Javascript

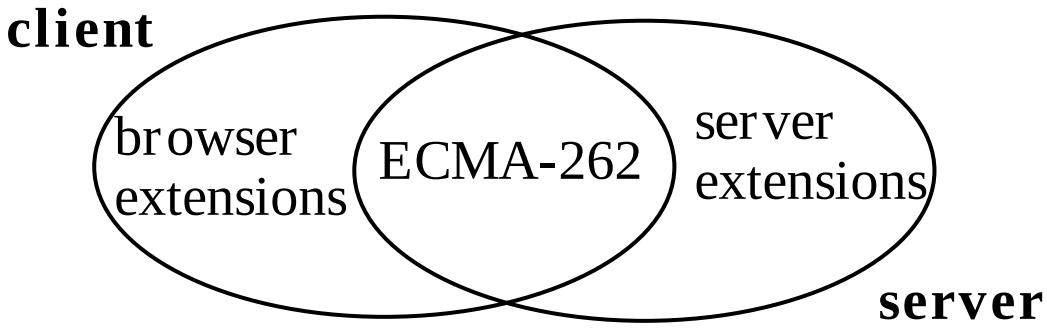
Introduzione

Javascript è uno scripting language, non un linguaggio di markup, non un linguaggio di programmazione. Più precisamente potremmo dirlo un linguaggio *application-embedded*.

Venne introdotto da Netscape nel 1995 come LiveScript, poi a seguito del successo di Java cambiato in Javascript, poi adottato (controvoglia) anche da Microsoft per Internet Explorer ma ribattezzato Jscript, adesso è sottoposto a processo di standardizzazione ad opera della società di standard ECMA, con il nome di ECMAscript o ECMA-262.

Javascript è:

- **cross-platform** (non si basa su nessuno specifico S.O. o architettura)
- **object-based** (un programma JS manipola oggetti predefiniti dall'applicazione)
- **interpretato** (il codice viene letto ed eseguito così come è stato scritto, senza passaggi intermedi)
- **embedded** (il programma è inserito e convive con la pagina HTML)
- sia **client-side** che **server-side** (ma con alcune differenze):



Cosa serve Javascript

- a creare di volta in volta il contenuto delle pagine HTML sulla base di situazioni o basi di dati esterne.
- ad attivare azioni client-side sugli oggetti del documento HTML (controlli sui form, determinazione della destinazione di un link, ecc.)
- a realizzare semplici applicazioni client-side

Il codice Javascript viene inserito nella pagina HTML. Esso viene eseguito via via che la pagina viene esaminata, oppure quando avvengono gli eventi a cui gli script sono assegnati.

Il codice client-side è contenuto in blocchi <SCRIPT> </SCRIPT>, mentre il codice server-side è contenuto in blocchi <SERVER> </SERVER>, e viene sostituito con il risultante HTML prima di essere spedito al browser.

Differenze e similitudini con Java

Sintassi simile al C	Sintassi simile al C
Interpretato	compilato in bytecode
Integrato con la pagina HTML	immerso nell'HTML ma isolato
Object-based	object-oriented
Loose typing	strong typing
Dynamic binding	static binding
Scripting language	programming language
Abbastanza sicuro	Molto sicuro

Usare Javascript in HTML

- Usando il tag SCRIPT nel corpo della pagina HTML
- Come valore di attributi di tag HTML
- Come gestione di eventi predeterminati

Dove mettere il codice Javascript

- Tra <SCRIPT e </SCRIPT> nell'header
- Tra <SCRIPT> e </SCRIPT> nel body
- In linea negli attributi dei tag e nei gestori di eventi
- Ponendo il codice Javascript in un file esterno
- Usare il tag NOSCRIPT

Esempio 1


Netscape: Esempio Javascript
Back
Forward
Home
Edit
Reload
Images
Print
Find
Stop
Location :
file:///Thick%20as%20a%20brick/Roba%20d'uso/Lucidi%20lezioni/Comunicazione/
Esempio Javascript
<HEAD>
<TITLE>Esempio Javascript</TITLE>
</HEAD>
<BODY>
<P>Starting... </P>
<SCRIPT LANGUAGE="JavaScript">
<!-- Hide script from old browsers
document.write("<P>Hello <B>world</B></P>");
// End script hiding from old browsers -->
</SCRIPT>
<P> All done. </P>
</BODY>
Starting...
Hello world
All done.

Le funzioni e la struttura dei programmi Javascript

Meccanismo di strutturazione ed organizzazione del programma

Meccanismo di astrazione e generalizzazione di procedure

Funzioni che ritornano valori, e funzioni che non ritornano valori

La sintassi delle funzioni

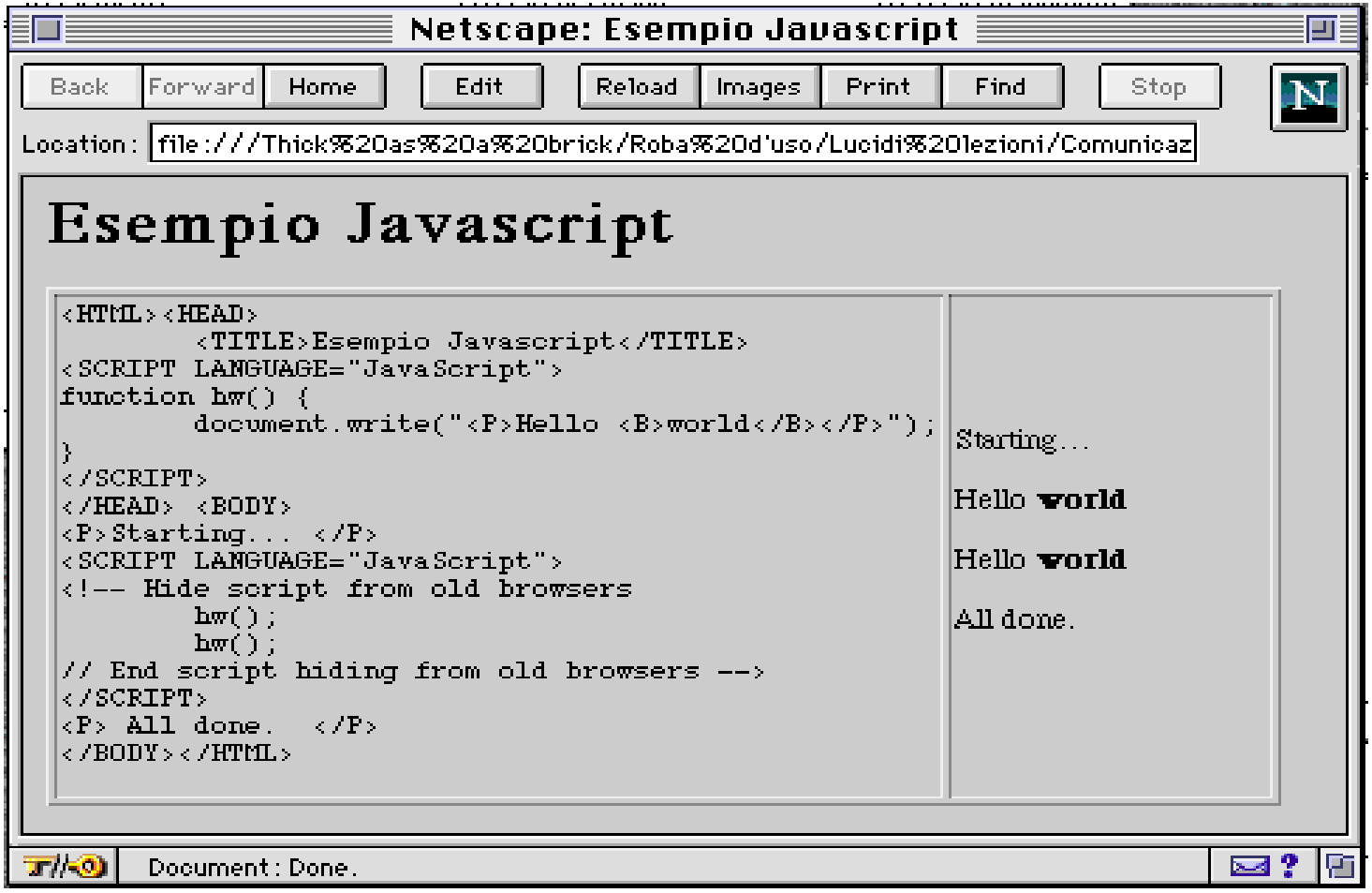
Definizione
(senza valore di ritorno)
	`function nome(pformali) {`
	`  corpo funzione ;`
	`}`

Uso (senza valore di ritorno)	`nome(pattuali) ;`

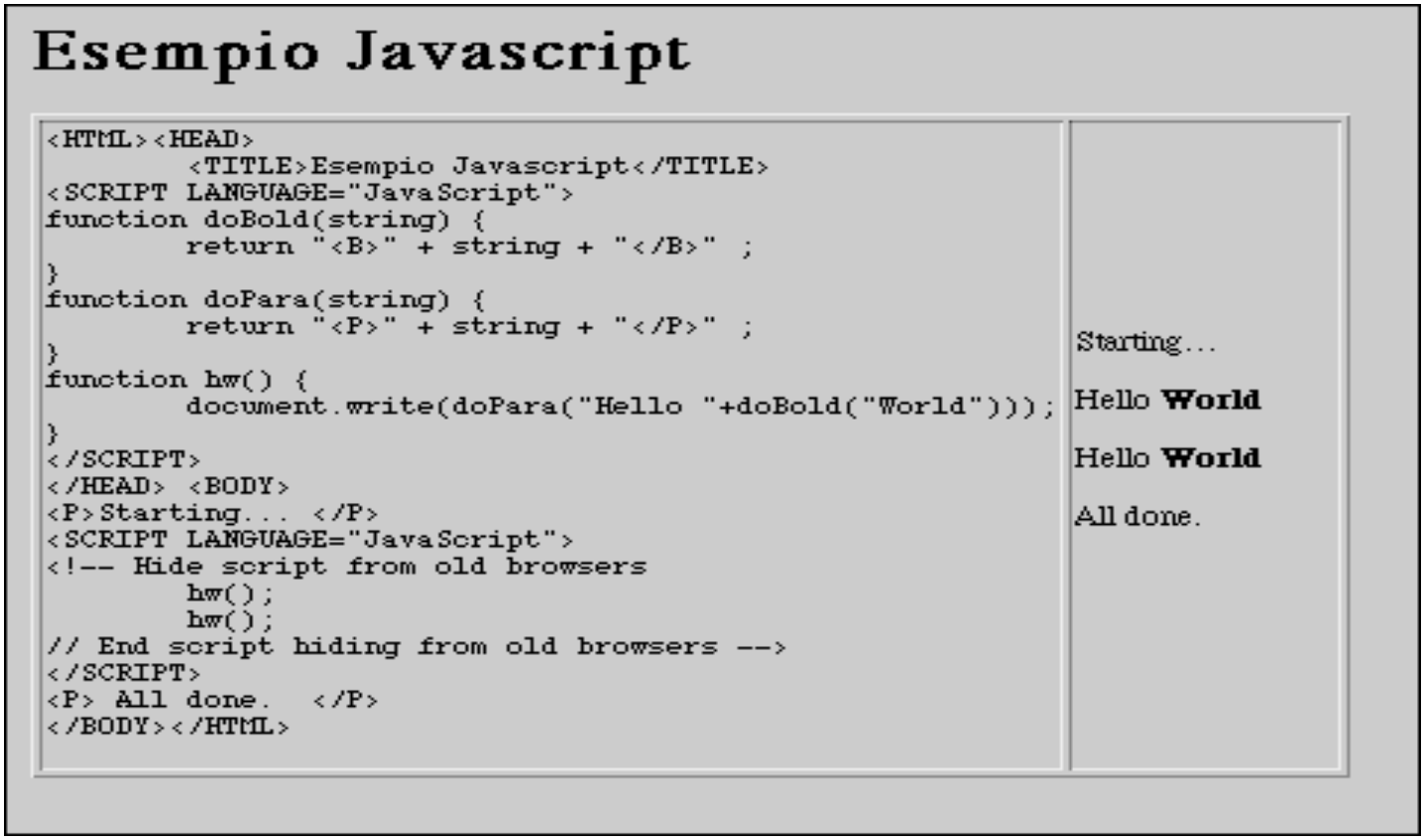
Definizione
(con valore di ritorno)
	`function nome(pformali) {`
	`  corpo funzione ;`
	`  return valore ;`
	`}`

Uso (con valore di ritorno)	`a = b + nome(pattuali) ;` *(o altra espressione alfanumerica)*

Esempio 2


Netscape: Esempio Javascript
Back
Forward
Home
Edit
Reload
Images
Print
Find
Stop
Location: file:///Thick%20as%20a%20brick/Roba%20d'uso/Lucidi%20lezioni/Comunicaz
Esempio Javascript
<HTML><HEAD>
<TITLE>Esempio Javascript</TITLE>
<SCRIPT LANGUAGE="JavaScript">
function hw() {
document.write("<P>Hello <B>world</B></P>");
}
</SCRIPT>
</HEAD> <BODY>
<P>Starting... </P>
<SCRIPT LANGUAGE="JavaScript">
<!-- Hide script from old browsers
hw();
hw();
// End script hiding from old browsers -->
</SCRIPT>
<P> All done. </P>
</BODY></HTML>
Starting...
Hello world
Hello world
All done.
Document: Done.

Esempio 3


Esempio Javascript
<HTML><HEAD>
<TITLE>Esempio Javascript</TITLE>
<SCRIPT LANGUAGE="JavaScript">
function doBold(string) {
return "<B>" + string + "</B>" ;
}
function doPara(string) {
return "<P>" + string + "</P>" ;
}
function hw() {
document.write(doPara("Hello "+doBold("World")));
}
</SCRIPT>
</HEAD> <BODY>
<P>Starting... </P>
<SCRIPT LANGUAGE="JavaScript">
<!-- Hide script from old browsers
hw();
hw();
// End script hiding from old browsers -->
</SCRIPT>
<P> All done. </P>
</BODY></HTML>
Starting...
Hello World
Hello World
All done.

Esempio 4

```
<HTML><HEAD>
        <TITLE>Esempio Javascript</TITLE>
<SCRIPT LANGUAGE="JavaScript">
function doTag(tag, text) {
        return "<"+tag+">"+text+"</"+tag+">" ;
}
function doBold(string) {
        return doTag("B",string) ;
}
function doPara(string) {
        return doTag("P", string) ;
}
function hw() {
        document.write(doPara("Hello "+doBold("World")));
}
</SCRIPT>
</HEAD> <BODY>
<P>Starting... </P>
<SCRIPT LANGUAGE="JavaScript">
<!-- Hide script from old browsers
        hw();
        hw();
// End script hiding from old browsers -->
</SCRIPT>
<P> All done.  </P>
</BODY></HTML>
```

Starting...

Hello **World**

Hello **World**

All done.

Costrutti sintattici (1ª parte)

Condizionali	`if (espressione logica) {`
	`  blocco per condizione vera ;`
	`} else {`
	`  blocco per condizione falsa ;`
	`}`

Ciclo	`while (espressione logica) {`
	`  istruzioni ;`
	`}`
	`istruzioni ;`

Assegnazione	`contenitore = espressione ;`
	`contenitore += espressione ;`
	`contenitore -= espressione ;`
	`contenitore++ ;`
	`++contenitore ;`
	`contenitore-- ;`
	`--contenitore ;`

Contenitore	Entità del programma che contiene stabilmente un valore. Sono contenitori le variabili, gli elementi degli array, le proprietà degli oggetti predefiniti, ecc. I contenitori NON sono tipati in Javascript.
Costanti (o letterali)	Un valore numerico, logico o stringa prefissato e non modificabile. Javascript riconosce quattro tipi di costanti: • Numeri a = 42 • Stringhe b = "Ciao mamma" • Booleani c = true • null d = null

Espressione	Qualunque sequenza di operazioni tra costanti e contenitori, utilizzando operatori, funzioni pre-definite, funzioni dell'utente ed altre espressioni, purchè compatibile come tipi, che determini un unico valore finale.
Operatori	Esistono in Javascript operatori aritmetici, logici, di stringhe, di confronto, bitwise e speciali. *(Discorso complesso: anche gli assegnamenti sono operatori.)*

Operatori

Operatori aritmetici

Somma	`a + b`	Sottrazione	`a - b`
Moltiplicazione	`a * b`	Divisione	`a / b`
Modulo (resto)	`a % b`	Meno unario	`- a`

Operatori logici

And	`a && b`	a && b è vero se sono veri **sia** a **sia** b
Or	`a || b`	a || b è vero se è vero **o** a, **o** b, **o** entrambi
Not	`! a`	! a è vero se a è falso

Operatori di stringhe

Concatenazione	`a + b`	Se a vale "ciao" e b vale " mamma", a + b vale "ciao mamma"

Operatori di confronto

Uguale	`a == b`	Diverso	`a != b`
Minore	`a < b`	Minore o uguale	`a <= b`
Maggiore	`a > b`	Maggiore o uguale	`a >= b`

Operatori speciali

Op. condizionale	`a ? b : c`	Se a è vero, vale b, altrimenti c
Op. virgola	`a , b`	Valuta entrambi gli operandi, e ritorna il valore del secondo: ad es. 5 , 7 vale 7.

Esempio 5

```
<HTML><HEAD>
        <TITLE>Esempio Javascript</TITLE>
<SCRIPT LANGUAGE="JavaScript">
function doTag(tag, text) {
        return "<"+tag+">"+text+"</"+tag+">" ;
}
function doItalic(string) { return doTag("I",string) ; }
function doBold(string)   { return doTag("B",string) ; }
function doPara(string)   { return doTag("P",string) ; }

function multiplehw() {
        a = doPara("Hello " + doBold("World")) ;
        b = doPara("Hello " + doItalic("People")) ;
        i = 0 ;
        while (i < 5) {
                if (i%2==0) {
                        document.write(a);
                } else {
                        document.write(b);
                }
                i++ ;
        }
}
</SCRIPT>
</HEAD> <BODY>
<SCRIPT LANGUAGE="JavaScript">
<!-- Hide script from old browsers
        multiplehw();
// End script hiding from old browsers -->
</SCRIPT>
</BODY></HTML>
```

Hello **World**

Hello *People*

Hello **World**

Hello *People*

Hello **World**

Oggetti ed eventi in Javascript

Introduzione

Javascript ha un certo numero di oggetti predefiniti, che possono essere manipolati da programma. Gli oggetti possono contenere (avere definiti) altri “sottooggetti” analogamente manipolabili.

Ad esempio Window è l’oggetto principale di Javascript, e contiene un certo numero di sotto-oggetti, come Location, Document, History, Frame, detti “membri”.

Gli oggetti e valori contenuti in un oggetto si chiamano “proprietà”. Questo e’ analogo ai campi di un record. Oltre alle proprietà di un oggetto ci sono i suoi “metodi”, che sono funzioni e procedure “interne” e specifiche dell’oggetto stesso. Quindi un oggetto è una collezione di proprietà e metodi.

A volte le proprietà di un oggetto non sono oggetti semplici, ma collezioni numerate, dette “array”. Ad esempio, ogni finestra è composta da zero o più frame, che sono numerati ed ordinati secondo la loro definizione.

```
window.frame[0]
```

è il primo frame definito nella finestra corrente

```
window.frame[1]
```

è il secondo frame definito, etc.

```
window.frame[1].document.title
```

è il titolo del documento contenuto nel secondo frame della finestra

La notazione a punti serve per identificare un elemento (proprietà o metodo) di un oggetto. La notazione a punti va esaminata dal generico allo specifico, da sinistra a destra. Le proprietà sono individuati dalla presenza di un nome semplice, i metodi dalla presenza di un nome seguito dalle parentesi.

```
window.document.form[0].text[1].value
```

è il valore del secondo elemento di input del primo form del documento contenuto nella finestra. Poichè il valore è un'oggetto stringa, posso applicare un metodo applicabile alle stringhe:

```
window.document.form[0].text[1].value.toUpperCase()
```

una funzione che restituisce la stringa in maiuscolo.

L’identificativo “this” serve per specificare la finestra o il documento a cui ci stiamo riferendo in maniera breve e generica.

Nel contesto di window.document,

```
this.form[0].text[1].value.toUpperCase()
```

restituisce in maiuscolo il valore del secondo elemento di testo del primo form del documento della finestra.

In tutti i casi in cui l’oggetto ha un nome univoco, è possibile sostituire la sua posizione nell’array con il suo nome. Per esempio, se il secondo elemento di testo si chiama “Pippo” e il form si chiama “Pluto”,

```
window.document.form[0].text[1].value.toUpperCase()
```

è equivalente a

```
window.document.Pluto.Pippo.value.toUpperCase()
```

Eventi

E' possibile associare degli script Javascript agli eventi che accadono ad una finestra. Così è possibile specificare comportamenti da eseguire quando l'utente esegue azioni sugli oggetti del documento, o quando avvengono per altre cause delle situazioni importanti (ad es. si carica il documento). Gli script che vengono eseguiti in seguito ad un evento si chiamano **event handler**.

Gli eventi vengono associati al tag HTML dell'oggetto interessato: i campi di un form, il body, i link. La forma di un event handler è la seguente:

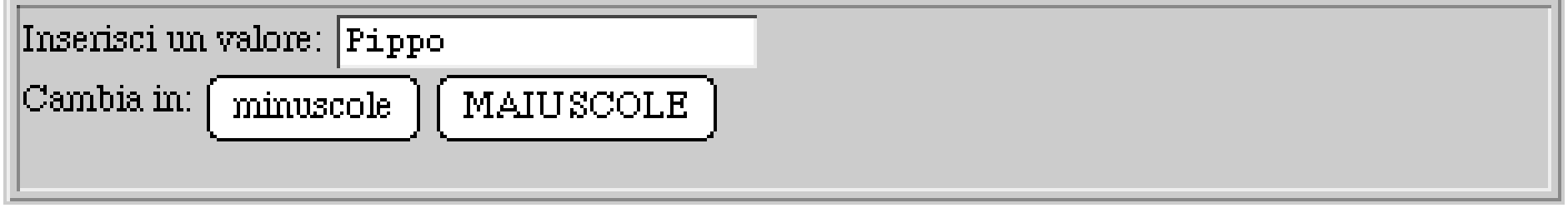
```
<TAG attributi onEvento="comando javascript; …">
```

Ad esempio

```
<INPUT type=button value="Calcola" onClick="compute()">
```

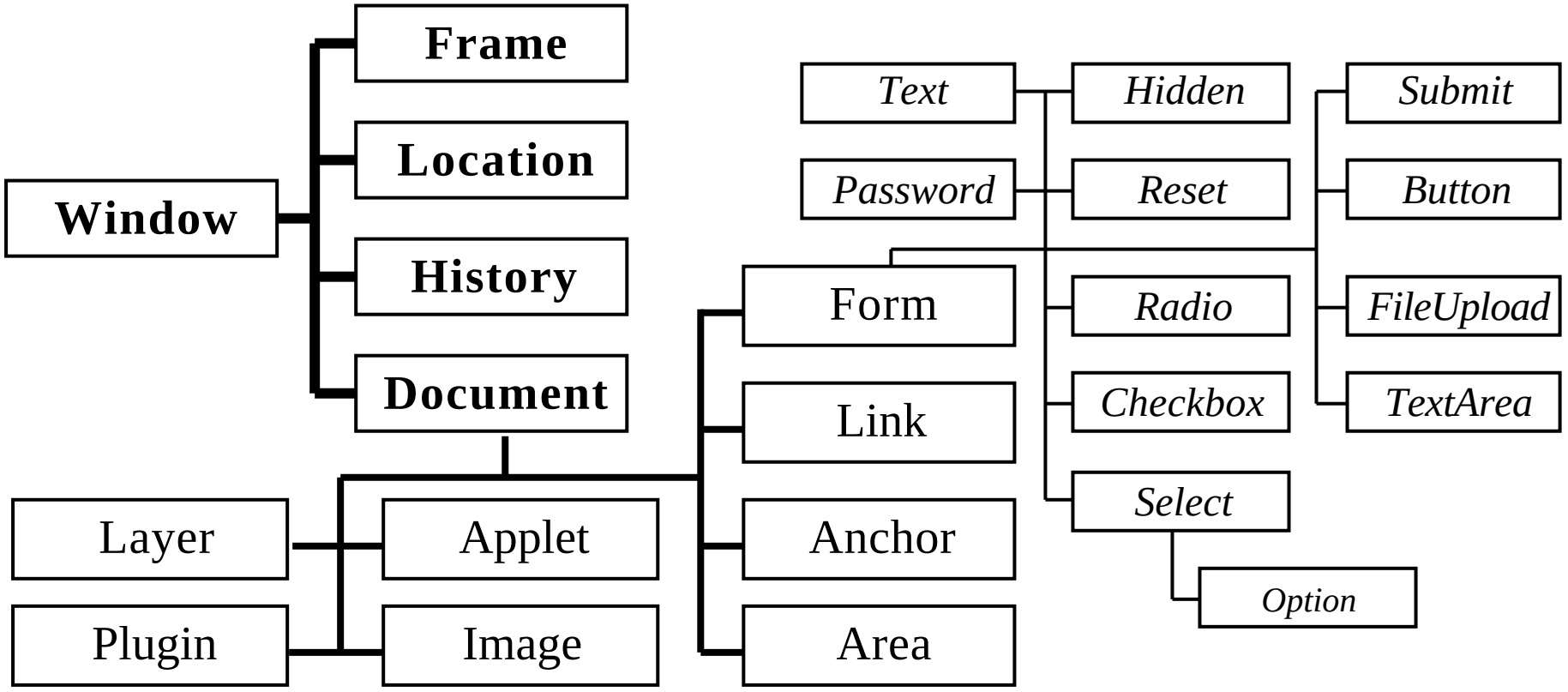
Esempio 6

```
<SCRIPT>
function cambia(form, tipo) {
  if (tipo=='a') {
    form.text1.value = form.text1.value.toLowerCase()
  } else {
    form.text1.value = form.text1.value.toUpperCase()
  }
}
</SCRIPT>
<FORM NAME="myform">
Inserisci un valore:
  <INPUT TYPE="text" NAME="text1" VALUE="Pippo" SIZE=20>
  <BR>Cambia in:
    <INPUT TYPE="button" Value="minuscole" onClick="cambia(this.form,'a')">
    <INPUT TYPE="button" Value="MAIUSCOLE" onClick="cambia(this.form,'A')">
</FORM>
```



Gerarchia degli oggetti (semplificata)

Ci sono due oggetti principali in ogni sessione di Netscape Navigator: la (o le) finestre aperte, e Navigator stesso. Ogni finestra ha una serie di proprietà associate, e formano una gerarchia:



Proprietà di window (lista parziale)

frames	Un array di tutte le frame di una finestra
history	Un oggetto di informazioni sugli URI visitati all'interno della finestra.
location	Un oggetto di informazioni sull'URL corrente.
document	Un oggetto di informazioni sul documento visualizzato.
self	Un sinonimo per la finestra attuale.
name	Il nome unico di questa finestra
length	Il numero di frame definito nella finestra.
status	Il messaggio visualizzato nella barra di stato.
defaultStatus	Il messaggio di default message visualizzato nella bara di stato

Altre proprietà

parent, top, locationbar, menubar, personalbar, scrollbars, statusbar, toolbar, closed, opener, innerHeight, innerWidth, outerHeight, outerWidth, pageXOffset, pageYOffset.

Metodi di window (lista parziale)

alert	Mostra una finestra di dialogo di tipo Alert
confirm	Mostra una finestra di dialogo di tipo Confirm
prompt	Mostra una finestra con un campo di inserimento
back	Equivalente a premere il pulsante back
forward	Equivalente a premere il pulsante forward.
home	Equivalente a premere il pulsante home
close	Chiude la finestra.
open	Apre una nuova finestra.
print	Stampa il contenuto della finestra o del frame.
find	Trova la stringa specificata nel contenuto della finestra.

Eventi di window (lista completa)

onBlur	Avviene quando il frame o la finestra perde il “fuoco”, ovvero non è più la finestra o il frame principale
onDragDrop	Avviene quando l'utente trascina qualcosa con il mouse
onError	Avviene quando esiste un errore Javascript
onFocus	Avviene quando il frame o la finestra ricevono il fuoco, cioé diventano la finestra o il frame principale
onLoad	Avviene quando il documento specificato viene caricato nella finestra
onMove	Avviene quando l'utente muove la finestra
onResize	Avviene quando l'utente ridimensiona la finestra
onUnload	Avviene quando il documento specificato viene rimosso dalla finestra.

Proprietà di document (lista parziale)

title	Il contenuto del tag TITLE
URL	L'URL del documento
anchors	Un array degli anchor definiti
applets	Un array delle applet definite
forms	Un array delle form definite
images	Un array delle immagini definite
layers	Un array dei layer definiti
links	Un array dei link definiti
plugins	Un array dei plugin definiti
alinkColor	Il contenuto dell'attributo ALINK
bgColor	Il contenuto dell'attributo BGCOLOR.
fgColor	Il contenuto dell'attributo TEXT.
linkColor	Il contenuto dell'attributo LINK.
vlinkColor	Il contenuto dell'attributo VLINK.

Metodi di document (lista parziale)

close	Chiude una sequenza di istruzioni di scrittura e forza la visualizzazione del documento.
open	Apre una sequenza di istruzioni di scrittura
write	Scrive una o più espressioni HTML
writeln	Scrive una o più espressioni HTML seguite da un ritorno a capo
getSelection	Restituisce una stringa del testo attualmente selezionato

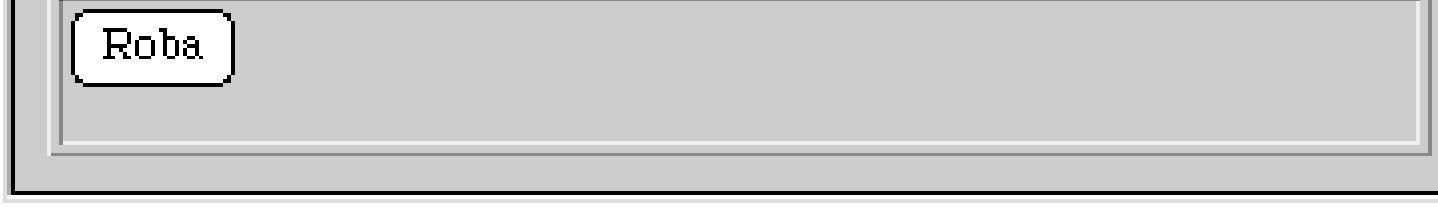
Eventi di document (lista completa)

onClick	Avviene in presenza di un click semplice
onDblClick	Avviene in presenza di un click doppio
onKeyDown	Avviene quando un tasto viene mantenuto premuto
onKeyPress	Avviene quando un tasto viene premuto
onKeyUp	Avviene quando un tasto viene rilasciato
onMouseDown	Avviene quando il pulsante del mouse viene premuto
onMouseUp	Avviene quando il pulsante del mouse viene rilasciato

Esempio 7

```
<BODY   BGCOLOR="lightgrey"
                onBlur = 'document.bgcolor = "white"'
                onFocus = 'document.bgcolor = "lightgrey"'
>
```

```
<SCRIPT>
function roba() {
        this.status = "Secondo documento"
        this.alert("Benvenuto tra noi")
}
</SCRIPT>
<FORM NAME="myform">
    <INPUT TYPE="button" Value="Roba" onClick="roba()">
</FORM>
```



Form ed elementi del form

Ogni form di un documento é un oggetto distinto. Vi si può riferire tramite il nome (specificato con l'attributo NAME nel tag FORM) o specificando l'ordine di apparizione nell'array `document.forms`. L'array `elements` contiene una voce per ogni elemento dell'array.

Proprietà di un form (elenco parziale)

action	Il contenuto dell'attributo ACTION
elements	L'array di tutti gli elementi del form.
encoding	Il contenuto dell'attributo ENCTYPE
length	Il numero di elementi del form
method	Il contenuto dell'attributo METHOD
name	Il contenuto dell'attributo NAME

Metodi di un form (elenco parziale)

reset	Equivalente a premere sul pulsante di reset
submit	Equivalente a premere un pulsante submit (viene attivata l'azione specificata)

Eventi di un form (elenco completo)

onReset	Avviene quando l'utente preme sul pulsante Reset o si attiva il metodo reset da programma
onSubmit	Avviene quando l'utente preme sul pulsante Submit o si attiva il metodo submit da programma

L'oggetto TEXT

Ogni oggetto input di tipo TEXT di un form è un oggetto di tipo text. Si accede a questo oggetto o specificandone il nome o la posizione nell'array elements del form.

Proprietà di un TEXT (elenco completo)

defaultValue	Il contenuto dell'attributo VALUE
form	Il nome del form che contiene il campo
name	Il contenuto dell'attributo NAME
type	Il contenuto dell'attributo TYPE
value	Il contenuto attuale del campo

Metodi di un TEXT (elenco parziale)

blur	Toglie il fuoco dall'oggetto
focus	Dà il fuoco all'oggetto
select	Seleziona il valore dell'oggetto

Eventi di un TEXT (elenco completo)

onBlur	Avviene quando l'utente sposta il fuoco ad un altro oggetto
onChange	Avviene quando l'utente cambia il valore del campo
onFocus	Avviene quando l'utente sposta il fuoco a questo oggetto
onSelect	Avviene quando l'utente seleziona tutto o in parte il contenuto del campo

L'oggetto BUTTON

Ogni oggetto input di tipo BUTTON di un form è un oggetto di tipo button. Si accede a questo oggetto o specificandone il nome o la posizione nell'array elements del form.

Proprietà di un BUTTON (elenco completo)

form	Il nome del form che contiene il campo
name	Il contenuto dell'attributo NAME
type	Il contenuto dell'attributo TYPE
value	Il contenuto attuale del campo

Metodi di un BUTTON (elenco parziale)

blur	Toglie il fuoco dall'oggetto
focus	Dà il fuoco all'oggetto
click	Fa click sul bottone

Eventi di un BUTTON (elenco completo)

onBlur	Avviene quando l'utente sposta il fuoco ad un altro oggetto
onClick	Avviene quando l'utente fa click sul pulsante
onFocus	Avviene quando l'utente sposta il fuoco a questo oggetto
onMouseDown	Avviene quando l'utente preme e tiene premuto il pulsante del mouse sul bottone
onMouseUp	Avviene quando l'utente rilascia il pulsante del mouse sul bottone

Gli oggetti del CORE: string, date, math

Gli oggetti del core (nucleo) sono oggetti di Javascript utilizzabili sia sul client che sul server, e rappresentano il principale obiettivo di standardizzazione del comitato ECMA. Essi costituiscono un insieme di oggetti indipendenti dal tipo di applicazioni realizzabili e di utilità generale. I più importanti sono l'oggetto string, l'oggetto date e l'oggetto math.

String

Una classe di oggetti per rappresentare una qualunque sequenza di caratteri stampabili. Molti oggetti Javascript sono stringhe (per esempio, il nome degli oggetti, il valore, etc.)

Proprietà di string (elenco parziale)

length	Il numero di caratteri della stringaMetodi di string (elenco parziale)
charAt	restituisce il carattere all'indice specificato

	es.: "Esempio".charAt(3) vale "m"
charCodeAt	Restituisce il codice ISO-Latin-1 (ASCII) del carattere specificato.
concat	Concatena il testo di due stringhe in una nuova.
indexOf	ritorna la posizione della prima occorrenza del valore specificato nella stringa
	Es.: "Esempio".indexOf("m") vale 3
lastIndexOf	ritorna la posizione dell'ultima occorrenza del valore specificato nella stringa
	Es.: "mamma".lastIndexOf("m") vale 3
substr	restituisce una stringa che inizia alla locazione specificata della stringa ed é lunga un numero specificato di caratteri
	Es.: "Esempio".substr(2,3) vale "emp"
substring	restituisce una stringa che inizia alla locazione specificata della stringa ed arriva fino alla seconda locazione specificata:

	Es.: "Esempio".substring(2,3) vale "em"
toLowerCase	Restituisce la stringa tutta in minuscolo
	Es.: "Ecco".toLowerCase() vale "ecco"
toUpperCase	Restituisce la stringa tutta in maiuscolo
	Es.: "Ecco".toUpperCase() vale "ECCO"

Date

Una classe di oggetti che permette di trattare con date ed ore. Un oggetto di tipo date va sempre esplicitamente creato con il metodo new:

birthday = new Date("December 17, 1995 03:24:00")

birthday = new Date(95,11,17)

birthday = new Date(95,11,17,3,24,0)

Metodi di date (elenco parziale)

getDate	Restituisce il giorno del mese della data specificata
getDay	Restituisce il giorno della settimana della data specificata.
getHours	Restituisce l'ora della data specificata
getMinutes	Restituisce i minuti della data specificata.
getMonth	Restituisce il mese della data specificata.
getSeconds	Restituisce i secondi della data specificata.

getTime	Il numero di millisecondi dalla mezzanotte del 1 gennaio 1970 (la prima data specificabile con gli oggetti di tipo date). Utile per avere un numero sempre diverso, e anche per assegnare a due date lo stesso valore: Es.: a.setTime(b.getTime())
getTimezoneOffset	Restituisce la differenza di fuso orario tra la data specificata e il tempo medio di Greenwich
getYear	Restituisce l'anno della data specificata.
setDate	Attribuisce il giorno del mese della data specificata.
setHours	Attribuisce l'ora della data specificata.
setMinutes	Attribuisce i minuti della data specificata.
setMonth	Attribuisce il mese della data specificata.
setSeconds	Attribuisce i secondi della data specificata.
setTime	Attribuisce un valore ad una data (in millisecondi dalla mezzanotte del 1 gennaio 1970)
setYear	Attribuisce l'anno della data specificata.

Math

Math è uno specifico oggetto (statico) che dà accesso a numerose funzioni e costanti matematiche. Non è possibile definire oggetti di questo tipo, ma esso esiste sempre ed é sempre utilizzabile. Va usato sempre nella notazione col punto:

math.max(2,5) vale 5

math.sqrt(9) vale 3

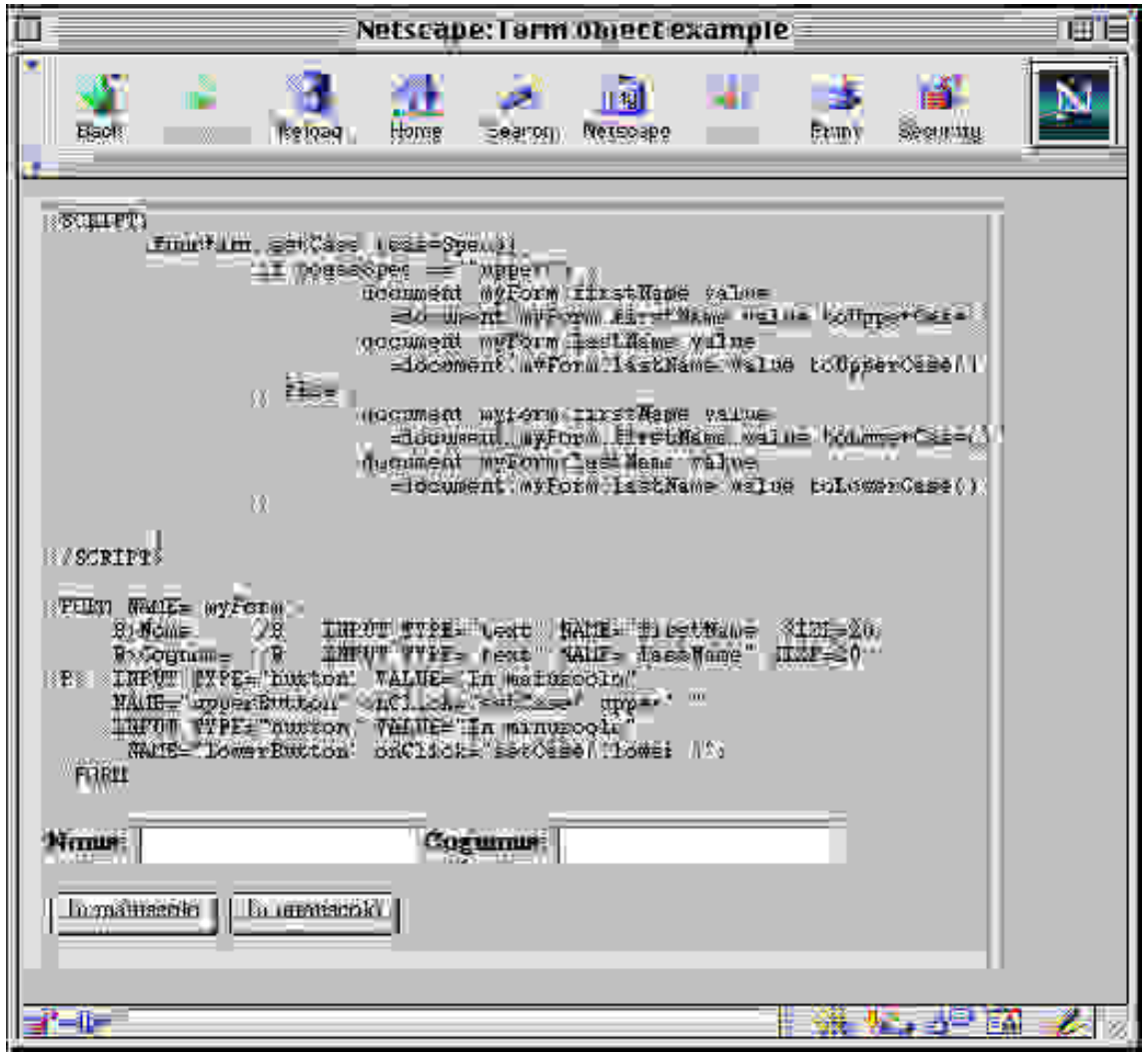
Proprietà di math (elenco parziale)

E	Costante di Eulero, circa 2.718.
PI	Pi greco, circa 3.14159.
SQRT1_2	radice quadrata di 1/2; circa 0.707.
SQRT2	radice quadrata di 2, circa 1.414.

Metodi di math (elenco parziale)

abs	Restituisce il valore assoluto del numero Es.: math.abs(-5) = 5
cos	Restituisce il coseno del numero
exp	Restituisce il valore e elevato al numero
log	Restituisce il logaritmo in base e del numero
max	Restituisce il maggiore tra due numeri
min	Restituisce il minore tra due numeri
pow	Restituisce l'elevamento a potenza di due numeri Es.: math.pow(3,2) = 3^2 = 9
random	Restituisce un numero casuale compreso tra 0 e 1
round	Restituisce l'intero più vicino ad un numero Es.: math.round(4.7) = 5 math.round(4.3) = 4
sin	Restituisce il seno del numero
sqrt	Restituisce la radice quadrata del numero Es.: math.sqrt(9) = 3

Esempio 8


Netscape: Form Object example
Back
Reload
Home
Search
Netscape
Print
Security
Nome:
Cognome:

Esempio 9

La calcolatrice

Il codice HTML della calcolatrice

```
<SCRIPT>
   var display, oldregister, operator, nuovo, real, esp ;

   function init() {
       real = false
       nuovo = true
       register = 0
       oldregister = 0
       document.calcolatrice.Display.value = register
   }

   function Number(n) {
       if (nuovo) {
           register = n
           nuovo = false
           real = false
           esp = 0
       } else {
           if (real) {
               esp++
               register += n/Math.pow(10,esp)
           } else {
               register = register*10+n
           }
       }
       document.calcolatrice.Display.value = register
   }
```

```
function Operator(x) {
    nuovo = true
    operator = x
    oldregister = register
    register = 0
}

function Exec() {
    if (operator == "+") {
        register = oldregister + register
    } else if (operator == "-") {
        register = oldregister - register
    } else if (operator == "*") {
        register = oldregister * register
    } else if (operator == "/") {
        register = oldregister / register
    }
    nuovo = true
    operator = ""
    oldregister = 0
    document.calcolatrice.Display.value = register
}

function SignChange() {
    register = - register
    document.calcolatrice.Display.value = register
}
```

```
function DecimalDot() {
      if (nuovo) {
          register = 0
      }
      real = true
      nuovo = false
   }

   function ClearValue(){
      register = 0
      nuovo = true
      document.calcolatrice.Display.value = register
   }

</SCRIPT>
</head>

<body>
<FORM NAME="calcolatrice">
<TABLE BORDER=1>
   <TR><TD COLSPAN=4 ALIGN=CENTER><INPUT NAME="Display" TYPE=TEXT SIZE=20
VALUE="0">
   <TR><TD><INPUT TYPE=BUTTON NAME="B7" VALUE=" 7 " OnClick='Number(7)'>
       <TD><INPUT TYPE=BUTTON NAME="B8" VALUE=" 8 " OnClick='Number(8)'>
       <TD><INPUT TYPE=BUTTON NAME="B9" VALUE=" 9 " OnClick='Number(9)'>
       <TD><INPUT TYPE=BUTTON NAME="BDIV" VALUE="/" OnClick='Operator("/")'>
   <TR><TD><INPUT TYPE=BUTTON NAME="B4" VALUE=" 4 " OnClick='Number(4)'>
       <TD><INPUT TYPE=BUTTON NAME="B5" VALUE=" 5 " OnClick='Number(5)'>
       <TD><INPUT TYPE=BUTTON NAME="B6" VALUE=" 6 " OnClick='Number(6)'>
       <TD><INPUT TYPE=BUTTON NAME="BMULT" VALUE="*" OnClick='Operator("*")'>
   <TR><TD><INPUT TYPE=BUTTON NAME="B1" VALUE=" 1 " OnClick='Number(1)'>
```

```
      <TD><INPUT TYPE=BUTTON NAME="B2" VALUE=" 2 " OnClick='Number(2)'>
      <TD><INPUT TYPE=BUTTON NAME="B3" VALUE=" 3 " OnClick='Number(3)'>
      <TD><INPUT TYPE=BUTTON NAME="BMIN" VALUE="-" OnClick='Operator("-")'>
   <TR><TD><INPUT TYPE=BUTTON NAME="BPM" VALUE="+/-" OnClick='SignChange()'>
      <TD><INPUT TYPE=BUTTON NAME="B0" VALUE=" 0 " OnClick='Number(0)'>
      <TD><INPUT TYPE=BUTTON NAME="BPT" VALUE=" . " OnClick='DecimalDot()'>
      <TD><INPUT TYPE=BUTTON NAME="BPLUS" VALUE="+" OnClick='Operator("+")'>
   <TR><TD COLSPAN=2 ALIGN=CENTER><INPUT NAME="Equal" TYPE=BUTTON VALUE="
=      " OnClick='Exec()'>
      <TD COLSPAN=2 ALIGN=CENTER><INPUT NAME="Cancel" TYPE=BUTTON VALUE="
C      " OnClick='ClearValue()'>
</TABLE>
</FORM>
<SCRIPT>
   init();
</SCRIPT>
</body>

</html>
```