



JavaScript and jQuery Course

Instructor Teresa Pelkie

Session 4

Topic: How to use objects to work with data

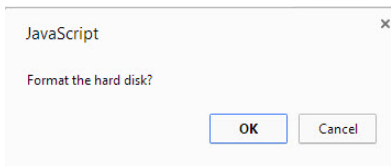
Chapter 3: How to work with objects, functions, events — pages 92-99

Windows Object — top level object

- alert() method
- prompt() method
- confirm() method, **new in this chapter**

confirm() method - page 93

The confirm() method displays a dialog box with a specified message, along with an **OK** and **Cancel** button. A confirm box is used if you want the user to verify or accept something. The confirm() method returns **true** if the user clicks **OK**, and **false** otherwise.



```
var response = confirm('Format the hard disk?');
if(response == true)
{
    alert('You clicked OK!');
}
else
{
    alert('You clicked Cancel!');
}
```

Document Object - top level object in the Document Object Model (DOM) - page 93

See example files in sample code that start with 04_DOM

Three methods that we will look at:

<code>document.write()</code>	Obsolete for writing in <body> section
<code>document.writeln()</code>	
<code>document.getElementById(id_value)</code>	Most popular way to access an element

Textbox as an object - page 95

In JavaScript, all form elements, or any HTML element, can be accessed as an object by using `document.getElementById(id_value)`. That means that we need to assign a unique id attribute for each element. If you need a refresher on HTML forms see

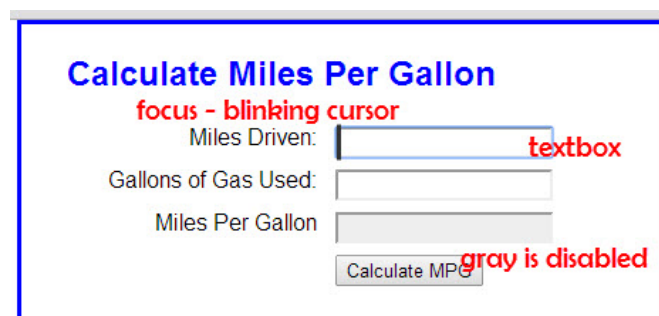
http://www.w3schools.com/html/html_forms.asp

Method of textbox

- `focus()` - gives input to and is displayed as a blinking cursor

Properties of textbox

- `value` - the string the user entered
- `disabled` - a boolean true / false value to disable user input



How to retrieve a value from a textbox - page 95

Without chaining

chaining means combining several methods at once

```
var firstName = document.getElementById("first_name"); // access object first
firstName = firstName.value;                          // access value
alert(firstName);                                     // display value
```

With chaining

```
// access the object and its value in one statement
var firstName = document.getElementById("first_name").value;
alert(firstName);
```

Retrieving a numeric value from a textbox - page 95

A textbox **always returns a string**, so you will need to convert a numeric string to a number:

- `parseInt()`
- `parseFloat()`

Do I need to use `<form>` `</form>` when using form elements just for user input?

If all you are doing is providing a user interface for interactions, then you can put the form element into the HTML without including the `<form>` `</form>` tags, as shown in the example in Chapter 3.

You should use the `<form>` `</form>` tags if:

- You want to execute a traditional form method (post or get)
- You want to capture the "submit" event programmatically
- You want to programmatically "reset" a form
- You want to separate fields with the same name

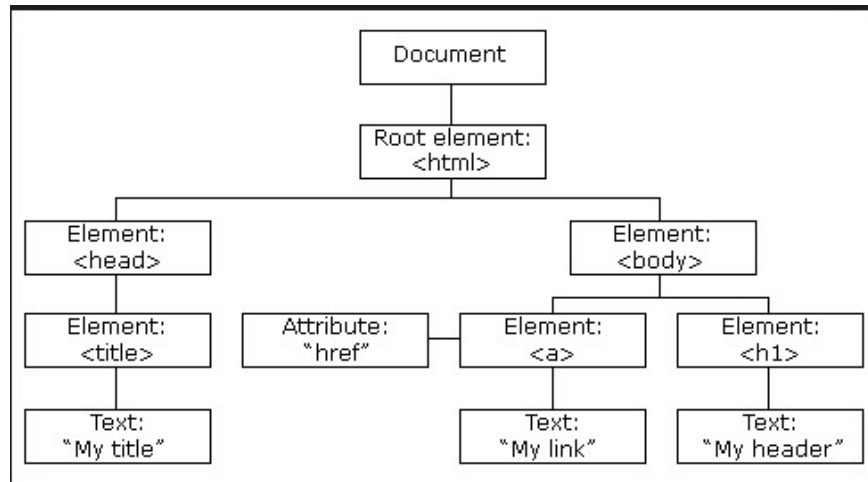
Using the DOM to change the text of an element - pages 98-99

The Document Object Model

All HTML documents are *trees*. Each level of the tree is described in the same manner as a human family tree, with ancestors, descendants, parents, children and siblings. In JavaScript all parts of the tree are referred to as **nodes**.

Types of nodes:

- element nodes - accessed by `getElementById()`
- text nodes - accessed by `firstChild.nodeValue`
- attribute nodes
- comment nodes



```

<html>
  <head>
    <title>My title</title>
  </head>
  <body>
    <a href="">My Link</a>
    <h1>My header</h1>
  </body>

```

How to modify the text of an HTML element

- access that element by creating an object reference to it

```
document.getElementById('id')
```

- access the text node of that element by using the **firstChild** property

```
document.getElementById('id').firstChild
```

- use the **nodeValue** property to access the text for that text node

```
document.getElementById('id').firstChild.nodeValue
```

- change it to what you wish

```
document.getElementById('id').firstChild.nodeValue = "New Content Here";
```

Why not just use the innerHTML property?

```
document.getElementById('id').innerHTML = "New Content Here";
```

The `innerHTML` property is technically not part of the DOM - it is part of the JavaScript language, although you commonly see it being used along with the DOM.

We want to use the DOM correctly at this point!

Can I add CSS to the nodeValue property?

No. This is how you change the CSS using the DOM:

```
document.getElementById('id').style.property = new style  
// note: use the JavaScript syntax in the CSS  
document.getElementById('id').style.backgroundColor = "yellow";
```

Can I add HTML to the nodeValue property?

No. It is a bit more complicated and we will cover this later.

Date() Object - pages 96-97

The `Date()` object provides information about the day, the day number in the month, month, year, hours, minutes, seconds, and milliseconds. You can retrieve the date/time, create a specific date/time, and perform date and time calculations using the date object.

The date object contains information about the date and time, **based on the date and time of the client computer (not of the web server where the page was served from)**.

The date object begins at January 1, 1970 at midnight (00:00:00) GMT (Greenwich Mean Time)

How to create a date object

```
var today = new Date() // note this is uppercase D
```

The date object returns the following information **based on today's date and time**:

day, month, date, year, time, time zone

Sat Jul 14 2012 16:49:12 GMT-0700 (Pacific Daylight Time)

Note: it is also possible to create a date object for a specific date and time (pages 172-174).

Methods of the Date object - page 97

<code>today.getDate()</code>	Returns the day number of the month (1 - 31) <code>var monthDay = today.getDate();</code>
<code>today.getDay()</code>	Returns an integer for the day of the week Sunday = 0, Monday = 1, ... Saturday = 6 <code>var dayofweek = today.getDay();</code>
<code>today.getMonth()</code>	Returns an integer for the month January = 0, ... December = 11 <code>var month = today.getMonth();</code>
<code>today.getFullYear()</code>	Returns the year as 4 digits <code>var fullYear = today.getFullYear();</code>
<code>today.getYear()</code>	Returns the year as 2 digits <code>var year = today.getYear();</code>

More Methods of the Date object - Not in textbook

<code>today.getTime()</code>	Returns milliseconds since midnight 1/1/1970 <code>var time = today.getTime();</code>
<code>today.getHours()</code>	Returns the current hour, 0 to 23, universal time <code>var currentHour = today.getHours();</code>
<code>today.getMinutes()</code>	Returns the current minute, 0 to 59 <code>var currentMinute = today.getMinutes();</code>
<code>today.getSeconds()</code>	Returns the current second, 0 to 59 <code>var currentSecond = today.getSeconds();</code>
<code>today.getMilliseconds()</code>	Returns the current millisecond, 0 to 999 <code>var currentMilliseconds = today.getMilliseconds();</code>

See http://www.w3schools.com/jsref/jsref_obj_date.asp

More Methods of the Date object

You can set a date in the past or future in order to perform calculations

<code>setDate()</code>	Sets the day of the month of a date object
<code>setFullYear()</code>	Sets the year (four digits) of a date object
<code>setHours()</code>	Sets the hour of a date object
<code>setMilliseconds()</code>	Sets the milliseconds of a date object
<code>setMinutes()</code>	Set the minutes of a date object
<code>setMonth()</code>	Sets the month of a date object
<code>setSeconds()</code>	Sets the seconds of a date object
<code>setTime()</code>	Sets a date and time by adding or subtracting a specified number of milliseconds to/from midnight January 1, 1970

Date Calculations

You can do calculations with the dates, such as determining the number of days until Christmas or any other special day.

The date object starts at January 1, 1970, at Midnight (00:00:00) GMT. Each date object actually hold the number of milliseconds that have elapsed from that time until the date specified in the date object.

- There are 86,400,000 milliseconds in a day.
- To determine the number of days, divide the number of milliseconds in the date object by 86,400,000. You will probably get a decimal value.
- To round the decimal value to a whole number, use the `Math.round()` method.
- That will give you the number of days that have passed since January 1, 1970 for one date
- Get the number of days that have passed since January 1, 1970 for the second date
- Subtract the two dates
- That will tell you the difference between them, in the number of days.

Date calculations are performed using milliseconds

Millisecond Equivalents:

one second = 1,000 milliseconds

one minute = 1,000 * 60 (60 seconds in a minute) = 60,000 milliseconds

one hour = 60,000 * 60 (60 minutes in an hour) = 3,600,000 milliseconds

one day = 3,600,000 * 24 (24 hours in a day) = 86,400,000 milliseconds

one week = 86,400,000 * 7 (7 days in a week) = 604,800,000 milliseconds

one day = 1000 * 60 * 60 * 24 = 86,400,000 milliseconds

Date Object Video

<http://www.sights.com/csis138/video/class07-date/class07-date.html>

This is a video I created in 2008. I am referencing files you do not have but it provides an outline of the basics.

The String Object - pages 68-69 and page 97

Any variable that contains text or number to be treated as text is a string and is also a string object. **Note that this is uppercase "S".**

Any variable that contains text or a literal is automatically a string object, which means that we can access its properties and methods.

See handout bit.ly/js-02 for additional coverage of the string object that is not in the textbook

Methods of the String Object

- `indexOf(search_string, position_to_start_from)`

returns the position of the first occurrence of a search string, starting at the indicated index position. If no position is given, it starts at 0. This method returns -1 if the value to search for never occurs

- `substr(start_position, length_to_extract)`

extracts the specified number of characters from a string, beginning at a specified index position, if number of characters is not given, it extracts to the end of the string

- `toLowerCase()` - converts a string to lowercase letters
- `toUpperCase()` - converts a string to uppercase letters

Property of the String Object

- `length` - determines the number of characters in a string starting at 0

Sample Files for this session

File Name	Description
04_confirm.html	Page that shows an example of the JavaScript confirm() function.
04_date.html	The Date object and some of its methods.
04_date_calculation_days_epoch.html	Date object, calculating the number of days since the start of the epoch.
04_date_calculation_days_epoch2.html	Date object, calculating the number of milliseconds since the start of the epoch.
04_date_calculation_days_until_NewYear.html	Date object, calculate the number of days until the next New Year's day.
04_date_complete.html	Date object, methods to format a complete date string.
04_date_complete2.html	Date object, methods to format a complete date string. Shows an alternative way to declare/initialize an array, use of the window.onload function.
04_date_other_ways_to_display.html	Date object, displayed in different formats.
04_DOM_change_text_node_value.html	Change text on a page using the DOM.
04_DOM_change_text_node_value2.html	Change text on a page using the DOM.
04_DOM_change_text_node_value_and_css.html	Change text on a page using the DOM, also change background color.
04_DOM_text_box_retrieve_value.html	Retrieve a value from a text box object without "chaining".
04_DOM_text_box_retrieve_value_chaining.html	Retrieve a value from a text box object with "chaining".
04_function_creating_clock.html	Displays a digital clock.
04_get_day_name_array.html	Get the day name, using an array of day names.
04_get_day_name_if_condition.html	Get the day name, using an if/else if sequence.
04_get_month_name_array.html	Get the month name, using an array of month names.
04_get_month_name_if_condition.html	Get the month name, using an if/else if sequence.
04_set_new_date.html	Initialize a Date object to a given date.
04_string_indexOf.html	Use String.indexOf() to find the location of one string in another.
04_string_length.html	Use the String.length property to determine the length of a string.

04_string_substr.html	Use the <code>string.substr()</code> method to extract part of a string from a string.
04_string_toUpperCase.html	Use the <code>string.toUpperCase()</code> method to convert all of the letters in a string to uppercae.
04_time_methods_for_getting_time.html	Examples of getting the time value.
04_time_methods_for_getting_time_ with_formatting.html	Examples of getting and formatting the time value.