Guida Java Script

JavaScript è un linguaggio script per il Web. JavaScript è usato viene usato in milioni di pagine Web per aggiungere funzionalità, validare forms, individuare i browser usati e molto altro. JavaScript è facile da imparare!
Cos'è JavaScript?
- JavaScript è stato creato per aggiungere interattività alle pagine HTML
- JavaScript è un linguaggio di scripting, non un vero e proprio linguaggio di programmazione
- JavaScript è normalmente incorporato direttamente nelle pagine HTML
- JavaScript è un linguaggio interpretato (questo significa che uno script javascript non deve essere precompilato per poter essere esguito, a differenza dei linguaggi di programmazione veri e propri)
- JavaScript può essere usato da chiunque senza acquistare una licenza Java e JavaScript sono la STESSA COSA?

Certamente NO!

Java e JavaScript sono due linguaggi completamente differenti in tutto!

Java (sviluppato da Sun Microsystems) è un potente e molto più complesso linguaggio di programmazione vera e propria, pari a C e C++. Cosa può fare JavaScript?
- JavaScript fornisce uno strumenti di programmazione ai designer HTML - i designers HTML non sono generalmente dei programmatori, ma JavaScript ha una sintassi per la creazione di script così semplice da poter essere usata tranquillamenta da chiunque abbia una conoscenza di HTML e XHTML!

JavaScript può mettere testo dinamico in una pagina HTML - Una semplice stringa JavaScript del tipo:

document.write("

" + name + "

")

scrive una variabile di testo in una pagina HTML
- JavaScript è in grado di interagire con gli eventi - Un JavaScript può essere impostato per eseguire un'azione quando succede qualcosa, come alla fine del caricamento di una pagina o quando un utente fa clic su un elemento HTML
- JavaScript è in grado di leggere e scrivere gli elementi HTML - Uno scritp JavaScript è in grado di leggere e modificare il contenuto di un elemento HTML
- JavaScript può essere utilizzata per convalidare i dati- JavaScript può essere utilizzato per la convalida di dati prima che che gl istessi vengano sottoposto al server.
- JavaScript può essere utilizzato per individuare il browser dei visitatori - JavaScript può essere utilizzato per individuare il browser del visitatore, e - a seconda del browser - caricare un'altra pagina specificamente progettata per tale browser.
- JavaScript può essere usato per creare i cookie - JavaScript può essere usato per archiviare e recuperare le informazioni sul computer del visitatore Il suo nome originale è ECMAScript

Il nome ufficiale di JavaScript è "ECMAScript". Lo standard è stato sviluppato e mantenuto dalla ECMA organisation.

ECMA-262 è lo standard ufficiale JavaScript. Lo standard è basato su JavaScript (Netscape) e JScript (Microsoft).

Questo linguaggio fu inventato da Brendan Eich alla Netscape (con Navigator 2.0), ed è apparso in tutti i browser Netscape e Microsoft dal 1996.

Lo sviluppo di ECMA-262 è iniziata nel 1996, e la prima edizione fu adottata dalla ECMA Assemblea Generale nel giugno 1997.

Fu quindi approvato a livello internazionale ISO (ISO / IEC 16262) nel 1998.

Lo sviluppo dello standard è ancora in corso.{mospagebreak title=JS Come usarlo} JavaScript - Come usarlo.

Per inserire del codice JavaScript in una pagina HTML si usa il tag
<script>

ESEMPIO <html>
<body>
<script type="text/javascript">
document.write("Hello World!");
</script>
</body>Il codice di cui sopra produrrà questo output su una pagina HTML:

Hello World!

Spiegazione dell'esempio

Per inserire un JavaScript in una pagina HTM, viene usato il tag <script>.All'interno del tag <script> inseriamo il tipo di attributo per definire il linguaggio di scripting.

Quindi scriviamo <script type="text/javascript"> e chiudiamo lo script con il tag </script>:<html>
<body>
<script type="text/javascript">
...
</script>
</body>
</html>

Il termine document.write è un comando standard JavaScript per la scrittura dell'output della pagina.

Inserendoil comando document.write tra i tag <script>e </script> , il browser riconosce che si tratta di un comando JavaScript e lo esegue. In questo caso il browser scriverà Hello World! all'interno della pagina: <html>
<body>
<script type="text/javascript">
document.write("Hello World!");
</script>
</body>
</html>

Note: Se non si inserisce il tag <script>, il browser avrebbe trattato il comando document.write("Hello World!") come normale testo,ed avrebbe scritto l'intera riga così com'è nella pagina HTML.Commenti HTML per la gestione dei Browsers

I browser che non supportano JavaScript mostrano come contenuto della pagina, testo puro e semplice.

Per evitare che accada questo, lo standard JavaScript ha introdotto un tag HTML adatto a prevenire questo, e nascondere il codice JavaScript (qualora non fosse supportato) nella pagina HTML. Basta aggiungere il tag di apertura di un commento <!-- prima della dichiarazione JavaScript, e il tag --> (fine del commento) alla fine della stessa dichiarazione JavaScript, come riportato di seguito:<html>
<body>
<script type="text/javascript">
<!--
document.write("Hello World!");
//-->
</script>
</body>
</html>

I due slashes alla fine del commento (//) è il simbolo del commento adottato proprio da JavaScriptl, necessario ad impedire che JavaScript esegua il tag --> {mospagebreak title=JS Dove}Dove si inserisce il codice JavaScript

JavaScript generalmente è eseguito immediatamente, mentre la pagina viene caricata nel browser. Ma questo non è sempre quello che vogliamo. A volte ci vuole per eseguire uno script quando il caricamento di una pagina, altre volte, quando un utente attiva un evento.

Scripts all'interno dell'head: Quando si inserisce uno script nell'head della pagina, questo viene caricato durante il caricamento della pagina stessa. Lo script viene eseguito nel momento in cui accade un certo evento o quando direttamente chiamato a seguito di un'azione specifica. <html>
<head>
<script type="text/javascript">
....
</script>
</head>

Scripts all'interno del body : Scripts viene eseguito al caricamento della pagina, generando un effetto immediato sull'output della pagina caricata.<html>
<head>

```
</head>
<body>
<script type="text/javascript">
....
</script>
</body>
```

Scripts sia nel body che nell'head della pagina: è possibile inserire un numero illimitato di script, sia nel body che nell'head della pagina HTML

Esempio<html>
<head>
<script type="text/javascript">
....
</script>
</head>
<body>
<script type="text/javascript">
....
</script>
</body>

Utilizzo di JavaScript esterno

A volte si ha la necessità di utilizzare lo stesso codice JavaScript su più pagine senza doverlo riscrivere ogni volta su ogni pagina.

Per semplificare il lavoro, si può scrivere il codice JavaScript su un file esterno. L'estensione per i files JavaScript è .js

Nota: In tal caso il file .js non contiene il tag <script>, ma solo i comandi che generelamente scriveremmo all'iterno di tale tag se dovessimo inserire lo script direttamente nella pagina HTML.

Per usare uno script esterno alla pagina, basta inserire l'attributo "src" all'interno del tag <script> che richiami il file .js e il suo percorso esatto, come segue:<html>
<head>
<script type="text/javascript" src="xxx.js"></script>
</head>
<body>
</body>
</html>

Nota: Ricorda di inserire lo script che richiama il file .js dove normalmente questo dovrebbe essere inserito (all'interno del body o dell'head) !{mospagebreak title=JS Comandi}Le Dichiarazioni JavaScript

JavaScript è fatto da una sequenza di dichiarazioni (comandi) che vengono eseguiti dal browser. JavaScript è Case Sensitive

Diversamente dall'HTML, JavaScript è case sensitive - pertanto occorre presetare molta attenzione a maiscole/minuscole quando si scrive una dichiarazione JavaScript, quando si crea o chiama una variabile, un oggetto o una funzione.Dichiarazioni JavaScript

Una dichiarazione JavaScript è un comando rivolto al browser. Un comando dice al browser quello che deve o non deve fare, come si deve comportare insomma.

Ad esempio nel seguente esempio un comando dice al browser di scrivere "Hello Dolly" all'interno della pagina:

```
document.write("Hello Dolly");
```

Ogni comando JavaScript termina con un punto e virgola.

In realtà il punto e virgola è opzionale (in accordo con gli standard JavaScript), si suppone che il browser sia in grado di interpretare la fine della linea come la fine della dichiarazione. Per questo si vedano in giro esempi di script JavaScript senza il punto e virgola alla fine. Ma è buona norma usare sempre e comunque il punto e virgola.

Nota: L'uso del punto e virgola permette di scrivere più dichiarazioni su una sola riga. Il codice JavaScript

In codice JavaScript è una sequenza di dichiarazioni (o comandi) JavaScript.

Ogni comando viene eseguito dal browser nell'ordine in cui i comandi sono stati scritti all'interno del codice.

Nel seguente esempio tramite una sequenza di tre comandi JavaScript, creiamo una header e due paragrafi successivi all'interno di una pagina web:<script type="text/javascript">
document.write("<h1>This is a header</h1>");
document.write(" This is a paragraph ");
document.write(" This is another paragraph ");
</script>

L'output è il seguente:This is a header

This is a paragraph

This is another paragraphBlocchi JavaScript

I comandi JavaScript possono essere raggruppati in blocchi.

Un blocco di comandi JavaScript inizia con l'apertura di una parentesi graffa {,e finisce con la chiusura della parentesi graffa }.

Lo scopo dei blocchi JavaScript è quello di fare in modo che tutti i comandi vengano eseguiti contemporaneamente.

In questo esempio verrà scritta una header e due paragrafi di una pagina web:

<script type="text/javascript">
{
document.write("<h1>This is a header</h1>");
document.write(" This is a paragraph ");
document.write(" This is another paragraph ");
}
</script>

L'esempio proposto è solo una dimostrazione, non troppo utile a dire il vero, per mostrare come si usa un blocco di codice JavaScript. Normalmente i blocchi vengono usati per script più complessi, ad esempio per raggruppare due o più comandi in una funzione o condizione (es. un blocco di comandi viene eseguito se si verificano determinate condizioni).

{mospagebreak title=JS Commenti}I Commenti JavaScript

I commenti nel codice JavaScript vengono usati per rendere il codice stesso più leggibile e comprensibile.Commenti JavaScripts su singola riga.

I commenti vengono inseriti per rendere il JavaScript più comprensibile, per spiegare il perchè di un certo comando ad esempio.

Una singola linea di commento inizia con il doppio slash //.

Esempio:<script type="text/javascript">
// This will write a header:
document.write("<h1>This is a header</h1>");
// This will write two paragraphs:
document.write(" This is a paragraph ");
document.write(" This is another paragraph ");
</script>

Commenti JavaScript Multi-Line

I commenti multi line iniziano con /* e terminano con */

Esempio:<script type="text/javascript">
/*

```
The code below will write
one header and two paragraphs
*/
document.write("<h1>This is a header</h1>");
document.write("<p>This is a paragraph</p>");
document.write("<p>This is another paragraph</p>");
</script>
```

Uso dei commenti per impedire l'esecuzione di un comando

In questo esempio, il commento è utilizzato per impedire l'esecuzione di una singola linea di codice:

```
<script type="text/javascript">
document.write("<h1>This is a header</h1>");
document.write("<p>This is a paragraph</p>");
//document.write("<p>This is another paragraph</p>");
</script>
```

In questo esempio, il commento è utilizzato per impedire l'esecuzione di più linee di comandi:

```
<script type="text/javascript">
/*
document.write("<h1>This is a header</h1>");
document.write("<p>This is a paragraph</p>");
document.write("<p>This is another paragraph</p>");
*/
</script>
```

Uso dei comment alla fine di una linea di codice

In questo esempio il commento viene aggiunto alla fine di una linea di codice:

```
<script type="text/javascript">
document.write("Hello"); // This will write "Hello"
document.write("Dolly"); // This will write "Dolly"
</script>
```

{mospagebreak title=JS Variabili}Le Variabili JavaScript

Le Variabili sono dei "contenitori" per memorizzare informationi.Ricordi l'algebra imparata a scuola?

Ricordi espressioni d'algebra del tipo x=5, y=6, z=x+y

Ad una lettera (per esempio x) diamo un certo valore (per esempio 5), e questa può essere per esempio usata per calcolare il valore di un'altra lettera (per esempio z) in espressioni del tipo z=x+y

Queste lettere sono chiamate variabili , e le variabili possono essere usate per conservare dei valori (x=5) o espressioni (z=x+y).

Variabili JavaScript

Come nell'algebra, le variabilli JavaScript vengono usate per memorizzare valori o espressioni.

Il nome della variabile può essere breve (es. x) oppure più lungo e descrittivo (es. varname).

Regole JavaScript per i nomi delle variabili:
- Il nome delle variabile è case-sensitive (quindi y e Y sono due variabili diverse)
- Il nome di una variabile deve inizare con una lettera o underscore _

Nota: Siccome JavaScript è case-sensitive, così pure i nomi delle variabili sono case-sensitive.Esempio

Il valore di una variabile può cambiare durante l'esecuzione di uno script. È possibile fare riferimento a una variabile con il suo nome per visualizzare o modificare il suo valore.

```
<script type="text/javascript">
var firstname;
firstname="Hege";
document.write(firstname);
document.write("<br />");
```

firstname="Tove";
document.write(firstname);
</script>

Lo script sopra, dichiara una variabile, assegna un valore ad esso, il valore viene visualizzato(Hege), quindi viene modificato , e visualizzato il nuovo valore(Tove).Dechiarare (Creare) una Variabile JavaScript

Quando si crea una variabili in JavaScript viene usato il termine "dichiarare".

Una variabile JavaScript viene "dichiarata" con la dichiarazione var :var x;
var carname;

Dopo averla dichiarata, la variabile è vuota (non le abbiamo ancora associato un valore).

Ad ogni modo è comunque possibile assegnare un valore alla variabile nel momento in cui la si dichiara, come nell'esempio sottostante:var x=5;
var carname="Volvo";

Dopo l'esecuzione del codice sopra la vairiabile x assume il valore 5, e la variabile carname il valore Volvo.

Nota: Quando si assegna un valore non numerico ad una variabile (es. sopra per Volvo), occorre scrivere il valore di testo delimitato dalle virgolette .Variabili non dichiarate

Se si associa un valore ad una variabile non precedentemente dichiarata, la variabile sarà automaticamente dichiarata.

Es. questo:x=5;
carname="Volvo";

Ha lo stesso valore di:var x=5;
var carname="Volvo";

"Re"dichiarazione di una variabile JavaScript

Se una variabile già dichiarata, viene dichiarata ancora successivamente, il valore associato alla variabile viene conservato.

Esempio:var x=5;
var x;

Dopo l'esecuzione del comando precedente il valore della variabile x è ancora 5.

Operazioni aritmetiche con JavaScript

Come con l'algebra, è possibile fare operazioni aritmetiche con le variabili JavaScript y=x-5;
z=y+5;

Nel prossimo capitolo capiremo come fare :)

{mospagebreak title=JS Operatori}Operatori JavaScript Operatori artimetici JavaScript

Gli operatori aritmetici sono usati per eseguire operazioni matematiche fra le variabili e/o i valori di queste variabili.

Dato y=5, la tabella sottostante spiega le comuni operazioni aritmetiche: Operatori Descrizione Esempi Risultati + Addizione x=y+2 x=7 - Sottrazione x=y-2 x=3 * Moltiplicazione x=y*2 x=10 / Divisione x=y/2 x=2.5 % Resto(Modulo) x=y%2 x=1 ++ Incremento x=++y x=6 -- Decremento x=--y x=4

Operatori d'assegnamento JavaScript

Gli operatori di assegnamento sono usati per assegnare dei valori alle variabili JavaScript.

Supponiamo che x=10 e y=5, la tabella seguente mostra l'utilizzo degli operatori di assegnamento:Operatori Esempi

Uguale a
Risultato = x=y x=5 += x+=y x=x+y x=15 -= x-=y x=x-y x=5 *= x*=y x=x*y x=50 /= x/=y x=x/y x=2 %= x%=y x=x%y x=0

L'operatore + usato in una stringa

L'operatore + può anche essere utilizzato per sommare variabili, che siano valori o stringhe di testo.

Esempi:txt1="What a very";
txt2="nice day";
txt3=txt1+txt2;

Dopo aver eseguito lo script precedente , la variabile txt3 è data dall'unione delle due stringhe di testo (senza spazio)

"What a verynice day".

Per aggiungere uno spazio fra le due stringhe basta semplicemente inserire uno spazio in una delle due stringhe, come segue:txt1="What a very ";
txt2="nice day";
txt3=txt1+txt2;

O inserire uno spazio nell'espressione, come segue: txt1="What a very";
txt2="nice day";
txt3=txt1+" "+txt2;

In entrambi i casi il risultato sarà:

"What a very nice day"Sommare stringhe e numeri

Guarda gli esempi seguenti ed i risultati che ne derivano: x=5+5;
document.write(x);

x="5"+"5";
document.write(x);

x=5+"5";
document.write(x);

x="5"+5;
document.write(x);

RISULTATI:

10

55

55

55

La regola generale è questa:

" La somma di uno numero con una stringa di testo dà come risultato una stringa di testo".

{mospagebreak title=JS Operatori Logici
Operatori di Comparazione} Operatori Logici e di comparazioneOperatoro di Comparazione (o confronto)

Gli operatori di confronto sono usati per determinare uguaglianze o differenze fra le variabili o i valori delle variabili.

Ad esempio, assumiamo che x=5, la seguente tabella spiega chiaramenti come operano gli operatori di confronto: Operatori Descrizione Esempi == uguale a
x==8 è falsa === esattamente uguale a (value and type) x===5 è vera
x=="5" è falsa
!= diverso x!=8 è vera
> maggiore x>8 falsa < minore x<8 è vera
>= maggiore o uguale x>=8 falsa <= minore o uguale
x<=8 è vera
Come si usano:

Gli operatori di comparazione possono essere usati nelle espressioni condizionale, per confrantare due valori e far seguire un'azione a seconda del rislutato della comparazione, ad esempio:if (age<18) document.write("Too young");

Tratteremo nel prossimo capitolo le "espressioni condizionali", quindi... nessuna apprensione!Operatori logici

Gli operatori logici servono per creare espressioni booleane complesse. Il loro risultato è sempre del tipo vero/falso, ossia "1"se l'espressione logica è verificat, "0" se non lo è.

Assumiamo che x=6 and y=3, la tabella che segue illustra gli operatori logici ed alcuni esempi di utilizzo: Operatori Descrizione Esempi && and (x < 10 && y > 1) è vera || or (x==5 || y==5) è falsa
! not (negazione) !(x==y) è vera

Operatore Condizionale

JavaScript contiene anche un operatore condizionale che assegna un valore ad una variabile in base a una certa condizione.Sintassivariablename=(condition)?value1:value2 Esempio:
greeting=(visitor=="PRES")?"Dear President ":"Dear ";

In questa espressione se la variabile visitor è identica a PRES, allora la variabile greeting avrà come valore "Dear President", diversamente avrà come valore "Dear".

{mospagebreak title=JS if/else}JavaScript If...Else Espressioni condizionali

Le espressioni condizionali in JavaScript sono usate per eseguire delle azioni sulla base di determinate condizioni.Espressioni condizionali

Le espressioni condizionali vengono usate per eseguire azioni specifiche al verificarsi di un determinato evento o condizione.

JavaScript ha le seguenti epressioni condizionali:
- if - si usa questa dichiarazione, se si desidera eseguire il codice solo se una determinata condizione è vera
- if...else - si usa questa dichiarazione se si desidera eseguire il codice se la condizione è vera e un altro codice se la condizione è falsa
- if...else if....else - si usa questa dichiarazione si desidera selezionare uno dei tanti blocchi di codice da eseguire
- switch - si usa questa dichiarazione si desidera selezionare uno dei tanti blocchi di codice da eseguire

Esempi

Utilizzo di If - Esempio:

```
<script type="text/javascript">
 var d = new Date();
 var time = d.getHours();

 if (time < 10)
 {
 document.write("Good morning");
 }
 </script>
```

E' un esempio dell'utilizzo della condizione if ; ossia, se l'ora è inferiore a 10 l'output della pagina genererà il messaggio "Good morning"

Utilizzo di if/else - Esempio:

```
<script type="text/javascript">
 var d = new Date();
 var time = d.getHours();

 if (time < 10)
 {
 document.write("Good morning");
 }
 else
 {
 document.write("Good day");
 }
 </script>
```

Se è verificata la condizione time<10 allora la pagina darà il messaggio "Good Morning" , altrimenti genererà il messaggio "Good day".

Utilizzo di if ... else if ... else - Esempio:

```
<script type="text/javascript">
 var d = new Date();
 var time = d.getHours();
 if (time<10)
 {
 document.write("Good morning");
 }
 else if (time>=10 && time<16)
 {
 document.write("Good day");
 }
 else
 {
 document.write("Hello World!");
 }
 </script>
```

Random Link, con l'utilizzo delle espressioni condizionali:

```
<script type="text/javascript">
 var r=Math.random();
 if (r>0.5)
 {
 document.write("<a href='http://www.pagineinrete.it'>Pagine in Rete!</a>");
 }
 else
 {
 document.write("<a href='http://www.pagineinrete.it/index.php/Guide/Guida-Java-Script'>Guida Js</a>");
 }
 </script>
```

Questo esempio nella pagina viene mostrato un link , caricato in random col 50% di possibilità per ciascuno di essi. If

È necessario utilizzare la dichiarazione if se se si desidera eseguire il codice solo se una determinata condizione è vera.Sintassiif (condition)
{
code to be executed if condition is true
}

Nota: È necessario scrivere if in minuscolo. Se si scrive IF (in maiuscolo) viene generato un errore JavaScript!Esempio 1<script type="text/javascript">

```
//Write a "Good morning" greeting if
//the time is less than 10 var d=new Date();
var time=d.getHours();

if (time<10)
{
document.write("Good morning");
}
</script>
```

Esempio 2

```
<script type="text/javascript">
//Write "Lunch-time!" if the time is 11 var d=new Date();
var time=d.getHours();

if (time==11)
{
document.write("Lunch-time!");
}
</script>
```

Nota: Quando si confrontano le variabili che si deve sempre utilizzare due segni uguale (==)!If...else

Si usano per fare in modo che se una condizione è vera venga eseguito un certo blocco, in alternativa (se la condizione non è verificata) viene eseguito il secondo blocco di codice.Sintassiif (condizione)

```
{
codica da eseguire se la condizione è vera
}
else
{
codica da eseguire se la condizione è falsa}
```

Esempio

```
<script type="text/javascript">
//If the time is less than 10,
//you will get a "Good morning" greeting.
//Otherwise you will get a "Good day" greeting. var d = new Date();
var time = d.getHours();

if (time < 10)
{
document.write("Good morning!");
}
else
{
document.write("Good day!");
}
</script>
```

If...else if...else

È necessario usare if....else if...else se si desidera selezionare uno dei tanti blocchi di codice. Sintassi

```
if (condizione1)
{
codice da eseguire se la condizione1 è vera
}
else if (condizione2)
{
codice da eseguire se la condizione 2 è vera
}
else
{
codice da eseguire se è la condizione1 , nè la condizione 2 sono vere
}
```

Esempi

```
<script type="text/javascript">
```

```
var d = new Date()
var time = d.getHours()
if (time<10)
{
document.write("Good morning");
}
else if (time>10 && time<16)
{
document.write("Good day");
}
else
{
document.write("Hello World!");
}
</script>
```

{mospagebreak title=JS Switch}JavaScript Switch

Si usa la dichairazione switch per selezionare uno dei tanti blocchi di codice contenuto nell'espressione. Sintassiswitch(n)

```
{
case 1:
  execute code block 1
  break;
case 2:
  execute code block 2
  break;
default:
  code to be executed if n is
  different from case 1 and 2
}
```

Spiegazione: in primo luogo abbiamo una sola espressione n (spesso più di una variabile), che viene valutata una sola volta.. Questo valore viene quindi confrontato con ciascun caso. Se c'è una corrispondenza, il blocco di codice associato a questo caso viene eseguito. L'uso del break serve ad impedire l'esecuzione automatica del blocco di codice successivo.

Esempio

```
<script type="text/javascript">
//You will receive a different greeting based
//on what day it is. Note that Sunday=0,
//Monday=1, Tuesday=2, etc. var d=new Date();
theDay=d.getDay();
switch (theDay)
{
case 5:
  document.write("Finally Friday");
  break;
case 6:
  document.write("Super Saturday");
  break;
case 0:
  document.write("Sleepy Sunday");
  break;
default:
  document.write("I'm looking forward to this weekend!");
}
</script>
```

{mospagebreak title=Js Popup}JavaScript Popup

Alert Box

L'alert box è spesso usato per essere certi che l'utente prenda visione di un certo messaggio.

Quando appare un alert box infatti, l'utente deve cliccare su "OK" per procedere alla lettura della pagina.

Sintassi:alert("sometext");

Boxdi conferma

UN box di conferma è usato per fare in modo che l'utente accetti o verifichi qualcosa.

Infatti, quando appare un box di conferma, l'utente deve cliccare su "OK" o "Cancella" per procedere.

se l'utente clicca OK, allora il box restituisce il valore "true"(vero).

Se l'utente clicca su "Cancel", allora viene restituito il valore "false" (falso).

Sintassi:confirm("sometext");

Prompt Box

Il prompt box viene usato per esmpio quand si vuole che l'utente inserisca un certo valore per poter procedere alla visualizzazione della pagina.

Infatti in tal caso l'utente deve inserire un valore di input, quindi cliccare su "OK" o "Cancel" per procedere.

Se l'utente clicca su "OK" il box restituisce il valore di input . Se l'utente clicca su "Cancella" il box restituisce valore nullo.

Sintassi:prompt("sometext","defaultvalue");

{mospagebreak title=JS Funzioni}Funzioni JavaScript

Una funzione è un blocco di codice (riutilizzabile) che viene eseguito a seguito di un evento, o quando direttamente chiamata.

È possibile richiamare una funzione da qualsiasi punto della pagina web You may call a function from anywhere within the page, o anche da altre pagine (se la funzione è integrata in un altro. Js file).

Le funzioni JS possono essere inserite sia nell' <head> che nel <body> di una pagina. Tuttavia, per assicurarci che la funzione sia di letta / caricata dal browser prima di essere espressamente chiamata, è buona norma inserirla all'interno dell'<head> della pagina.

Esempio:
```
<html>
<head>
<script type="text/javascript">
function displaymessage()
{
alert("Hello World!");
}
</script>
</head><body>
<form>
<input type="button" value="Click me!"
onclick="displaymessage()" >
</form>
</body>
</html>
```

Nel suddetto esempio se la linea alert("Hello world!!") non fosse stata inserita all'interno di una funzione, sarebbe stata eseguita subito, non appena questa fosse stata letta/caricata. In tal caso invece, il messaggio Hello world non appare prima che l'utente abbia cliccato sul pulsante "Click me!".Come si definisce una Funzione

La sintassi per creare una funzione è la seguente:function fnomefunzione(var1,var2,...,varX)
{
codice
}var1, var2, etc sono le variabili o i valori (parametri) che vengono che vengono passati attraverso la funzione. Le parentesi graffe { e } determinano l'inizio e la fine della funzione.La dichiarazione della funzione si effettua tramite l'istruzione function. l'istruzione function richiede che siano definiti anche i parametri della funzione, ovvero quei valori che questa prenderà in carico come input.I parametri di una funzione seguono sempre il nome della funzione stessa

all'interno di una parentesi, separati l'uno dall'altro da una virgola.

Nota: Una funzione senza parametri deve comunque avere le parentesi tonde (e) dopo il nome della funzione:function nomefunzione()
{
codice
}

Nota : l'uso delle parentesi graffa
Le parentesi graffa nella fase di dichiarazione della funzione ne identificano il blocco delle istruzioni.Valore di ritorno di una funzione

La dichiarazione di ritorno di una funzione JavaScript è usato per specificare il valore che viene restituito dalla funzione.Esempio

La funzione sottostante deve restiruire il prodotto di due numeri (a e b):function prod(a,b)
{
x=a*b;
return x;
}

Quando si chiama la funzione di cui sopra, si deve passare da due parametri:product=prod(2,3);

Il valore restituito dalla funzione prod () è 6, e sarà memorizzato nella variabile chiamata product.Durata di una Variabile JavaScript

Quando si dichiara una variabile tramite una funzione, la variabile è valida solo all'interno della funzione. Quando la funzione cessa di esistere, la variabile viene distrutta automaticamente. Per le varibili all'interno di una funzione si parla di "Variabili locali"

È possibile avere variabili con nomi uguali in funzioni differenti, questo perchè ogni variabile viene riconosciuta (ed è quindi valida) solo all'interno della funzione in cui viene dichiarata.

Ci sono poi le cosiddette "variabili globali", che hanno validità in tutta la pagina.

La durata di queste variabili si avvia quando queste vengono dichiarate, e termina quando la pagina viene chiusa.

{mospagebreak title=JS For Loop} JavaScript Loop For

Il loop for esegue una serie di istruzioni fino a che non è stato raggiunto il limite indicato da una condizione. I loops JavaScript

Spesso quando si scrive del codice capita di volere eseguire più volte lo stesso script fino al raggiungimento di una certa condizione.

Si parla in questo caso di "iterazioni" o "cicli iterativi"

In JavaScript esistono tre(sostanzialmente due) comandi di iterazione :
- for -tramite il quale viene eseguito un certo blocco di codice un numero determinato di volte.
- while - ripete un certo blocco di codice in loop finche una certa condizione è vera.

For

Il costrutto for per i cicli iterativi è usato quando si sa in anticipo quante volte il ciclo deve ripetersi.

Sintassifor (var=startvalue;var<=endvalue;var=var+increment)
{
  code to be executed
}

Esempio

Spiegazione: Nell'esempio precedente il loop inzia quando i=0. Il loop continua fino a che i è minore o uguale a 10

i aumenterà di 1 ogni volta che viene eseguito il ciclo.

Nota: L'incremento può anche essere negativo.<html>

```
<body>
<script type="text/javascript">
var i=0;
for (i=0;i<=10;i++)
{
document.write("The number is " + i);
document.write("<br />");
}
</script>
</body>
</html>
```

Risultato

The number is 0
The number is 1
The number is 2
The number is 3
The number is 4
The number is 5
The number is 6
The number is 7
The number is 8
The number is 9
The number is 10

{mospagebreak title=JS While loop} JavaScript While Loop

Il costrutto while viene usato quando si desidera eseguire e ripetere un certo ciclo al verificarsi di una certa condizione. Finchè la condizione è vera il ciclo viene eseguito ripetutamente.

Esempio:<html>

```
<body>
<script type="text/javascript">
var i=0;
while (i<=10)
{
document.write("Numero " + i);
document.write("<br />");
i=i+1;
}
</script>
</body>
</html>
```

Risultato:

Numero 0
Numero 1
Numero 2
Numero 3
Numero 4
Numero 5
Numero 6
Numero 7
Numero 8
Numero 9
Numero 10

Spiegazione: L'esempio sopra definisce un loop che inizia con i=0. Il loop continua fino a che i è minore o uguale a 10. i aumenta di uno dopo ogni cliclo di loop.Il do...while Loop

Il do...while loop è una variante del while loop. Con il do...while il loop viene eseguito sempre una volta, e ripetuto fintanto che la condizione specificata è vera. Se la condizione specificata non è vera, il cliclo viene comunque eseguito una volta, questo perchè la verifica della condizione viene fatta dopo l'esecuzione del primo ciclo

Esempio:
```
<html>
<body>
<script type="text/javascript">
var i=0;
do
{
document.write("Numero " + i);
document.write("<br />");
i=i+1;
}
while (i<0);
</script>
</body>
</html>
```

Risultato:Numero 0

{mospagebreak title=JS break loop}JavaScript Break e Continue

Break e continue sono due speciali costrutti usati nei loop.Break

Il comando break interrompe il ciclo e continua eseguendo il blocco di codice che si trova subito dopo il loop.

Esempio:
```
<html>
<body>
<script type="text/javascript">
var i=0;
for (i=0;i<=10;i++)
{
if (i==3)
{
break;
}
document.write("Numero " + i);
document.write("<br />");
}
</script>
</body>
</html>
```

Risultato:Numero 0
Numero 1
Numero 2

Continue

Il comando continue interrompe il loop corrente e continua con il valore successivo.

Esempio:
```
<html>
<body>
<script type="text/javascript">
var i=0
for (i=0;i<=10;i++)
{
if (i==3)
{
```

```
continue;
}
document.write("Numero " + i);
document.write("<br />");
}
</script>
</body>
</html>
```

Risultato:Numero 0
Numero 1
Numero 2
Numero 4
Numero 5
Numero 6
Numero 7
Numero 8
Numero 9
Numero 10

{mospagebreak title=Js For..in}JavaScript For...In

Il costrutto for...in (da non confondere con il foreach di altri linguaggi) è usato per creare un loop attraverso gli elementi di un array o attraverso le proprietà di un oggetto.

Il codice contenuto in un for ... in loop viene eseguito una sola volta per ogni elemento/proprietà.

Sintassi:

```
for (variable in oggetto)
{
    //istruzioni
}
```

L'argomento della variabile può essere un nome di variabili, gli elementi di un array, o la proprietà din un certo oggetto della pagina.Esempio:

Utilizzo di for...in tramite un array(attraverso questo loop vengono passati in rassegna li elementi dell'array):

```
<html>
<body>   <script type="text/javascript">
var x;
var mycars = new Array();
mycars[0] = "Saab";
mycars[1] = "Volvo";
mycars[2] = "BMW";

for (x in mycars)
{
document.write(mycars[x] + "<br />");
}
</script>   </body>
</html>
```

Risultato:Saab
Volvo
BMW

{mospagebreak title=JS Eventi}Eventi JavaScript

Con JavaScript si ha la possibilità di creare pagine dinamiche. Gli eventi sono quelle azioni che possono essere rilevate da JavaScript.

Ogni elemento di una pagina Web ha alcuni eventi che possono attivare funzioni JavaScript. Ad esempio, possiamo associare all'evento onClick di un pulsante l'attivazione di una certa funzione che verrà eseguita quando un utente fa clic sul pulsante. Gli eventi vengono definiti nei tag HTML di una pagina web.

Esempi di eventi:
- Un click del mouse
- Il caricamento di una pagina web o di un'immagine
- Il passaggio del mouse su uan certa area dela pagina
- La selezione di una casella all'interno di un form
- L'invio di un form HTML
- Un tasto

Nota: Gli eventi sono usati normalmente in associazione alle funzioni, e le funzioni non verrà eseguita prima che l'evento si verifichi.Alcuni Eventi

Onload e onUnload

Gli eventi onload e onUnload events vengono attivati quando l'utente entra o esce da una pagina, rispettivamente.

L'evento onload viene spesso usato per effettuare controlli sul tipo/versione di browser usato da un utente, di modo che possa essere caricata la pagina appropriata al browser in uso.

Entrambi gli eventi (onload e onUnload) vengono spesso usati per impostare i cookies quando un utente entra o esce da una pagina web. Ad esempio, si potrebbe avere una finestra popup che chiede il nome utente al suo primo arrivo alla tua pagina. Il nome viene memorizzato in un cookie. La prossima volta che il visitatore arriva alla tua pagina, si potrebbe avere un altro popup su cui è scritto un messaggio del tipo: " Benvenuto ' Carlo' ".

OnFocus, onBlur e onChange

Gli eventi onFocus, onBlur e onChange sono spesso usati in combinazione con la validazione dei campi di un form.

Sotto c'è un esempio di come si usa un evento onChange. La funzione checkEmail() viene chiamata ogni volta che un utente cambia il contenuto del del campo:<input type="text" size="30"
id="email" onchange="checkEmail()">

OnSubmit

L'evento onSubmit è usato per la validazione di tutti i campi di un form prima dell'invio.

Sotto c'è un esempio di come si usa l'evento onSubmit. La funzione checkForm() function verrà chiamata quando l'utente clicca sul pulsante d'invio del form. Sei i valori dei campi non sono ammessi, l'invio viene impedito. La funzione checkForm() restituisce i valori vero o falso. Se restituisce il valore true (vero) il form viene inviato, diversamente (se restituisce il valore falso), l'invio viene cancellato. <form method="post" action="xxx.htm"
onsubmit="return checkForm()">

OnMouseOver e onMouseOut

Gli eventi onMouseOver e onMouseOut sono spesso usati per creare pulsanti "animati".

Sotto c'è un esempio di un evento onMouseOver. Un messaggio di avviso appare ogni volta che il mouse passa sul pulsante: <a href="http://www.pagineinrete.it"
onmouseover="alert('Ci si collega al sito pagine in rete');return false">
<img src="button.gif" width="100" height="30">

{mospagebreak title=JS Try...Catch}JavaScript Try...Catch

Il costrutto try..catch è un utile costrutto per la gestione degli errori in JavaScript.

In realtà JavaScript usa due metodi per la gestione degli errori:
- Il costrutto try...catch
- L'evento onerror.

Costrutto Try...Catch

Il costrutto try...catch viene usato per testare un blocco di codice e vedere se ci sono degli errori, dove try contiene il codice da eseguire, e catch provvede alla gestione dell'errore.Syntaxtry
{

```
//Istruzioni}
catch(err)
{
//Gestione errore}
```

Nota: try...catch deve essere scritto usando lettere minuscole. Se si usano le lettere maiuscole, viene generato un errore JavaScript!

Esempio 1:

Sotto viene riportato un esempio in cui c'è un codice che dovrebbe dare all'utente un messaggio di benvenuto "Benevenuto ospite", quando l'utente clicca sul pulsante. Nel caso specifico, c'è un errore di battitura nella funzione message(), in particolare alert(), è stato scritto con ->adddalert(). Questo genera un errore JavaScript<html>

```
<head>
<script type="text/javascript">
function message()
{
adddlert("Benvenuto Ospite!");
}
</script>
</head>

<body>
<input type="button" value="View message" onclick="message()" />
</body>

</html>
```

Di seguito la stessa istruzione viene controllata tramite il costrutto try...catch:<html>

```
<head>
<script type="text/javascript">
var txt=""
function message()
{
try
  {
  adddlert("Benvenuto Ospite!");
  }
catch(err)
  {
  txt="Errore\n\n";
  txt+="Clicca OK per vedere la pagina,\n";
  txt+="o Cancel per tornare alla home page.\n\n";
  if(!confirm(txt))
    {
    document.location.href="http://www.pagineinrete.it/";
    }
  }
}
</script>
</head> <body>
<input type="button" value="View message" onclick="message()" />
</body> </html>
```

{mospagebreak title=JS Throw}

JavaScript Istruzione ThrowIstruzione Throw

L'istruzione throw permette di creare un'eccezione. Un'eccezione è un segnale che indica che si è verificata una condizione eccezionale o di errore.Usata assieme all'istruzione try...catch, permette di controllare il flusso del programma e generare messaggi di errore. Sintassithrow(exception)

L'eccezione può essere una stringa, un numero intero, una espressione Booleana o un oggetto.

Nota l'istruzione throw è scritto in minuscolo. Se si scrive in maiuscolo viene generato un errore JavaScriptEsempio 1

Nell'esempio sottostante viene determinato il valore di una variabile x. Se il valore di di x è maggiore di 10 o minore di 0 viene passata l'istruzione throw che genera l'appropriato messaggio di errore, attraverso l'argomento di chatch. <html>
<body>
<script type="text/javascript">
var x=prompt("Inserisci un numero compreso fra 0 e 10:","");
try
{
if(x>10)
throw "Err1";
else if(x<0)
throw "Err2";
}
catch(er)
{
if(er=="Err1")
alert("Errore! Valore troppo alto");
if(er == "Err2")
alert("Errore! Valore troppo basso");
}
</script>
</body>
</html>

{mospagebreak title=JS Evento onerror }JavaScript evento onerror Evento Onerror

Ho appena spiegato come usare l'istruzione try..catch (assieme a "throw") per creare dei messaggi di errore in una pagina web. L'evento onerror può essere usato per gli stessi scopi.

L'evento onerror viene scatenato ogni volta che si ha un errore di script nella pagina.

Per usare l'evento onerror, è necessario creare una funzione che gestisca l'errore. Poi si chiama la funzione con il gestore di eventi onerror. Il gestore di eventi viene chiamato con tre argomenti: msg (messaggio di errore), url (l'url della pagina che ha generato l'errore) e line (la linea di codice dove l'errore si è verificato). Sintassionerror=handleErr function handleErr(msg,url,l)
{
//Handle the error here
return true or false
}

Il valore restituito da onerror determina se il browser visualizza un messaggio di errore. Se restituisce false, il browser visualizza il messaggio di errore standard nella console JavaScript. Se restituisce true, il browser non visualizza il messaggio di errore standard.Esempio <html>
<head>
<script type="text/javascript">
onerror=handleErr;
var txt=""; function handleErr(msg,url,l)
{
txt="There was an error on this page.\n\n";
txt+="Error: " + msg + "\n";
txt+="URL: " + url + "\n";
txt+="Line: " + l + "\n\n";
txt+="Clicca OK per continuare.\n\n";
alert(txt);
return true;
} function message()
{
adddlert("Benvenuto ospite!");
}
</script>
</head> <body>
<input type="button" value="Guarda il messaggio" onclick="message()" />
</body> </html>

{mospagebreak title=JS Caratteri Speciali
 Linee Guida}

Caratteri Speciali JavaScript

In JavaScript è possibile aggiungere caratteri speciali in una stringa di testo utilizzando il backslash

In JavaScript, una stringa è racchiusa tra le virgolette o gli apici. Se scrivessimo cosìvar txt="Ci chiamano i "Cibernautici"";
document.write(txt);

la stringa suddetta verrebbe interrotta a Ci chiamano i

Per risolvere questo problema, ogni qual volta le virgolette fanno parte della stringa di testo è necessario mettere un backslash prima di ogni (") citazione .var txt="Ci chiamano i \"Cibernautici\" .";
document.write(txt);

Altro esempio:document.write ("Azienda Rossi \& figli!");

Che produce il seguente output Azienda Rossi & figli!

La tabella seguente elenca gli altri caratteri speciali (ovvero sequenze) che possono essere aggiunti ad una stringa di testo per formattare il testo. Sequenza
Interpretazione di Output \' Virgolette singole
\" Virgolette doppie \& ampersand \\ backslash (per scrivere il cararattere backslash altrimenti interpretato come sequenza di escape) \n Nuova riga (usata per andare a capo)
\r Ritorno carrello (meno usato, in genere è accoppiato con \n) \t tab orizzontale
\b backspace \f avanzamento pagina

Esempio:

document.write("questa è una riga \n e questa è un'altra riga");
JavaScript Linee Guida

Cose da sapere quando si scrive in JavaScript! JavaScript è Case Sensitive

Una funzione chiamata "myfunction" non è la stessa cosa di "myFunction" , allo stesso modo una variabile chiamata "myVar" è diversa da "myvar".

JavaScript è dunque case sensitive - quindi è importante prestare bene attenzione all'uso di maiuscole/ minuscole usate quando si crea una variabile, una funzione o un oggetto.Spazi

JavaScript ignora gli spazi. Gli spazi vengono aggiunti per rendere lo sript più leggibile, ma di fatto sono ignorati da JavaScript.

Le seguenti linee sono equivalenti:name="Marco";
name = "Marco";

Interruzione riga di codice.

Si può spezzare una riga di codice solo all'interno di una stringa, nel modo seguentedocument.write("Hello \
World!");

:

Se invece cerchiamo di spezzare una riga di codice al di fuori della stringa come scritto sotto, si ottiene un errore e l'interruzione dello script:document.write \
("Hello World!");