

# Corso : Tecnologie web AJAX    Febbraio/Marzo/Aprile 2009

**Prima Parte    :    Fondamenti di JAVASCRIPT**

**Prof. Sergio Cordella**

## **4^ Lezione**

Riferimenti web - contenuti adattati tratti da :

<http://web.tiscali.it/genero/javascript/>

Riferimenti bibliografici - contenuti adattati tratti da :

HTML.CSS,JAVASCRIPT - Gallo, Salerno Ed. Minerva Italiana

## **OOP - Object Oriented Programming e il JAVASCRIPT**

In questa lezione parleremo della programmazione ad oggetti e come usarla con il Javascript.

Fino adesso abbiamo operato su variabili che potevano contenere solo un valore sia esso numero, stringa o valore booleano. La programmazione orientata agli oggetti è nata per venire in aiuto ai programmatori al fine di semplificare la costruzione di un programma. Infatti, è possibile raccogliere una serie di dati, eterogenei per quanto riguarda il tipo, sotto un unico nome.

I vantaggi della programmazione ad oggetti **non vengono evidenziati** quando si scrivono **programmi di dimensioni ridotte**. Anzi potrebbe sembrare che porti ad una complicazione eccessiva rispetto al problema da risolvere. Per imparare a programmare ad oggetti **è però necessario partire da esempi semplici**. I vantaggi della OOP saranno evidenti quando vi troverete di fronte a problemi di una certa complessità.

E' utile ricordare che quando si programma ad oggetti, è ancora importante utilizzare i principi della programmazione strutturata (vedi capitoli precedenti). Cambia solo la visione del problema.

### **Le classi gli oggetti e il JAVASCRIPT**

#### **Che cosa è un oggetto**

Consideriamo l' **uomo**.

Il mondo uomo è composto da diversi **oggetti** (che in pratica sono gli uomini).

- L' oggetto uomo possiede diverse caratteristiche (**capelli, occhi, gambe, ecc.**).
- Ogni caratteristica può avere un diverso stato (**i capelli possono biondi o neri, gli occhi possono essere aperti o chiusi o azzurri, le gambe possono ferme o in movimento, etc...**).
- L'insieme degli uomini è definito **CLASSE**.
- Ogni singolo uomo è definito **OGGETTO**.
- Ogni oggetto (uomo) ha delle caratteristiche chiamate **PROPRIETA'**.
- Ogni proprietà (ex: capelli) può avere un diverso stato definito **METODO**.

Se noi volessimo, con un programma orientato agli oggetti (OOP), far camminare un uomo dovremmo scrivere:

- **uomo.gambe.camminano()**

All' interno delle parentesi tonde potremmo mettere il numero dei passi che le gambe devono fare.

- **uomo.capelli.colore(biondi)**

provate ad indovinare cosa definirebbe questo comando...

Ricapitolando: **l'elemento base della programmazione Object-Oriented è l'oggetto.** Per capire cosa si intende per oggetto si può partire prendendo spunto dalle cose della vita comune:

**l'automobile è un oggetto, la televisione è un oggetto, etc .....**

**Nel campo informatico :**

**una finestra di window (per esempio il Solitario) è un oggetto.**

**In particolare un oggetto può essere definito elencando sia i suoi attributi, sia il modo in cui interagisce con il sistema operativo e le altre finestre, cioè i suoi metodi.**

**Un oggetto ha una struttura, composta dagli attributi e dai metodi.**

**Possono esistere più oggetti che hanno gli stessi attributi, anche con valori diversi, e che dispongono degli stessi metodi. Si dice che questi metodi appartengono alla stessa CLASSE.**

**Una classe specifica gli attributi, senza indicarne il valore, e i metodi; un oggetto non può esistere se prima non viene creata una classe a cui l'oggetto deve appartenere.**

E' possibile realizzare un semplice diagramma degli oggetti per rappresentare appunto **l'oggetto finestra:**

**FINESTRA**

posizione sullo schermo = 100, 50 visibile = si titolo = "Solitario" colore di sfondo = "verde"
sposta (45, 100) chiudi () riduci ()

### ***Gli oggetti predefiniti del Javascript***

In Javascript esistono una serie di oggetti predefiniti a cui ci si può riferire tramite :

**Nome oggetto.Proprietà**

**Nome oggetto.Metodo**

## Proprietà comuni a tutti gli OGGETTI

### Proprietà LENGTH

La proprietà Length fornisce la lunghezza delle variabili stringa. Esempio :

**Var Str="Sergio"**

Uso : `document.write("la lunghezza del mio nome è "+Str.length+" caratteri.")`

**Stampa la lunghezza della variabile Str**

### Proprietà NAME

La proprietà name identifica il nome dell' oggetto. Esempio :

**<FORM NAME="F1">**

**Inserisci il tuo nome <INPUT TYPE="text" NAME="username">**

**</FORM>**

Uso : `document.write(F1.username.value)`

Stampa il valore della username

Vediamo un po' **come riferirsi** agli elementi HTML con gli oggetti del Javascript.

## RIFERIMENTI ASSOLUTI

La tecnica è quella del **riferimento assoluto** si ottiene attraverso la **dot notation** (.)

Si parte dall' **oggetto** piu' alto nella gerarchia (in questo caso F1) separandolo con un punto.

(In coda al presente documento TABELLA riassuntiva degli oggetti del JAVASCRIPT)

Se ad esempio si vuole far riferimento al **titolo del documento** (TITLE) in cui ci troviamo, dobbiamo usare :

**windows.document.title="Posso modificare il titolo"**

Quindi, il TITLE contenuto nell' HTML è stato modificato da un oggetto del javascript.

Quando, invece si vuole interagire con gli elementi di una **FORM HTML** e ci si vuole riferire ad uno degli elementi contenuti, si deve procedere come segue :

**window.document.forms[indice form].elements[indice elemento nella form].Nome proprietà**

- **forms[]** indica un array di form, il primo del quale è form[0] e corrisponde al primo form che trova scorrendo il documento HTML dall' alto verso il basso, il secondo form[1], etc...

- **elements[]** indica un array di elementi contenuti in un form. Il primo elemento è element[0], il secondo è elements[1], e così via ...
- **Nome proprietà** indica la proprietà che vogliamo accedere e può essere : value,name,length, etc..

**Dove :**

**value** restituisce il valore contenuto nell' elemento

**name** restituisce la stringa che contiene il nome dell' elemento

**length** restituisce la lunghezza dell' elemento

**Esempio : window.document.forms[1].elements[2].value="Ciao"**

Proviamo in pratica (**ESEMPIO 4.1**):

```
<HTML>
  <HEAD>
    <SCRIPT Language = "JavaScript">
      function Quadrato( )
      {
        document.forms[0].elements[2].value =
        (parseInt(document.forms[0].elements[1].value) *
         parseInt(document.forms[0].elements[1].value))
      }
    </SCRIPT>
  </HEAD>
  <BODY>
    <FORM >
    <INPUT TYPE="button" NAME="quadrato" VALUE="Eleva al quadrato" onClick="Quadrato( )">
    <<!-- Elemento 0 -->
    <INPUT TYPE="text" VALUE="" > <!--Elemento 1 -->
    <INPUT TYPE="text" VALUE="" > <!-- Elemento 2 -->
    </FORM>
  </BODY>
</HTML>
```

Analogamente a quanto visto per le FORM nell' HTML, dal punto di vista del javascript, possiamo far riferimento e trattare altri oggetti come **array di link**, **array di ancore**, **array di immagini** ad esempio :

**windows.document.images[Indice array di immagini]**

images[] indica un array delle immagini presenti nel nostro documento, la prima immagine corrisponde a images[0] a partire dall' inizio del documento html e via via le successive 1,2, etc...

lo stesso vale per gli array di **ancors[]** e **links[]**

## RIFERIMENTI RELATIVI

Quando si assegna con la proprietà Name, un nome è possibile riferirsi a tutti gli oggetti appartenenti (**detti per tale motivo riflessi**) utilizzando la **dot notation** rispettando la gerarchia degli oggetti:

**Nome Form.Nome Elemento.Nome proprietà**

**Nome Form** è il nome della form assegnato tramite la proprietà name

**Nome Elemento** è il nome assegnato, tramite la proprietà name, all' elemento nella form

**Nome proprietà** come al solito può essere name,value,length,etc ..

Esempio : **Form1.Elemento1.value**

Facciamo una prova, riprendendo l' esempio precedente, usando i **referimenti relativi** :

```
<HTML>
  <HEAD>
    <SCRIPT Language="JavaScript">
      function Quadrato( )
      {
        F1.risultato.value = (parseInt(F1.inserisci.value) *
        parseInt(F1.inserisci.value))
      }
    </SCRIPT>
  </HEAD>
  <BODY>
    <FORM NAME="F1">
      <INPUT TYPE="button" NAME="quadrato" VALUE="Eleva al quadrato"
      onClick="Quadrato( )">
      <INPUT TYPE="text" NAME="inserisci" VALUE="" >
      <INPUT TYPE="text" NAME="risultato" VALUE="" >
    </FORM>
  </BODY>
</HTML>
```

<b>Oggetto : BUTTON</b>
-------------------------

L' oggetto **Button** contiene i pulsanti speciali **SUBMIT,RESET** di HTML è caratterizzato da :

**Proprietà : name, type, e value**

**Metodi : click()** che simula il click del mouse su un oggetto

**Eventi : onClick** quando si clicca sul pulsante

Vediamo un esempio :

```
<html>
<head>
  <script language="javascript">
    function cambia( )
    {
      x= prompt("inserisci il nuovo nome del pulsante", "")
      f1.p1.value=x
      f1.p2.click( )
    }
  </script>
</head>
```

```

        function visualizza( )
        {
            alert("nome pulsante : " + f1.p1.name +" valore pulsante: " + f1.p1.value)
        }
    </script>
</head>
<body>
<form name="f1">
<input type="button" name="p1" value="vecchionomepulsante" onclick="cambia( )">
<input type="button" name="p2" value="visualizza" onclick="visualizza( )">
</form>
</body>
</html>

```

### Oggetto campo : TEXT

Con questo oggetto l' utente realizza un input

**Proprietà : name,type,value,size**

**Metodi : focus(),blur(), select()**

**Eventi : Focus,Blur,Select,Change** che sono gestiti da **onFocus,onBlur,onSelect,onChange**

Esempio :

```

<html>
    <head>
        <script language="javascript">
            function visualizzaa( )
            { f1.v.value= f1.a.value.length}
            function visualizzab( )
            { f1.v.value= f1.b.value.length}
            function visualizzac( )
            { f1.v.value= f1.c.value.length}
            function visualizzad( )
            { f1.v.value= f1.d.value.length}
        </script>
    </head>
    <body>
    <form name="f1">
    <input type="text" name="a" value="testoa" onfocus="visualizzaa( )">
    <input type="text" name="b" value="testob" onblur="visualizzab( )">
    <input type="text" name="c" value="testoc" onselect="visualizzac( )">
    <input type="text" name="d" value="testod" onchange="visualizzad( )">
    <input type="text" name="v" >
    </form>
    </body>
</html>

```

**Vediamo un altro esempio :**

```

<html>
<head>
<script type="text/javascript">
function upperCase()
{

```

```

var x=document.getElementById("fname").value;
document.getElementById("fname").value=x.toUpperCase();
}
</script>
</head><body>Enter your name:
<input type="text" id="fname" onblur="toUpperCase()"></body>
</html>

```

## L' Oggetto caselle di controllo : CHECKBOX

L' oggetto checkbox, permette di selezionare le opzioni necessarie secondo un modello on/off.

**Proprietà :** **checked** valore booleano vero o falso a seconda che il pulsante opzione sia attivato o no,  
**name, type, value**

**Metodi :** **click()**

**Eventi :** **onClick**

**Esempio :**

```

<html>
  <head>
    <script language="javascript">
      function controlla( )
      {
        if (f1.s.checked && f1.n.checked)
          f1.a.checked = true
      }
    </script>
  </head>
  <body>
    <form name="f1">
      stato civile (sposato/non sposato) <input type="checkbox" name="s" checked><br>
      nazionalità (italiana/straniera)   <input type="checkbox" name="n">
      agevolazioni sanitarie             <input type="checkbox" name="a">
      <input type="button" name="controlla" value="controlla" onclick="controlla( )">
    </form>
  </body>
</html>

```

## L' oggetto pulsante di opzione : RADIO

Un oggetto radio box , rappresenta un insieme di elementi **raggruppati da uno stesso nome**, che possono essere selezionati.

**Proprietà :** **checked** valore booleano vero o falso a seconda che il pulsante opzione sia attivato o no,  
**name, type, value, length** (rappresenta il numero di pulsanti di opzione presenti )

**Metodi :** **click()**

**Eventi :** **onClick**

**Esempio :**

```

<HTML>
  <HEAD>

```

```

<SCRIPT Language="JavaScript">
    function Visualizza( )
    {
        if (F1.R[0].checked) S="Maschile"
        else
            S="Femminile"
        alert("età: " + F1.eta.value + " Sesso: " + S)
    }
    function Modifica ( )
    {
        F1.R[0].checked = true
    }
</SCRIPT>
</HEAD>
<BODY>
    <FORM NAME="F1">
        maschile <INPUT TYPE="radio" NAME="R" CHECKED><br>
        femminile <INPUT TYPE="radio" NAME="R" >
        eta: <INPUT TYPE="text" NAME="eta">
    <INPUT TYPE="button" NAME="visualizza" VALUE="visualizza dati" onClick="Visualizza( )">
    <INPUT TYPE="button" NAME="modifica" VALUE="imposta maschile" onClick="Modifica( )">
    </FORM>
</BODY>
</HTML>

```

## L' oggetto : SELECT

**Insieme di valori selezionabili, presenti all' interno di un menu' a discesa.**

Se un oggetto SELECT contiene due opzioni si accederà a queste con l' array OPTIONS indicando i due elementi con :

**sel.option[0]**  
**sel.option[1]**

**Proprietà**

**name, value:** riflesse dai parametri NAME e VALUE dell'HTML

**type :** select-one che assegnano un solo valore

**length :** contiene il numero di opzioni

**selectedIndex :** contiene l' indice dell' opzione selezionata

**Metodi :** focus(),blur()

**Eventi :** onChange,onFocus,onBlur

**(ESEMPIO 4.2):**

```

<HTML>
    <HEAD>
        <SCRIPT Language="JavaScript">
            function Visualizza( )
            {
                if ( (indice = F1.pianeti.selectedIndex) != 0)    // è stato selezionato un pianeta
                {
                    F1.stato.options[0].selected=true
                    alert("pianeta: " + F1.pianeti.options[indice].text + "\n Indice: " + indice)
                }
            }
        </SCRIPT>
    </HEAD>
    <BODY>
        <FORM NAME="F1">
            <SELECT NAME="pianeti">
                <OPTION VALUE="1">Marte
                <OPTION VALUE="2">Venere
                <OPTION VALUE="3">Giove
                <OPTION VALUE="4">Saturno
            </SELECT>
            <INPUT TYPE="button" NAME="visualizza" VALUE="visualizza dati" onClick="Visualizza( )">
        </FORM>
    </BODY>
</HTML>

```



```

        }
        else
            F1.stato.options[1].selected=true
    }
</SCRIPT>
</HEAD>
<BODY>
    <FORM NAME="F1">
        <SELECT NAME="pianeti" onChange="Visualizza( )">
            <OPTION SELECTED> nessun pianeta
            <OPTION> Marte
            <OPTION> Venere
            <OPTION> Giove
        </SELECT>
        <SELECT NAME="stato">
            <OPTION> Pianeta selezionato
            <OPTION SELECTED> Pianeta da selezionare
        </SELECT>
    </FORM>
</BODY>
</HTML>

```

## Oggetto window

Esempio : `window.onload = "alert('pagina in caricamento...');";`

### Come si usa

Attenzione a non incappare in questo tipo di errori:

```

function saluta () {
    alert("pagina in caricamento");
};
window.onload = saluta();

```

In questo codice l'intenzione è quella di mostrare un avviso al caricamento della pagina;

**Un altro metodo** consiste nell'assegnare alla funzione un **oggetto function** creato sul momento:

```

window.onload = function () { alert("pagina in caricamento"); };

```

***TABELLA Javascript: Proprietà, metodi ed eventi degli oggetti***

Oggetti in collegamento al Browser			
Oggetti	Proprietà	Metodi	Eventi
Window	defaultStatus frames opener parent scroll self status top window	alert() blur() close() confirm() focus() open() prompt() clearTimeout() setTimeout ()	onLoad onUnload onBlur onFocus
Frame	defaultStatus frames opener parent scroll self status top window	alert() blur() close() confirm() focus() open() prompt() clearTimeout() setTimeout()	Nessuno (gli eventi onLoad and onUnload appartengono all'oggetto window)
Location	hash host hostname href pathname por protocol search	reload() replace()	Nessuno
History	length forward go	back()	Nessuno
Navigator	appName appVersion mimeTypes plugins userAgent	JavaEnabled()	Nessuno
document	alinkColor anchors applets area bgColor	Clear() close() open() write() writeln()	Nessuno (gli eventi onLoad and onUnload appartengono all'oggetto window).

	cookie fgColor forms images lastModified linkColor links location referrer title vlinkColor		
image	border complete height hspace lowsrc name src vspace width	Nessuno	Nessuno
form	action elements encoding FileUpload method name target	submit() reset()	onSubmit onReset
text	defaultValue name type value	focus() blur() select()	onBlur onCharge onFocus onSelect

Oggetti Javascript			
Oggetti	Proprietà	Metodi	Eventi
Array	length	Join reverse sort xx	Nessuno
Math	E LN10 LN2 LOG10E LOG2E PI SQRT1_2	<ul style="list-style-type: none"> <li>▪ abs()</li> <li>▪ acos()</li> <li>▪ asin()</li> <li>▪ atan()</li> <li>▪ atan2()</li> <li>▪ ceil()</li> <li>▪ cos()</li> </ul>	nessuno

	SQRT2	<ul style="list-style-type: none"> <li>▪ exp()</li> <li>▪ floor()</li> <li>▪ log()</li> <li>▪ max()</li> <li>▪ min()</li> <li>▪ pow()</li> <li>▪ random()</li> <li>▪ round()</li> <li>▪ sin()</li> <li>▪ sqrt()</li> <li>▪ tan()</li> </ul>	
String	length	toLowerCase() toUpperCase() substring() indexOf() lastIndexOf() bold() charAt()	nessuno
Date	nessuna	getDate() getHours() getMinutes() getMonths() getYear() setDate() setHours() setMinutes() setMonths() setYear()	Nessuno

Oggetti HTML-JAVASCRIPT			
Oggetti	Proprietà	Metodi	Eventi
	▪		
Text	defaultValue form name type value	blur () focus() select ()	onFocus onBlur onSelect onChange
Textarea	defaultValue form	blur() focus()	onFocus onBlur

	name type value	select()	onSelect onChange
Password	defaultValue form name type value	blur() focus() select ()	Nessuno
Button	form name type value	blur() click() focus()	onClick
Submit	form name type value	blur() click() focus()	onClick
Reset	form name type value	blur() click() focus()	onClick
Radio	checked defaultChecked form name type value	blur () click() focus()	onClick
Checkbox	checked defaultChecked form name type value	blur() click() focus()	onClick
Select	form length name options selectedIndex type	blur() focus()	onBlur onChange onFocus
Option	defaultSelected selected text value	blur() focus()	onBlur onChange onFocus

