JavaScript and the Document Object Model

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

OBJECTIVES

In this unit, the student will learn:

- how to write *unobtrusive JavaScript*
- what the HTML Document Object Model (DOM) is
- how to manipulate DOM elements in JavaScript
- how to change content and behavior of a web page dynamically
- how to change style of web page elements dynamically

Note: This unit only covers client-side functionality. Later we will see how to use the DOM with AJAX, involving the server.

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Unobtrusive JavaScript

Good programming style puts all the JavaScript into a separate file, not mingled with the HTML. This style is called `unobtrusive JavaScript.`

Obtrusive:

JS: `function goDoIt(){ ... }`

HTML: `<button onClick="goDoIt();">Go</button>`

Unobtrusive:

JS: `function goDoIt(){ ... }`

...

```
document.getElementById("goButton").onclick = goDoIt;
```

HTML: `<button id="goButton">Go</button>`

Note: no `()` after the function name when attaching as event handler in JS.

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Deferring execution

Problem: JS code runs as soon as it is loaded from `script` tag. But the code

```
document.getElementById("goButton").onclick = goDoIt;
```

cannot run that early.

- the `goButton` element does not yet exist
- need to defer execution till after page is loaded

Solution: create a function attached to `window.onload` event that performs the assignment. ▸ Example ▸ JS

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Anonymous functions

JavaScript allows unnamed functions to be assigned as event handlers. Simpler code:

```
window.onload = function ()
{
  // attach event handler
  document.getElementById("goButton").onclick = goDoIt;
}
```

Also note: event names are all lowercase, not camelCase like most variable and function names: `onload` and `onchange` not `onLoad` and `onChange`.

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Selecting an element

Functions belonging to `document` to select a specific element or set of elements:

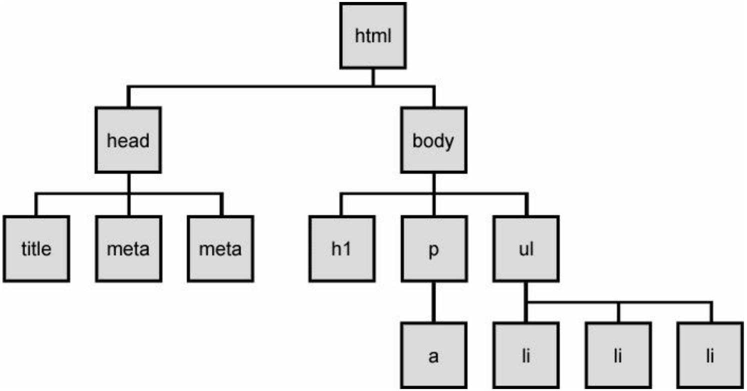
- **`document.getElementById(`*`id`*`)`**
  - element with attribute `id="`*`id`*`"`
- **`document.getElementsByName(`*`name`*`)`**
  - list of all elements with attribute `name="`*`name`*`"`
- **`document.querySelector(`*`selector`*`)`**
  - first element matching CSS selector (HTML5)
- **`document.querySelectorAll(`*`selector`*`)`**
  - list of all elements matching CSS selector (HTML5)

JavaScript and the Document Object Model

OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

The Document Object Model

Elements of a page form a hierarchy, the DOM tree:



Acknowledgement: image taken from Stepp et al. online slides, Chapter 9.

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Types of DOM nodes

The main DOM node types are:

- Element node
  - corresponds to HTML tag
  - can have element, text, and attribute child nodes
- Text node
  - textual content of an element
  - is child of that element
  - cannot have child nodes or attributes
- Attribute node
  - attribute/value pair
  - is child of element
  - can have text as child

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

The Node object

DOM nodes are objects. Their properties include:

- `className` — list of CSS classes of element
- `innerHTML` — content inside element, including tags
- `parentNode` — parent of node
- `firstChild` — first child of node
- and many more, some depending on type of node

These properties can be accessed and changed using JavaScript.

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

DOM properties of form fields

An element that is a form field has properties including

- **`value`** — string, the text in an `input` or `textarea` field
- **`selectedIndex`** — integer, index of selected option in a `select` list (numbering starts at 0).
- **`checked`** — boolean, if a box is checked
- **`disabled`** — boolean, if a field is inactive

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Ways to access a form field

With introduction of `id` attribute, there are now two ways to get at a form field in JavaScript.

Old way: via sequence of child references:
`document.tempform.degreesF.value`
Requires form to have `name="tempform"` and input element to have `name="degreesF"`.

▸ Example ▸ JS

New way: via `id`:
`document.getElementById("degreesF").value`
Requires element to have `id="degreesF"`.

▸ Example ▸ JS

Note that form processor on server accesses field only via `name`. Customary practice is to give each form field both a `name` and `id` attribute, with same value.

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Modifying content with the DOM

Giving a DOM block element new text content:

```
elem.innerHTML = "Hello, world!";
```

Here is how to implement one of the earlier examples that used `document.write()` to display the current date and time, now using modern DOM methods.

▸ Example ▸ JS

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Modifying content, cont'd

The **`innerHTML`** property accepts text as well as HTML tags, e.g.:

```
elem.innerHTML = "text <a href="page.html">link</a>";
```

▸ Example ▸ JS

But this is **BAD!** It can inject arbitrary and incorrect HTML content into the page. It mingles style with content, and is prone to errors and bugs*. **Don't do it!** Only assign plain text content.

There are other DOM techniques to achieve the same effect in cleaner ways.

*The example in green above has a mistake! Can you spot it?

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Modifying styles with the DOM

So we want to modify the style as well as the content of an element. Here's a way: use the `style` property.

▸ Example ▸ JS

Note:

- CSS style property names with hyphens become camelCased
- CSS property values are the same as in CSS, but enclosed in quotes

CSS `font-weight: bold;`
becomes
JS `elem.style.fontWeight = "bold";`

JavaScript and the Document Object Model

OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Adding new elements

OK, but we wanted only the last part of the sentence to be in red. How do we get there without putting tags in the `innerHTML`? Answer:

An element of the page can be given new children using these DOM methods:

- **`document.createElement("`*`type`*`")`** — create a new element node of the given type: any tag name
- **`document.createTextNode("`*`string`*`")`** — create a new text node containing the string
- **`elem.appendChild(`*`childElem`*`)`** — add child node as last child of `elem`.

▸ Example ▸ JS

View this page using Firefox DOM Inspector. Click on the elements of the body to open them and see the children created by JS. These do not appear using View Source.

JavaScript and the Document Object Model

OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

createTextNode vs. innerHTML vs. innerText vs. textContent

Why use `createTextNode(`*`string`*`)` instead of `innerHTML = `*`string`*?

- `innerHTML` can contain arbitrary tags as well as text.
- This is OK if source of text is trusted (e.g. string literal in the JS). Otherwise risks injection attack.
- Use `createTextNode` if source is untrusted, e.g. data from a form field.
- `innerText = `*`string`* also escapes tags, but is not in W3C standard, not supported by Firefox.
- `textContent = `*`string`* is W3C standard, but not supported by IE < 9.

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Unobtrusive style

The foregoing is OK, but it still mixes CSS and JS. Best practice avoids putting CSS into JavaScript code.

How: set the `className` property and create a CSS rule for that class.
JS: `elem.className = "redtext";`
CSS: `.redtext { color: red; }`

▸ Example ▸ JS ▸ CSS

Note: an element may belong to more than one class.

- `className` is list of classes separated by blank spaces.
- Managing them gets messy:
    - Best use jQuery methods, treated later.

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Dynamic style

Now, let's put the DOM techniques together with event-driven programming to make a page whose content styles change dynamically in response to events.
In this example we introduce a few more new things:

- `document.getElementsByTagName("tag")` — returns list of all elements with specified tag name.
- Traversal of a list of elements (a NodeList object)
- The `this` object — refers to element that fired the event being handled.

▸ Example ▸ JS ▸ CSS

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Checking a form before submitting

Now let's revisit the example from previous unit, using JavaScript to make sure that a form has been properly filled in before submitting it.
This example uses DOM techniques to access the form fields and their labels. It uses style changes to highlight the erroneous fields. It does away with the alerts in favor of an error box.

▸ Example ▸ JS ▸ CSS

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Traversing the DOM tree

Every DOM element has these properties for moving around in the DOM tree

- `childNodes` — array of children of the element
- `firstChild`, `lastChild` — first and last members of `childNodes` list.
- `nextSibling`, `previousSibling` — previous and next node with same parent.
- `parentNode` — element that has this node as a child.

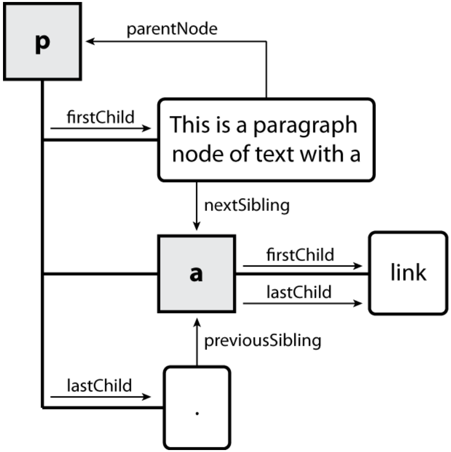
Older browsers sometimes construct the DOM tree slightly differently. See browser compatibility.

▸ Example ▸ JS

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

DOM tree traversal example

```
<p id="foo">This is a paragraph of text with a
<a href="/path/to/another/page.html">link</a>.</p>
```



Example taken from Stepp et al. online slides, Chapter 9. Gray boxes are element nodes, white boxes are text nodes.

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Traversing the DOM tree compatibly

The following code (from w3schools.com) shows how to skip the text nodes that some browsers create and other browsers do not.

```
function get_nextSibling(n)
{
  y=n.nextSibling;
  while (y.nodeType!=1) // element nodes are type 1
  {
     y=y.nextSibling;
  }
  return y;
}
```

JavaScript and the Document Object Model

OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree

Modifying the DOM tree

We have already seen how to create and append new child nodes. There are also methods for deleting or replacing children.

Every DOM element object also these methods:

- **`elem.appendChild(node)`** — add child node as last child of `elem`.
- **`elem.insertBefore(new, old)`** — add new node as child in front of old node
- **`elem.removeChild(node)`** — remove the given child node of element.
- **`elem.replaceChild(new, old)`** — replaces the old child with the new one.

▸ Example ▸ JS

JavaScript and the Document Object Model
OBJECTIVES
Unobtrusive JavaScript
Deferring execution
Anonymous functions
Selecting an element
The Document Object Model
Types of DOM nodes
The Node object
DOM properties of form fields
Ways to access a form field
Modifying content with the DOM
Modifying content, cont'd
Modifying styles with the DOM
Adding new elements
createTextNode vs. innerHTML vs. innerText vs. textContent
Unobtrusive style
Dynamic style
Checking a form before submitting
Traversing the DOM tree
DOM tree traversal example
Traversing the DOM tree compatibly
Modifying the DOM tree