

JavaScript

Using HTML Script Tag

`<script>`

Script code here

`</script>`

Use to tell the browser the beginning and ending point of scripting language in HTML Doc.

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

JavaScript code here

`</script>`

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

`</script>`

<script> location

- Any number of <script> are allowed in the HTML document
- <script> can be placed in <head> or <body> or both
- Trick: placing scripts at the bottom of <body> improves speed of page rendering

External JavaScript

- In the external JavaScript file cannot contain `<script>`
- Advantages:
 - HTML and JavaScript are separated physically
 - Easier to maintain
 - Proxy server or browser caches can store frequently used JS file – speed up page loading

Variables

- Declaring variables: *var* keyword
- Variables are case sensitive
- Avoid using reserved as variables name
- The variable values (or type) can include number, string, Boolean and null
- JavaScript allows virtually any value to be assigned to any variable
- Special characters can be used in string type variables (ex. \t, \n, \\, \", \')

Variables

- Examples

```
var web;
```

```
var str="web technology";
```

```
var str1='web technology';
```

```
var x=120;
```

```
var code=true;
```

```
var t=null;
```

```
y=200.5;
```

Hoisting Behavior

- Hoisting is the behavior of moving all declarations to the top of the current scope
 - All variables in the scope can be used right from the start of the scope (before the declaration of variables)
 - Only variables declared with var keyword
- Keyword 'let' doesn't have this behavior

Functions

- Declaring function

```
function functionname()  
{  
    code  
}
```

- Function names are case sensitive
- The function name must begin with a letter or underscore and cannot contain any space

Functions

- Functions can have one or more parameters

```
function func1(var1, var2)
```

```
{ document.write ("var1="+var1+"", var2="+var2); }
```

Nameless function

- Sometimes called anonymous function
- Function without name
 - Ex: `(function() { ... });`
- Usage:
 - Immediately invoked function
 - `<button onclick="(function() { alert('Hello World'); })();">Click</button>`
 - Using anonymous functions as arguments
 - Assign the function to var. for calling later
 - `let test = function() { alert('Hello World'); }; test();`
- Arrow function is a shorthand for declaring anonymous function
 - `let test = () => alert('Hello World');`

Operators

- Mathematical Operators
 - +, -, *, /, %, ++, --
- Assignment Operators
 - =, +=, -=, *=, /=, %=
- Comparison Operators
 - ==, !=, >, <, >=, <=, ===, !==
- Logical Operators
 - &&, ||, !, &, |, ^, >>, >>>, <<

Conditional Statements

- if/else

```
If (condition) {  
    javascript statement  
}  
  
else {  
    javascript statement  
}
```

Conditional Statements

- switch

```
switch(varname) {  
    case "X":  
        javascript statement;  
        break;  
    case "Y":  
        javascript statement;  
        break;  
    default:  
        javascript statement;    }
```

Loops

- for
- while
- do ... while

Event Handlers

- Event is something that happens when viewer of the page perform some actions such as clicking a mouse button
- Event Handlers can be used to identify the occurring event and then perform a task or a set of task
- With Event Handler, the page can react to the action of the viewer

Event Handler

- Using event handler in an HTML element

```
<input type="button" value="Click Me!" onclick="JavaScript code here" />
```

Example

```
<body>
```

```
<form>
```

```
<input type="button" value="Click Me!"
```

```
onclick="window.alert('Hi!');window.alert('Bye!');" />
```

```
</form>
```

```
</body>
```


Js_event_01.js

```
function hi_and_bye() {  
    window.alert('Hi!');  
    window.alert('Bye!');  
}
```

```
<body>  
<form>  
<input type="button" value="Click Me!" onclick="hi_and_bye();" />  
</form>  
<script type="text/javascript" src="js_event_01.js"></script>  
</body>
```

Js_event_01.js

```
function hi_and_bye() {
```

```
    window.alert('Hi!');
```

```
    window.alert('Bye!');
```

```
}
```

```
var hi_button = document.getElementById("say_hi");
```

```
hi_button.onclick = hi_and_bye;
```

```
<body>
```

```
<form>
```

```
<input type="button" value="Click Me!" id="say_hi" />
```

```
</form>
```

```
<script type="text/javascript" src="js_event_01.js"></script>
```

```
</body>
```

Event Handlers

- The blur event: onblur

Example

```
<form>
```

```
<input type="text" onblur="window.alert('Hey! Come back!');" /><br />
```

```
<input type="text" />
```

```
</form>
```

Event Handlers

- The click event: onclick

Example

```
<body>
```

```
<form>
```

```
<input type="button" value="Do not Click Here"
```

```
onclick="window.alert('I told you not to click me!');">
```

```
</form>
```

```
</body>
```

Event Handlers

- The click event: onclick

Example

```
<body>
```

```
<a href="http://www.kmitl.ac.th" onclick="return false;">
```

```
Click me</a>
```

```
</body>
```

Event Handlers

- The focus event: onfocus

Example

<form>

Enter Your Name:

```
<input type="text" onfocus="window.alert('Don\'t forget to  
capitalize!');" />  
</form>
```

Event Handlers

- The mouse over event: onmouseover

Example

```
<a href="http://www.kmitl.ac.th"
```

```
onmouseover="window.alert('mouse over');">
```

```
Try Clicking Me!</a>
```

Event Handlers

- The submit event: onsubmit

Example

```
<form onsubmit="window.alert('Thank You');">
```

```
What's your name?<br />
```

```
<input type="text" id="thename" /><br />
```

```
<input type="submit" value="Submit Form">
```

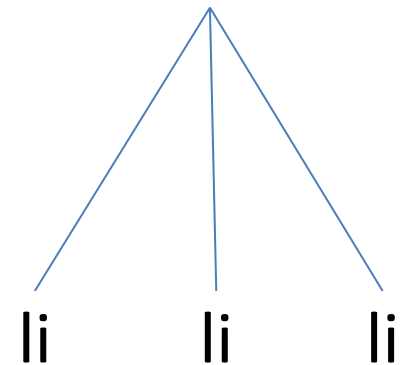
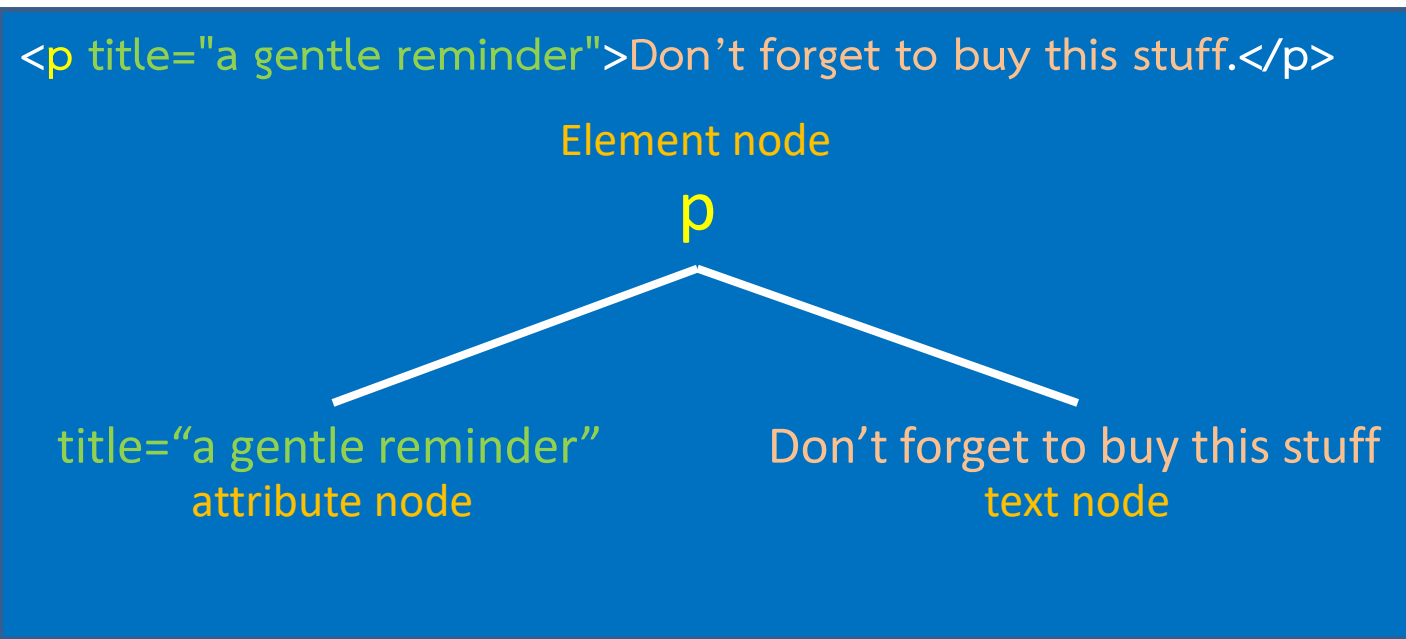
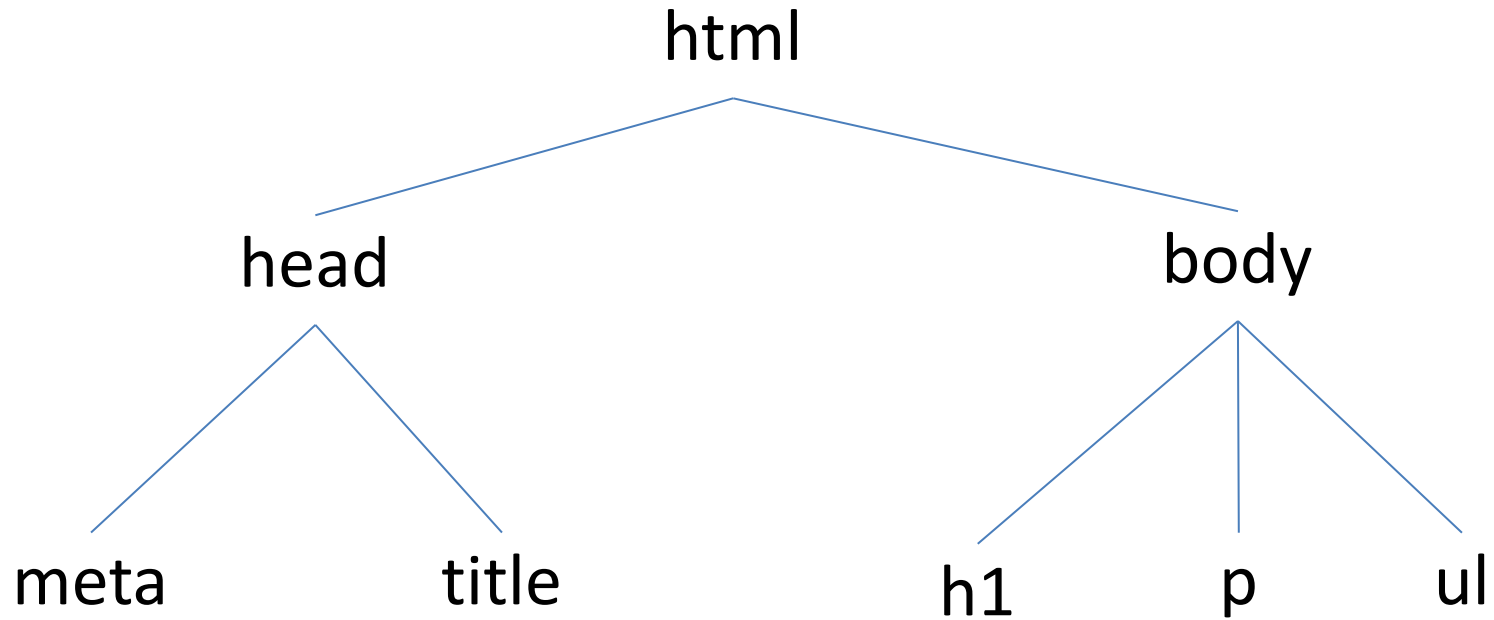
```
</form>
```


DOM

- DOM: Document Object Model
- DOM represents a document as a family tree

Example

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
  <meta http-equiv="content-type" content="text/html; charset=utf-8" />
  <title>Shopping list</title>
</head>
<body>
  <h1>What to buy</h1>
  <p title="a gentle reminder">Don't forget to buy this stuff.</p>
  <ul id="purchases">
    <li>A tin of beans</li>
    <li>Cheese</li>
    <li>Milk</li>
  </ul>
</body>
</html>
```



Working with Document Object

- Properties of Document Object
 - Image property
 - Form property
- Method of Document Object
 - getElementById
 - getElementsByClassName
 - getElementsByTagName

The getElementById() method

- Allow access to an element by the value of its id attribute

Example

```
<div id="some_text">This is some text.</div>
```

```
var text_element = document.getElementById("some_text");
```

The `getElementsByClassName()` method

The `getElementsByTagName()` method

- Get an array filled with all the elements in the document that have specified class/tag name

Document node creation

```
<body>
```

```
<div id="div1" title="All about me!">
```

This page is about me, me, and... me!

```
</div>
```

```
</body>
```

```
var me_div = document.getElementById("div1");  
var inner_div = document.createElement("div"); <div></div>  
var inner_div_text = document.createTextNode("More...");  
inner_div.appendChild(inner_div_text); <div>More...</div>  
me_div.appendChild(inner_div);
```

Result

```
<body>
```

```
<div id="div1" title="All about me!">
```

This page is about me, me, and... me!

```
<div>
```

More...

```
</div>
```

```
</div>
```

```
</body>
```


getAttribute() method

- Get the values of attribute of an element

Example

```

```

```
var i_id = document.getElementById("i1");  
var i_src = i_id.getAttribute("src");
```

setAttribute() method

- Set the values of attribute of an element

Example

```

```

```
var i_id = document.getElementById("i1");  
i_id.setAttribute("src", "images/xx.jpg");
```

Js_event_01.js

```
function test(v) {  
    window.alert(v.getAttribute("type"));  
}
```

```
<body>  
<form>  
<input type="button" value="Click Me!" onclick="test(this);" />  
</form>  
<script type="text/javascript" src="js_event_01.js"></script>  
</body>
```