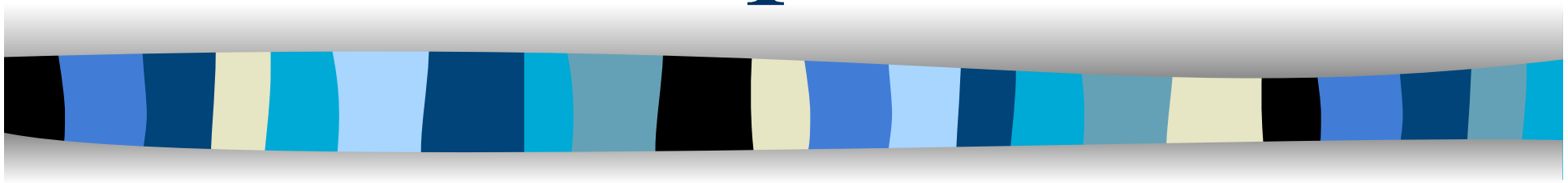
JavaScript - 6



Browser Object Model
(BOM)

L'oggetto **window**

- L'oggetto **window** possiede varie proprietà e metodi
  - Esiste un nucleo di tali proprietà e metodi, che esamineremo in seguito, supportati da tutti i browser
- Per verificare quali sono le proprietà dell'oggetto **window** supportate è sufficiente eseguire lo script della slide successiva

Stampa proprietà

```
<SCRIPT TYPE="text/javascript">
var prop="";
var i=0;
for (prop in window) {
    i++;
    document.write(i + " <b>");
    document.write(prop+ ":</b> ");
    document.write(window[prop]+"<br>");
}
</SCRIPT>
```

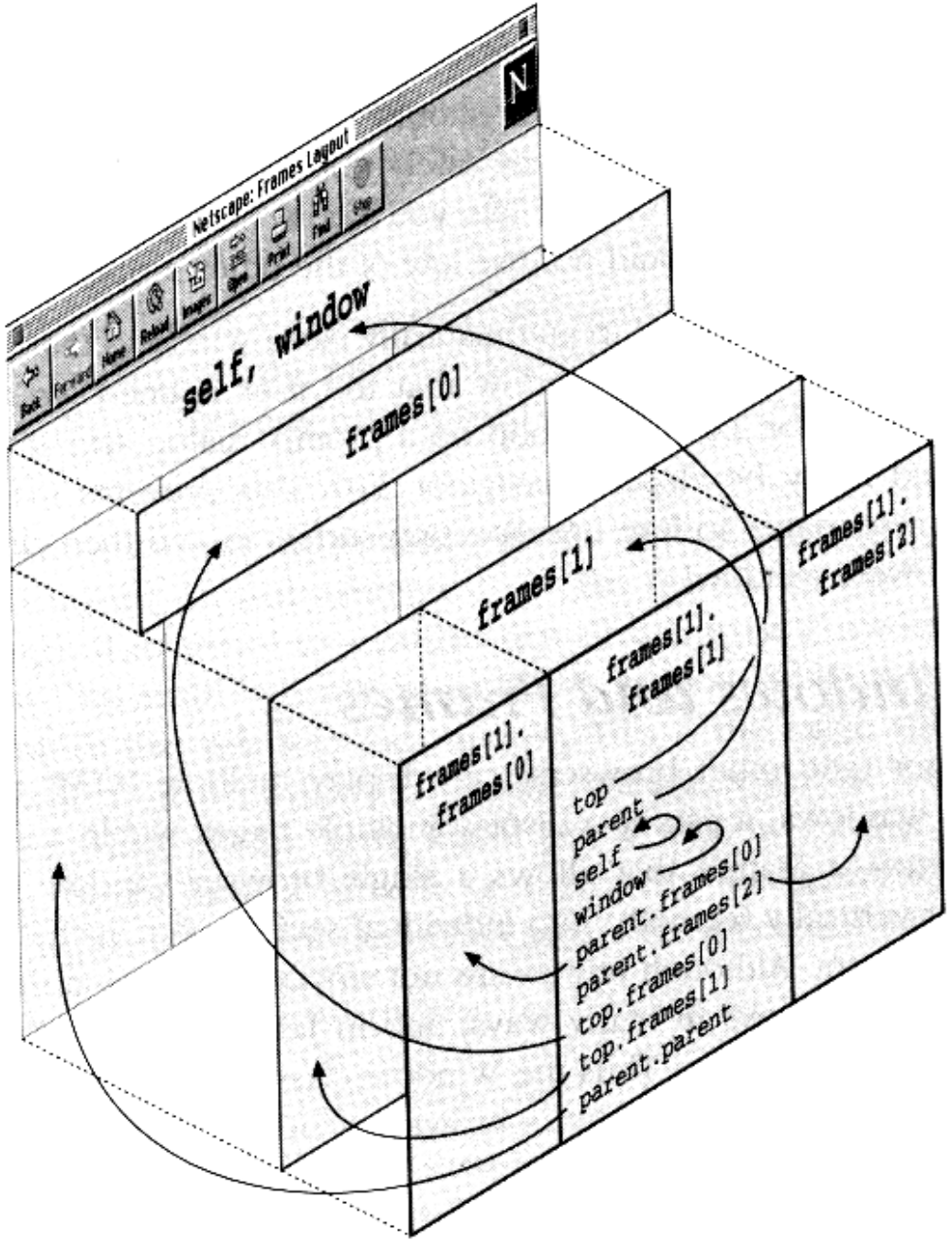
Proprietà principali – 1

- defaultStatus e status
  - Indicano la linea di testo che compare sulla linea di stato del browser
  - defaultStatus = “benvenuto nel mio sito”
    - A volte sono a sola lettura per problemi di sicurezza (in relazione al phishing)
- document
  - Rappresenta il documento HTML mostrato nella finestra del browser
- window e self
  - Proprietà auto-referenziali

Proprietà principali – 2

- frames[]
  - Array di oggetti Window che rappresentano i frame della finestra
- length
  - Indica il numero di frame contenuti nella finestra
- parent
  - Riferimento alla finestra genitore del frame corrente
- top
  - Riferimento alla finestra principale che contiene il frame corrente

Graficamente...



Proprietà principali – 3

- name
  - nome della finestra (assegnato con open())
- opener
  - nome della finestra che ha aperto quella corrente usando open()
- history
  - Riferimento all'oggetto **History** che rappresenta i percorso di navigazione seguito
- location
  - Riferimento all'oggetto **Location** che rappresenta l'URL mostrata nella finestra

Altre proprietà

- innerHeight innerWidth
  - Dimensione dell'area di visualizzazione
- outerHeight outerWidth
  - Dimensione del browser incluso barre
- pageXOffset pageYOffset
  - Numero di pixel la pagina è stata scrollata
- screenLeft screenX
  - Distanza da lato sinistro dello schermo
- screenTop screenY
  - Distanza da lato superioredello schermo

Metodi principali – 1

- alert(), prompt(), confirm()
  - Già esaminati
- close()
  - Chiude la finestra
- open()
  - Apre una nuova finestra
- moveBy(dx,dy) e moveTo(x,y)
  - Muovono la finestra

Non supportati da tutti i browser per motivi di sicurezza

Non accetta valori negativi, ma IE <6 li accettava …..

Metodi principali – 2

- resizeBy(dx,dy) e resizeTo(x,y)
  - Ridimensionano la finestra
- scrollBy(dx,dy) e scrollTo(x,y)
  - Scrollano il contenuto della finestra
- print()
  - Invoca la dialog box di stampa
- focus() e blur()
  - Assegnano e rimuovono il focus a/da una specifica finestra

Cosa succede oggi?

- Si possono muovere e/o ridimensionare solo le finestre del browser che sono state aperte con Javascript

- Non è possibile ridimensionare la finestra principale (Firefox) o finestre contenute in browser che già mostrano altre schede (tab)

Metodi principali – 3

- setInterval( )
  - invoca una funzione o valuta un'espressione ogni qualvolta trascorre un certo numero di millisecondi
- clearInterval()
  - elimina gli intervalli per le azioni settate con setInterval( )
- setTimeout( )
  - invoca una funzione o valuta un'espressione dopo che sono trascorsi un certo numero di millisecondi
- clearTimeout( )
  - elimina i timeout settati con setTimeout( )

open()

- Sintassi (restituisce un riferimento a window)

window.open(url, nome, caratteristiche, rimpiazza)

  - url indica l'URL del documento da caricare
  - nome è una stringa opzionale che indica il nome da assegnare alla finestra per riferimenti futuri
  - caratteristiche è una stringa che indica quali caratteristiche deve possedere la finestra
    - Altezza, larghezza, barre, ….
  - rimpiazza è un argomento booleano che indica se la pagina caricata nella nuova finestra deve creare una nuova entry nella storia della navigazione (history) oppure sostituisce l'entry attuale

caratteristiche

- Sono delle parole chiavi opzionali seguite da un valore
  - height, width, location, status
  - left, top (coordinate dell'angolo superiore sinistro della finestra del browser)
- alcune caratteristiche sono booleane
  - toolbar, menubar, scrollbars, resizable, directories

Pagina principale

```
<SCRIPT LANGUAGE="javascript" TYPE="text/javascript">
    leftPos = screen.width-225;
    car = "width=225,height=200,left="+leftPos+",top=0"
   newWindow = window.open("indice.html", "newWin", car);
</SCRIPT>

<BODY>
<CENTER>
    <H1>Questa &egrave; la pagina principale</H1>
    <H1>Qui verrano mostrate tutte le pagine del sito</H1>
</CENTER>
</BODY>
```

indice.html

```
<SCRIPT LANGUAGE="javascript" TYPE="text/javascript">
    function updateParent(newURL) {
        opener.location.href = newURL; }
</SCRIPT>
<H1 style="align:center;" >Pannello di Controllo</H1>
<UL>
<LI><A HREF="javascript:updateParent('paginaUno.html')">
        Pagina uno</A><BR>
<LI>< A HREF="javascript:updateParent('paginaDue.html')">
        Pagina due</A><BR>
<LI>< A HREF="javascript:updateParent('paginaTre.html')">
        Pagina tre</A>
</UL>
```

Se newURL è un'url esterna al sito, allora updateParent non potrà più aggiornare la pagina opener – **same origin policy**

setTimeout()

- Setta un timeout per l'esecuzione di codice JavaScript
- timeoutID = setTimeout(cod,ms);
  - cod è una stringa contenente istruzioni JavaScript da eseguire dopo che sono trascorsi ms millisecondi.
  - In genere, cod è il nome di una funzione JavaScript
    - Può avere dei parametri costanti
  - Restituisce un identificativo utilizzato per far riferimento al codice da eseguire

clearTimeout()

- Cancella l'esecuzione un timeout schedulato con setTimeout()
- Sintassi

  clearTimeout(timeoutID)
  - timeoutID è l'identificativo restituito da setTimeout

- Una funzione può rischedulare se stessa
  - basta inserire una chiamata a setTimeout come ultima istruzione della funzione stessa

Esempio 1 – HTML

```
<IMG name="foto" src="prima.png">
<FORM name="modulo">
Fra quanti secondi vuoi cambiare l'immagine?
    
<INPUT TYPE="text" name="secondi"
   size="2">   
<INPUT TYPE="button" value="Cambia"
   onClick="settaT()">    
<INPUT TYPE="button" value="Annulla"
   onClick="annulla()">
</FORM>
```

Esempio 1 – JavaScript

```
var tID = 0; // variabile globale
function settaT() {
var sec = parseInt(document.modulo.secondi.value);
if(!tID) // controllo importante evita di settare un nuovo
         // timeOut se un timeOut è stato già settato
    tID= setTimeout(cambia,sec*1000);
}

function cambia() {document.foto.src="seconda.png";}

function annulla() {
if (tID)
    clearTimeout(tID);  tID=0;}
```

Esempio 2 – HTML

```
<BODY>
Esempio di set/clearTimeout<P>

<IMG SRC="prima.gif" NAME="immagine"
        onMouseOver="alterna()"
        onMouseOut="clearTimeout(tID)" >
</BODY>
```

Esempio 2 – JavaScript

```
<SCRIPT TYPE="text/javascript">
var indice = 0; // Indica quale immagine è caricata
var immagini = new Array();
immagini[0] = "prima.gif";
immagini[1] = "seconda.gif";
var tID=0;

function alterna(){
//indice = (indice == 1) ? 0:1;
indice = (indice +1)%2;
document.immagine.src = immagini[indice];
tID = setTimeout(alterna,100); }
</SCRIPT>
```

Esempio orologio – HTML

```
<body onload="mostraora()">
<h3>Ora locale</h3>
<form name="orologio">
<input type="text" name="tempo" size="7">
</form>
</body>
```

```
input {
        border: solid red 2px;
        text-align: center;
        padding: 2px;
}
```

Esempio orologio – JavaScript

```
function mostraora() {
    var d=new Date(); h=d.getHours();
    m=d.getMinutes(); s=d.getSeconds();
    if(s<=9) s="0"+s;
    if(m<=9) m="0"+m;
    if(h<=9) h="0"+h;
    var time = h+":"+m+":"+s;
    document.orologio.tempo.value = time;
    setTimeout(mostraora,1000);
}
```

Nota

- Al posto di

```
tID= setTimeout(mostraora,sec*1000);
```

avremmo potuto utilizzare

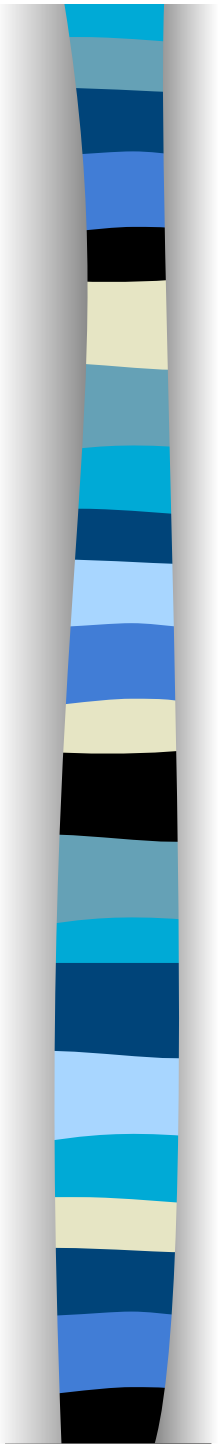
```
tID= setTimeout("mostraora()",sec*1000);
```

Passare una stringa - "mostraora()" - implica un'invocazione di eval sulla stringa passata

Qualche altro dettaglio

- Come passare a setTimeout funzioni che prendono in input dei parametri?

```
<form name="modulo">
<input type="button" value="mostra 1" onclick="attiva1(’paperino')">
<input type="button" value="mostra 2" onclick="attiva2('pippo')">
<input type="button" value="mostra 3" onclick="attiva3('topolino')">
<input type="button" value="mostra 4" onclick="attiva4(’minnie')">
<input type="button" value="arresta"    onclick="arresta()">
</form>
```



```
//variabili globali:
        var tid = 0;
        var strOut = ""

function arresta(){
 if(tid) {
    clearTimeout(tid);
    tid = 0;
  }}

function mostra(param) {
   alert(param);
   tid=0; }
```

```
function attiva1(val){
strOut = val;
if (!tid)  tid = setTimeout( function() { mostra(strOut)}, 1000);
}
```

Dopo 1 secondo manda in esecuzione la funzione definita con la funzione letterale

```
function attiva2(val){
strOut = val;
if (!tid)    tid = setTimeout( "mostra(strOut)", 1000);
}
```

```
function attiva3(val){
strOut = val;
if (!tid)   tid = setTimeout(mostra(strOut) , 1000);
}
```

Crea problemi!!!! Invoca prima mostra e poi setTimeout

Mancano le virgolette!!

Quali sono i problemi?

- La funzione attiva3 invoca mostra(strOut) immediatamente
  - non aspetta 1000 millisecondi
- La funzione attiva3 attiva il timeout sul valore ritornato da mostra(strOut)
  - valore nullo
- Il timeout non viene mai azzerato

```
function attiva4(val){
strOut = val;
if (!tid)   tid = setTimeout( f , 1000);
}
```

```
if (!tid)  tid = setTimeout( function() { mostra(strOut)} , 1000);
```

è più o meno equivalente a

```
if (!tid)  tid = setTimeout( f , 1000);
```

dove f è definita come

```
function f() {mostra(strOut)}
```

setTimeout e funzioni con parametri

- Per invocare, attraverso setTimeOut, una funzione che prende in input dei valori
  - Creiamo una stringa contenente la chiamata a funzione
  - Passiamo la chiamata a funzione a setTimeOut
- Nel prossimo esempio stampiamo i numeri da 1 a 9 in una casella di testo, essi vengono stampati uno al secondo

Esempio – HTML

```
<BODY>
<form name="modulo">
<input type="text" name="valore" size="1">
</form>
<SCRIPT TYPE="text/javascript">
var n = 1;
stampaNum(n);   // visualizza il numero n nella
                // casella di testo valore
</SCRIPT>
</BODY>
```

Esempio – JavaScript

```
<SCRIPT TYPE="text/javascript">

function stampaNum(x){
if (x < 10) {
    document.modulo.valore.value = x;
    x++
    var func = "stampaNum(" + x + ");";
    setTimeout(func,1000);
    }
} </SCRIPT>
```

Non ci sono variabili globali

setInterval()

- Permette di eseguire periodicamente del codice (o funzione) JavaScript
- Sintassi

  intID = setInterval(cod,inter)

  - cod è il codice da eseguire
  - inter intervallo espresso in millisecondi tra invocazioni successive

  intID = setInterval(funz, inter, par_1, par_2, …)

  - funz funzione JavaScript
  - par_1, par_2 … parametri di funz

clearInterval()

- Cancella l'esecuzione periodica di codice JavaScript schedulato con setInterval()
- Sintassi

  clearInterval(intervalID)

  – intevalID è l'identificativo restituito da setInterval()
- Con setInterval() si può realizzare facilmente uno *slideshow*

Esempio setInterval – HTML

```
<TABLE> <TR>
<TD> <IMG SRC="a.gif" height="200"
          NAME="immagine" border="0">
<TD>
<FORM>
<INPUT TYPE="button" VALUE=" Inizia "
       onClick="inizia()"><P>
<INPUT TYPE="button" VALUE="Ferma"
       onClick="ferma()">
</FORM>
</TABLE>
```

slideshow

Esempio setInterval – JavaScript (1)

```
<SCRIPT TYPE="text/javascript">
var intID = 0;
var indice = 0;
var immagini = new Array();
for(i=0;i<4;i++)
   immagini[i] = new Image();
// pre-carichiamo le immagini
immagini[0].src = "a.gif"; immagini[1].src = "b.gif";
immagini[2].src = "c.gif"; immagini[3].src = "d.gif";
```

Esempio setInterval – JavaScript (2)

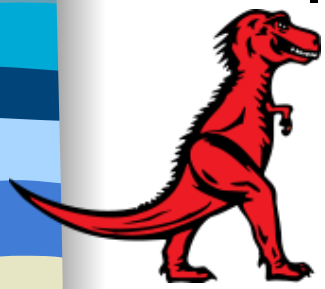
```
function inizia() {
 if(!intID) //IMPORTANTE!!!
   intID = setInterval("cambia()",1000); }

function cambia() {
indice = (indice == 3) ? 0:++indice;
document.immagine.src = immagini[indice].src;  }

function ferma() { if(intID) { clearInterval(intID);
    intID=0; } }
</SCRIPT>
```

L'oggetto navigator

- Rappresenta il browser in uso e permette di ottenere informazioni su nome, versione …
- Ogni browser può avere proprietà non standard
- Per accedere ad una proprietà si usa la sintassi
  - navigator.NomeProprietà

Proprietà di navigator – 1

- appCodeName
  - Il nome in codice del browser



    - “Mozilla” sia per IE che NN
      - Usato all’inizio per il software di Netscape Navigator
      - Contrazione di *Mosaic killer* (in slang killer = killa, quindi Mozilla = Moz+illa)
- appName
  - nome ufficiale del browser
    - “Netscape” per Safari, Firefox, Chrome
    - “Microsoft Internet Explorer” per IE

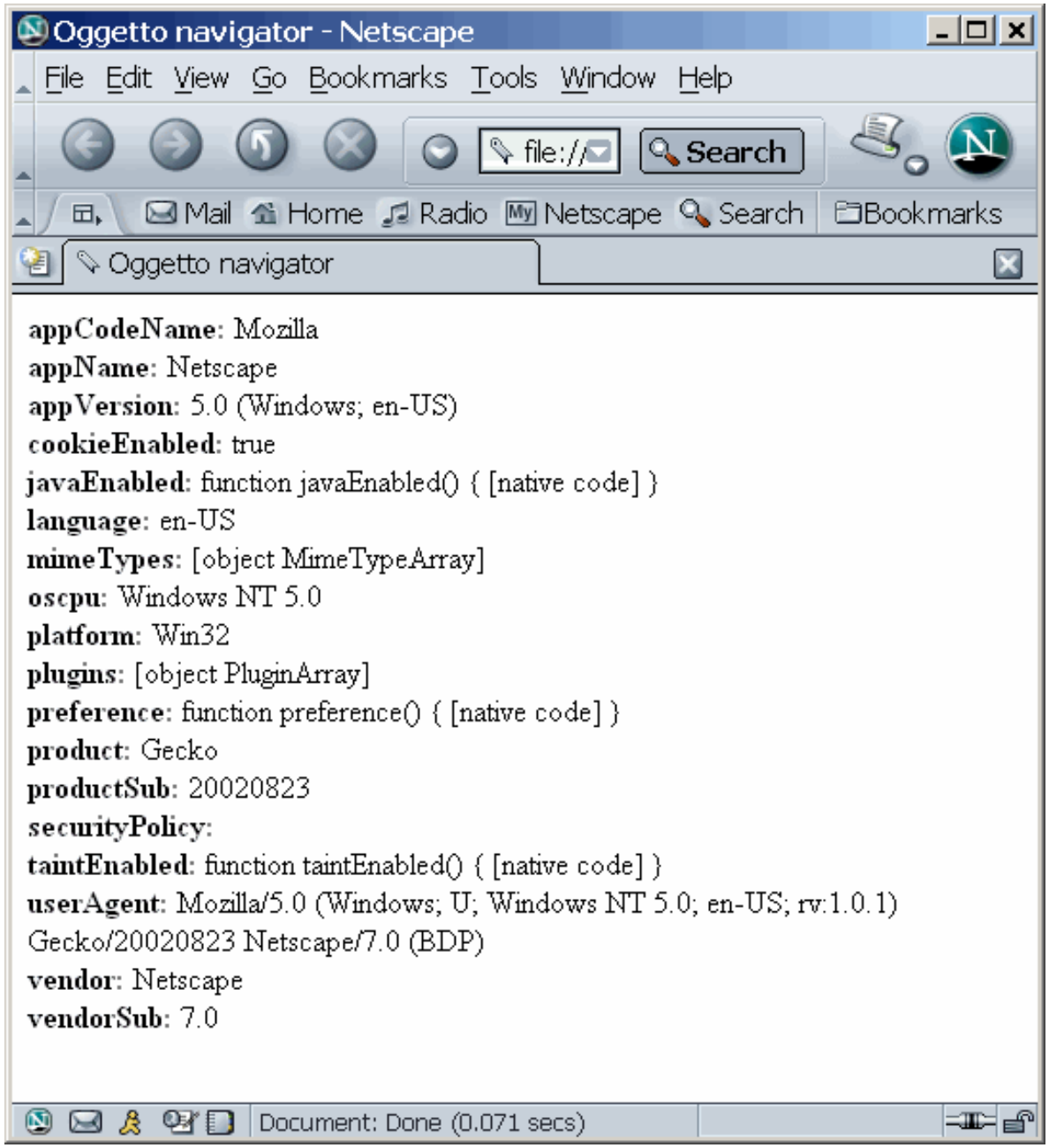
Proprietà di navigator – 2

- appVersion
  - Versione del codice del browser ed altri dati
    - “4.0 (compatible; MSIE 6.0; Windows NT 5.0)” per IE6
    - “5.0 (Windows; en-US)” per NN7
- userAgent
  - stringa inviata dal browser al server nelle richieste di pagine
    - Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)
    - Mozilla/5.0 (Windows; U; Windows NT 5.0; en-US; rv: 1.0.1) Gecko/20020823 Netscape/7.0 (BDP)
- platform
  - sistema operativo dove gira il browser

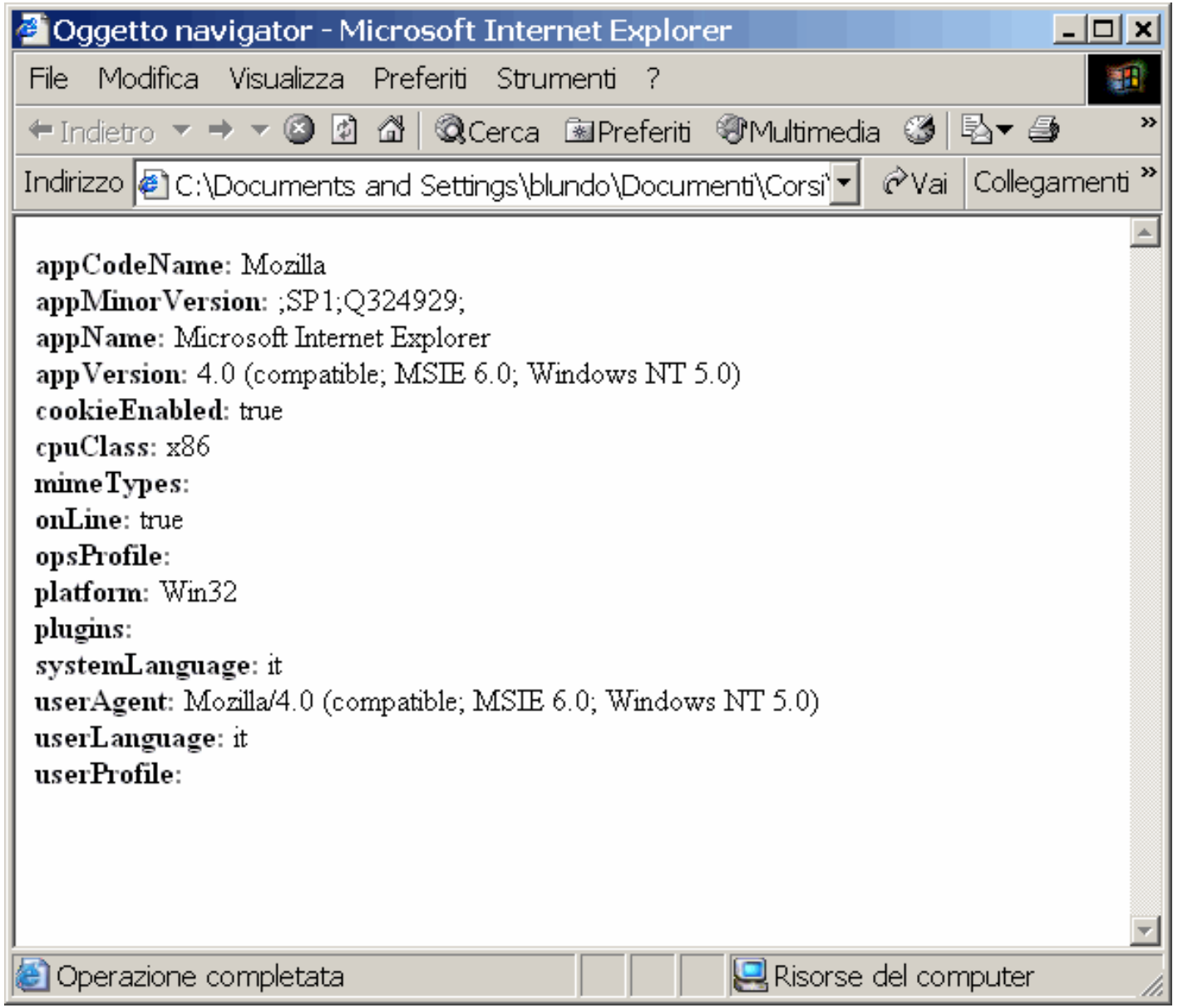
Proprietà di navigator – 3

- cookieEnabled
  - Valore booleano indicante se i cookie sono abilitati
- language
  - lingua di default del browser (solo NN4+)
- userLanguage
  - Lingua preferita dall'utente (solo IE4+) equivalente di language
- javaEnabled()
  - Verifica se Java è disponibile e supportato dal browser

L'oggetto navigator di NN



L'oggetto navigator di IE6



L'oggetto screen

- Fornisce informazioni sulla risoluzione dello schermo dell'utente e sul numero di colori che supporta
- Si potrebbero usare le informazioni contenute in screen per decidere la grandezza o numero di colori delle immagini da caricare nel documento
- Per accedere ad una proprietà si usa la sintassi
  - screen.NomeProprietà

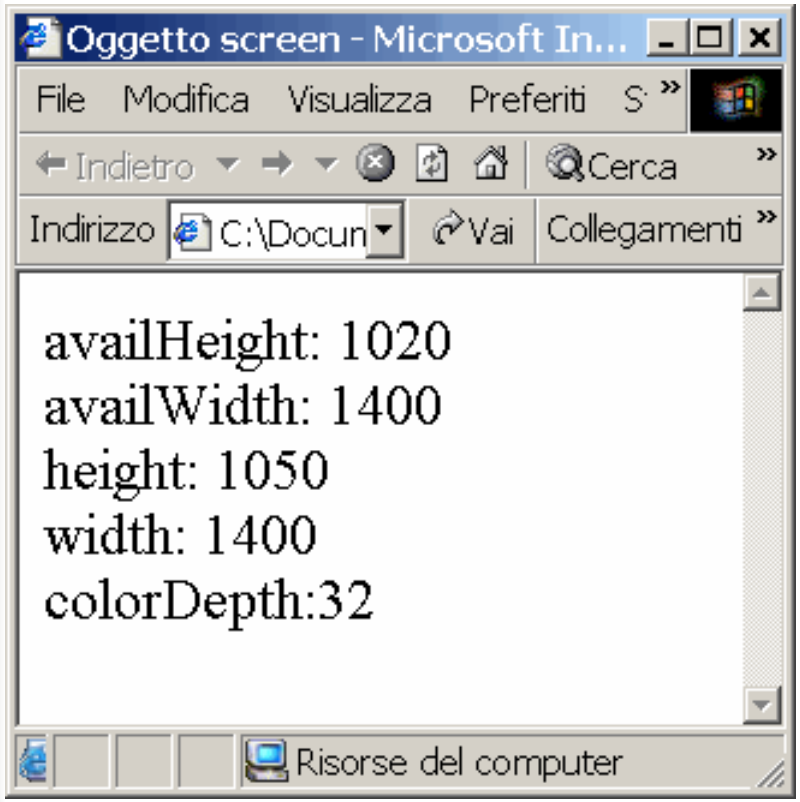
Proprietà di screen – 1

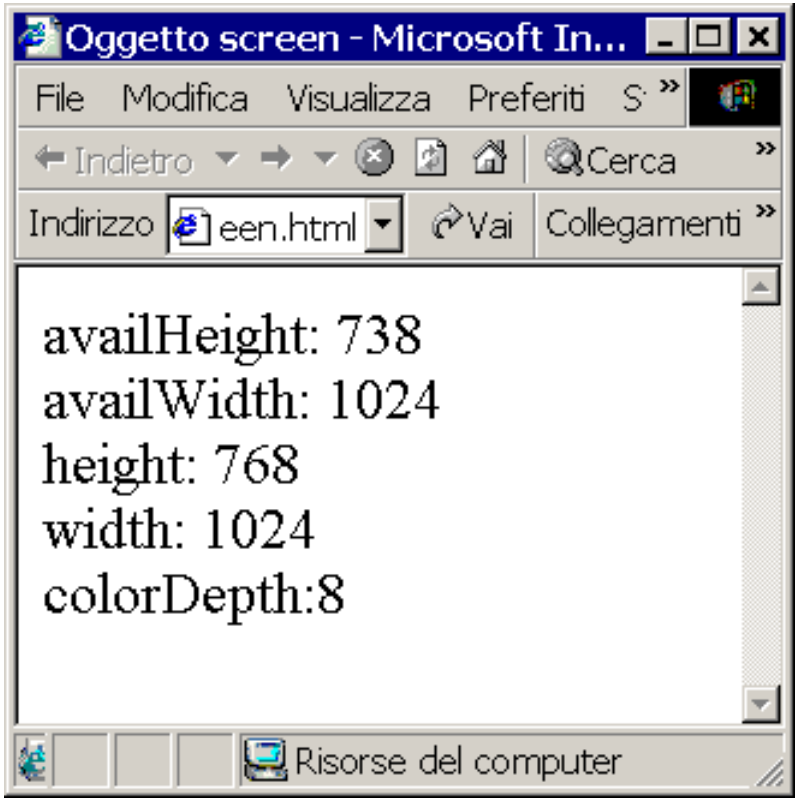
- availHeight
  - Specifica l'altezza in pixel disponibile dello schermo (esclude l'altezza della barra delle applicazioni se esiste)
- availWidth
  - Specifica la larghezza in pixel disponibile dello schermo (esclude la larghezza di eventuali barre di applicazioni)

Proprietà di screen – 2

- colorDepth
  - Specifica il logaritmo in base due del numero di colori allocati dal browser e disponibili per la visualizzazione delle immagini
- height
  - Specifica l'altezza in pixel dello schermo
- width
  - Specifica la larghezza in pixel dello schermo

Esempio IE6 + Windows 2000





L'oggetto location

- Rappresenta l'URL del documento che è correntemente mostrato nella finestra
- È possibile accedere a singole parti dell'URL
  - dominio, host, ...
- Tutte le proprietà sono accessibili in lettura e scrittura
  - basta modificarle per caricare un nuovo documento

Proprietà di location

- href
  - URL completo del documento attualmente visualizzato nella finestra del browser
- host
  - hostname e numero di porta
- Altre proprietà sono
  - protocol://hostname:port/pathname?search#hash

Esempio

http://www.abc.com:666/catalog/search.php?query=JS&match=2#result

protocol → “http:”

hostname → “www.abc.com”

port → “666”

host → “www.abc.com:666”

pathname → “/catalog/search.php”

search → “?query=JS&match=2”

hash → “#result”

Metodi di location

- reload()
  - ricarica l'URL corrente dalla cache o dal server
- replace(url)
  - carica una nuova pagina passata come argomento url
- Per caricare una nuova pagina nel browser è sufficiente scrivere
  - location.href = url-da-caricare

L'oggetto history

- Tiene traccia della cronologia di navigazione del browser
- Possiede tre metodi
  - back()
    - Va indietro nella cronologia
  - forward()
    - Va avanti nella cronologia
  - go(±Numero)
    - Va ±Numero elementi avanti/indietro nella cronologia

L'oggetto **document**

- Forse è l'oggetto più importante
  - responsabile di tutto quello che compare sulla pagina
  - controparte del tag **<BODY>**
- permette di implementare le caratteristiche dinamiche di HTML
- Alcune sue proprietà e metodi sono dipendenti dal browser

Proprietà principali di **document** – 1

- alinkColor, linkColor, vlinkColor
  - Colori dei link attivi, non visitati e visitati
- bgColor, fgColor
  - Colore di sfondo e di primo piano
- cookie
  - Proprietà che permette di leggere e scrivere cookie associati al documento
- domain
  - Dominio del documento. Usato per settare lo stesso dominio in finestre di un sito che vengono da web server differenti dello stesso dominio
    - E.g., www.oreilly.com e search.oreilly.com

Proprietà principali di **document** – 2

- lastModified
  - Contiene la data di ultima modifica del documento
- referrer
  - URL del documento contenente il link che abbiamo seguito per raggiungere il documento corrente
- title
  - Il testo tra **<TITLE>** e **</TITLE>**
- URL
  - Specifica l'URL da cui il documento è stato scaricato
- location
  - Sinonimo **deprecato** di URL (al posto di document.location si usa location.href)

Proprietà principali di **document** – 3

- applets[]
  - array delle applet contenute nel documento
- forms[]
  - Array dei moduli, **maggiori dettagli in seguito**
- embeds[]
  - Array degli oggetti contenuti nel documento (inseriti tramite <OBJECT> o <EMBED>)
- plugins[]
  - Sinonimo di embeds[], poco usato

Proprietà principali di **document** – 4

- anchors[]
  - Array delle ancore di un documento (testo o immagine della pagina che è target di un link)
  - Codificati con <A NAME=“ …” >
  - Proprietà: length e name
    - Uso: anchors.length e anchors[i].name
- links[]
  - Array dei link di un documento
  - Codificati con <A HREF=“ …” >
  - Proprietà: le stesse di **location**
- images[]
  - Array delle immagini presenti nel documento

Alcuni metodi di **document**

- open()
  - Apre il documento su cui andare a scrivere. Non necessario, la prima write apre automaticamente il documento
- write()
- close()
  - Serve per chiudere il documento.
  - Inserire sempre una chiamata a questo metodo dopo che abbiamo finito di scrivere nella finestra del browser
    - In caso contrario il browser non fermerà il simbolo di caricamento del documento