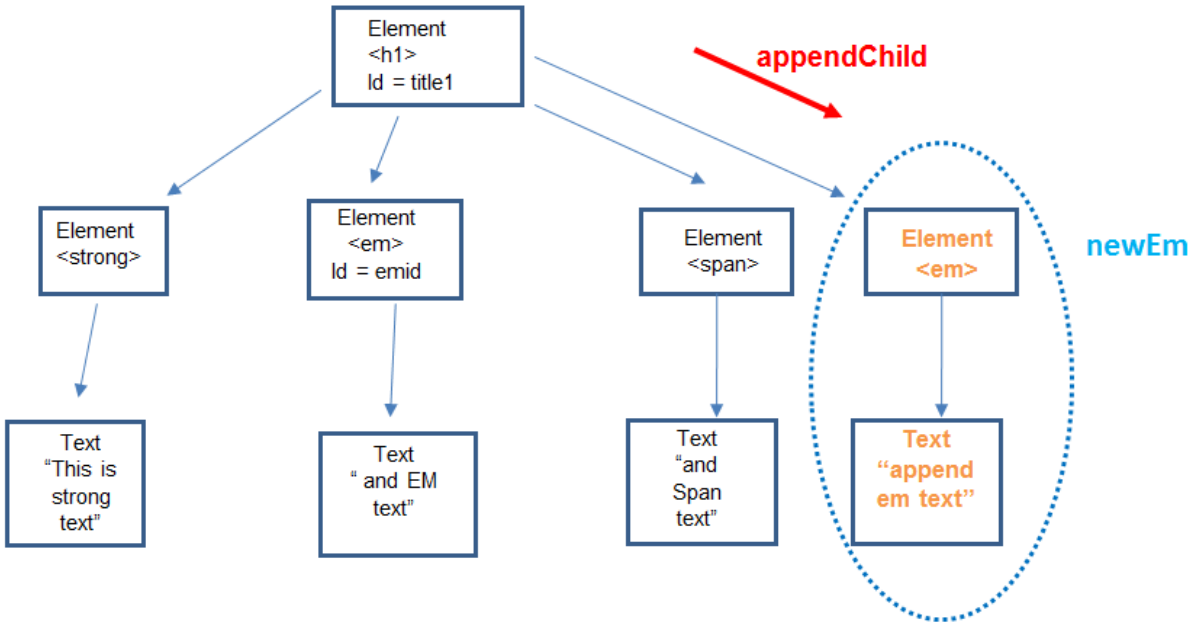


Example 9a

To append a new ‘em’ element at the end

- using AppendChild

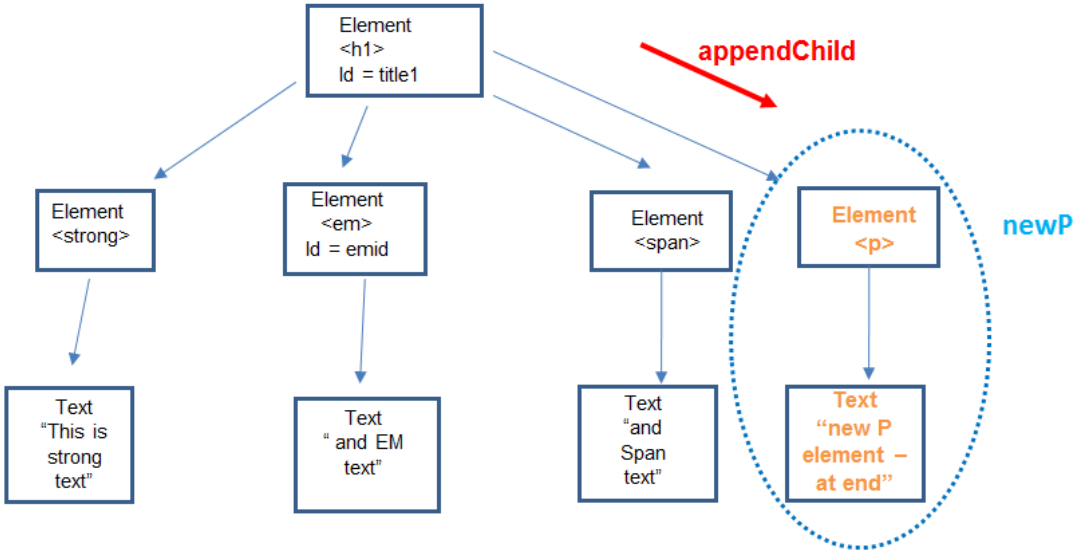
```
var newEm = document.createElement(‘em');
newEm.innerHTML = “append em text";
document.getElementById('title1').appendChild (newEm);
```



Example 9b

To append a new ‘p’ element at the end - using AppendChild

```
var newP = document.createElement('p');
newP.innerHTML = “new P element – at end";
document.getElementById('title1').appendChild( newP );
```



Additional Info

w3schools - HTML DOM Document Object

- http://www.w3schools.com/jsref/dom_obj_document.asp

w3schools - JavaScript HTML DOM Examples

- http://www.w3schools.com/JS/js_ex_dom.asp