JAVASCRIPT

Cristian Carrino

ENGIM PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

La storia di Javascript

Questo linguaggio fu sviluppato originariamente da Brendan Eich, sviluppatore al soldo della Netscape Communications, con il nome di Mocha ed in seguito di LiveScript.
Il nome attuale è successivo ed è frutto di una sorta di "avvicinamento" della sintassi del linguaggio a quella di Java, da qui il nome Javascript.
Quest'ultima definizione, tuttavia, ha suscitato non poca confusione tra gli utenti.

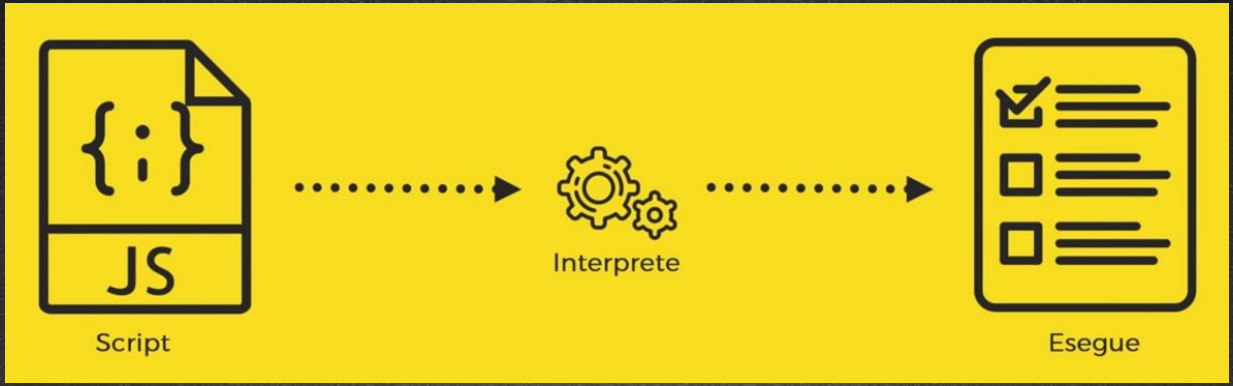
! Javascript, in realtà, non ha nulla a che spartire con Java se non qualche (piccola) somiglianza in alcune sintassi.

Javascript (spesso abbreviato come JS) è il linguaggio di scripting più diffuso sul Web.

Cosa vuol dire linguaggio di scripting?
Se il linguaggio di programmazione che abbiamo utilizzato, da solo non basta per funzionare ma ha bisogno di un interprete, allora si parla in realtà di linguaggio di scripting.
E' una differenza sottile e spesso non conosciuta.

E il codice scritto con un linguaggio di scripting si chiama script (e non programma).

L’interprete legge, verifica ed esegue lo script (cioè il codice)



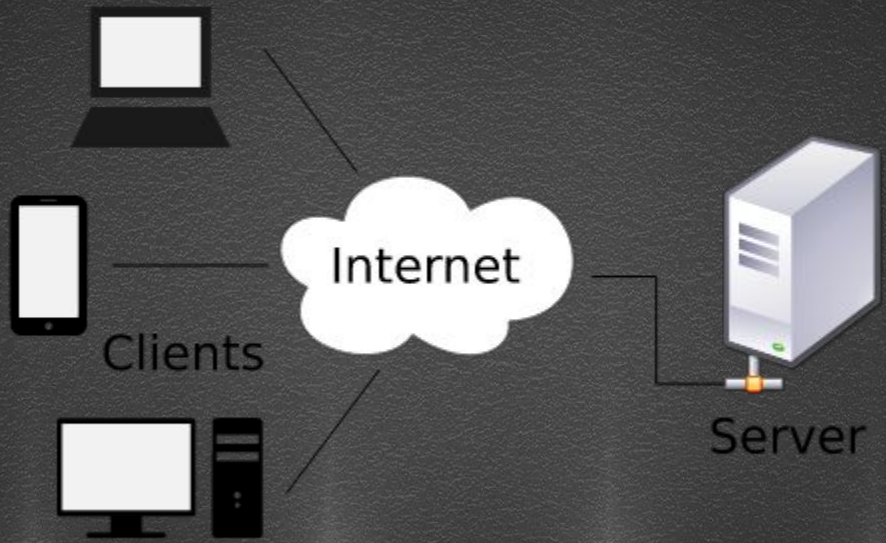
L’interprete che fa funzionare il codice javascript è il browser (Chrome, Internet Explorer, Edge, Firefox)

Per cosa viene utilizzato Javascript?

Javascript è utilizzato per “animare” le pagine web. Esempi

Inoltre, i linguaggi per il web si dividono in

- linguaggi lato client (javascript)
- linguaggi lato server (php)



Client

Computer, Smartphone, tablet e qualsiasi altro dispositivo con connessione a internet che fa una richiesta ad un server

Server

Un computer che sta dietro internet, che riceve la richiesta del client e gli ritorna una risposta (ad esempio una pagina web)

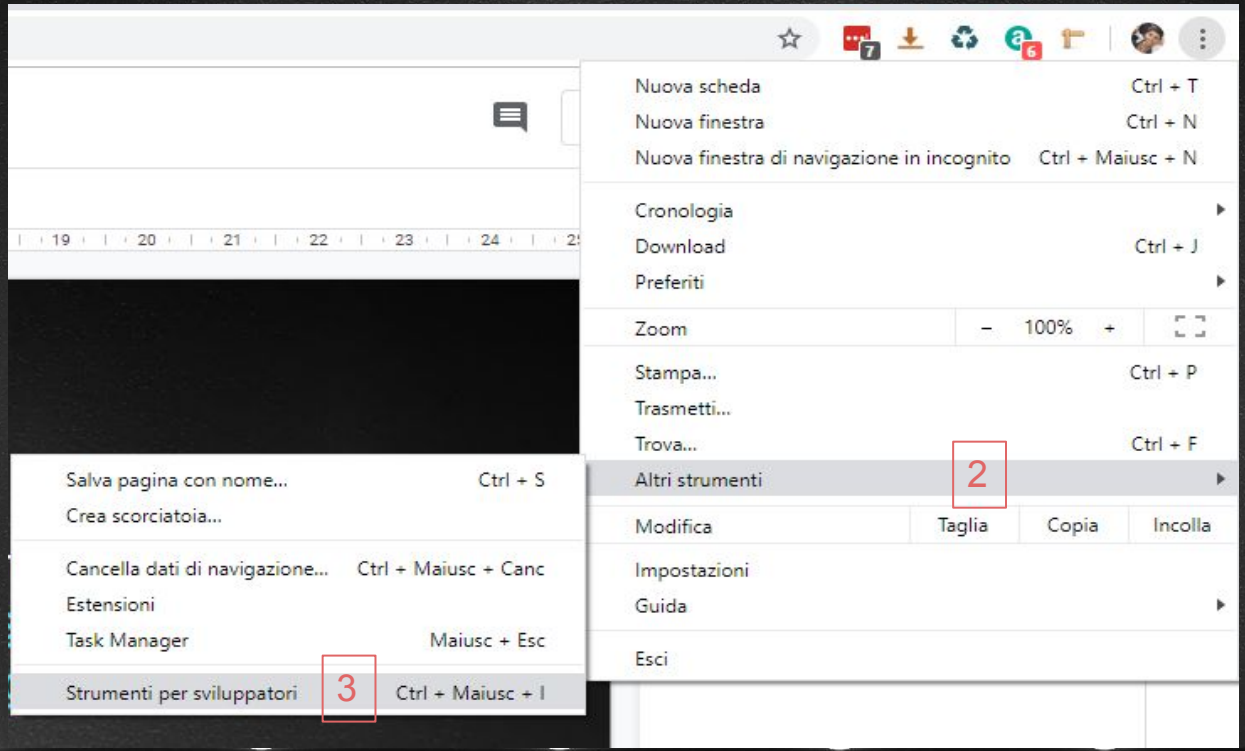
In realtà oggi...

LA CONSOLE DEL BROWSER

Cristian Carrino

ENGIM
PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

La console del browser è uno strumento FONDAMENTALE per uno sviluppatore



Cristian Carrino

LE VARIABILI

Cristian Carrino

ENGIM
PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

In un qualsiasi linguaggio di programmazione, le variabili corrispondono ai dati necessari al programma.

Una variabile, per poter essere utilizzata, deve essere prima dichiarata. Una volta dichiarata, posso utilizzarla (ad esempio per assegnarle un valore).

```
> var a;
  a = 4;
  console.log(a);
  4
```

Se provassimo ad utilizzare una variabile prima di averla dichiarata, otterremmo l'errore di variabile non definita.

```
> console.log(a);
▶ Uncaught ReferenceError: a is not defined
```

Una variabile può essere dichiarata ed inizializzata nello stesso momento (inizializzazione in linea).

```
> var a = 4;
```

Riguardo al nome della variabile

- Javascript è case-sensitive, questo significa che una variabile chiamata "miavariabile" sarà differente da un'altra variabile chiamata "MiaVariabile";
- non deve coincidere con una delle parole chiave del linguaggio;
- non può iniziare con un numero;
- non può contenere uno spazio;
- non può contenere caratteri speciali come, ad esempio, il punto (.) ed il trattino (-).

Tipi di dato

- stringa di testo
- valore numerico (intero o decimale)
- valore booleano (true/false)
- valore NULL

In una stringa, il testo deve essere racchiuso tra apici singoli ‘ oppure apici doppi “

```
> var stringa = 'Ciao a tutti';
  var stringa2 = "Ciao a tutti";
```

Se nelle mia stringa volessi mettere un apostrofo?

```
> var stringa = "La borsa è sotto l'albero"; // uso gli apici doppi
  var stringa = 'La borsa è sotto l\'albero'; // faccio escaping dell'apostrofo
```

Se nelle mia stringa volessi mettere i doppi apici?

```
> var stringa = 'Carlo ha detto "Ciao" a Luca'; // uso gli apici singoli
  var stringa = "Carlo ha detto \"Ciao\" a Luca"; // faccio escaping dei doppi apici
```

LE COSTANTI

Cristian Carrino

ENGIM
PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

Le costanti sono quei dati il cui valore, una volta assegnato, non può più essere modificato.

```
> const A = 12;
```

Infatti se in seguito provo a cambiarne il valore, mi ritorna errore di assegnazione ad una costante.

```
> A = 4;
✕ ▶Uncaught TypeError: Assignment to constant variable.
```

NB: per le costanti l'inizializzazione in linea è obbligatoria. Altrimenti mi ritorna errore di mancata inizializzazione nella dichiarazione della costante.

```
> const B;
Uncaught SyntaxError: Missing initializer in const declaration
```

NB: per convenzione tra programmatori, i nomi delle costanti si scrivono tutti in MAIUSCOLO. Così chi la ritrova nel codice intuisce subito che si tratti di una costante.

```
> const PIPPO = 8;
```

LO STRICT MODE

Cristian Carrino

ENGIM
PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

In realtà Javascript permetterebbe di inizializzare una variabile senza mettere la parola chiave var davanti.
In altre parole è valido scrivere:

```
> nuovaVariabile = 10;
```

Questo non sempre utile, anzi, spesso può portare confusione e può indurre lo sviluppatore a commettere degli errori.
E sarà più difficile poi, trovare l'errore facendo debug.

Per questo dalla versione 1.8.5 di Javascript, è stato introdotto il cosiddetto strict mode.

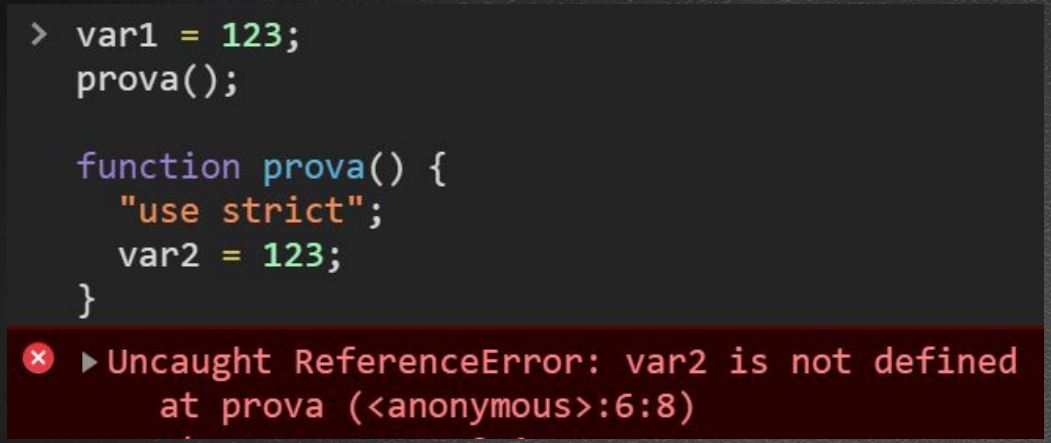
Lo strict mode si attiva semplicemente scrivendo la stringa “use strict”; prima del nostro codice ed introdurrebbe una sorta di “tolleranza zero” su alcune operazioni.

```
> "use strict";
  /*
  codice javascript
  */
```

Attivando lo strict mode saremo obbligati, ad esempio, a inserire la parola chiave var nella dichiarazione di una nuova variabile. Altrimenti ci ritornerà un errore.

```
> "use strict";
  nuovaVariabile = 10;
✕ ▶Uncaught ReferenceError: nuovaVariabile is not defined
      at <anonymous>:2:16
```

“use strict”; è valido solo nello scope in cui viene dichiarato.
Ad esempio:



“use strict”; è valido solo all’interno della funzione prova().
Infatti ho potuto dichiarare var1 senza la parola chiave var.

I COMMENTI

Cristian Carrino

ENGIM
PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

E’ buona norma commentare il nostro codice.
Questo può essere utile in due casi:

- è utilissimo per chiunque, un giorno, andrà a mettere mano al nostro codice perchè capirà più facilmente cosa quella porzione di codice faccia
- è utilissimo a noi stessi quando dovremo andare a mettere mano ad un codice che avevamo scritto molto tempo prima perché non ci ricorderemo minimamente cosa faccia

Ci sono due modi di scrivere commenti:

- commento in linea (// commento)
- blocco di commento (/* commento */)

```
> // questo è un commento in linea

  /*
  Questo è un blocco di commento.
  In questo modo posso scrivere più righe di commento.
  Anche molte righe se è necessario.
  */
```

OPERATORI MATEMATICI

Cristian Carrino

ENGIM PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

I 5 operatori matematici di base sono +, -, *, /, %

```
> const A = 12;
  const B = 4;
  var result;

  result = A + B; // somma
  result = A - B; // sottrazione
  result = A * B; // moltiplicazione
  result = A / B; // divisione
  result = A % B; // resto della divisione
```

Esistono altri 5 operatori matematici che servono ad abbreviare alcune formule e sono +=, -=, *=, /=, %=.

```
> a += 3;     // a = a + 3;
  a -= 3;     // a = a - 3;
  a *= 3;     // a = a * 3;
  a /= 3;     // a = a / 3;
  a %= 3;     // a = a % 3;
```

Esistono altri 2 operatori (++, --)

```
> a++;    // operatore di incremento, sarebbe a = a + 1;
  a--;    // operatore di decremento, sarebbe a = a - 1;
```

OPERATORE DI CONCATENAZIONE

Cristian Carrino

ENGIM PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

Il simbolo + oltre ad essere l'operatore della somma, serve anche per concatenare le stringhe.

```
> const TESTO = 'Il mio nome è: ';
  const NOME = 'Cristian ';
  const COGNOME = 'Carrino';
  var result = TESTO + NOME + COGNOME + '!!!';

  console.log(result);
  Il mio nome è: Cristian Carrino!!!
```

Quando uso il simbolo + tra un numero e una stringa, il numero viene prima trasformato in stringa e poi le due stringhe vengono concatenate

```
> const A = 46; // è un numero
  const B = 'Ciao';

  console.log(A + B);
```

```
46Ciao
```

Per trasformare una stringa (che rappresenta un numero) nel suo rispettivo numero, si usa la funzione parseInt()

```
> var stringa = '46'; // è una stringa
  var numero = parseInt(stringa); // è diventato un numero

  console.log(stringa, numero);
```

```
  46 46
```

Per trasformare un numero in una stringa, come abbiamo visto, basta concatenarlo con una stringa. Quindi...

```
> var numero = 56;  // è un numero
  var stringa = numero + '';  // è una stringa

  console.log(numero, stringa);
```

```
56 "56"
```

OPERATORI DI CONFRONTO

Cristian Carrino

ENGIM PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

Gli operatori di confronto sono degli operatori che confrontano il contenuto di due variabili e ritornano true o false.

I 6 operatori di confronto sono

- == uguale
- != diverso
- < minore
- <= minore o uguale
- > maggiore
- >= maggiore o uguale

Esempio

```
> const A = 12;
  const B = 4;
  var result;

  result = (A == B); // result sarà false
  result = (A != B); // result sarà true
  result = (A < B); // result sarà false
  result = (A <= B); // result sarà false
  result = (A > B); // result sarà true
  result = (A >= B); // result sarà true
```

NB: le parentesi tonde in questo caso non sono obbligatorie ma aiutano a comprendere meglio il codice

OPERATORI LOGICI

Cristian Carrino

ENGIM PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

Gli operatori logici fanno il confronto tra due variabili booleane (cioè che sono true o false) e restituiscono un risultato booleano (true o false)

I 3 operatori logici sono

- && AND (basta un false e il risultato sarà false)
- || OR (basta un true e il risultato sarà true)
- ! NOT (diventa il contrario di prima)

Esempio

```
> const A = true;
  const B = true;
  const C = false;
  var result;

  result = (A && B && C);    // result sarà false
  result = (A || B || C);    // result sarà true
  result = !A;               // result sarà false
  result = !C;               // result sarà true
```

Esempio più complicato

```
> var a = 10;
  var b = 3;
  var c = 9;
  var d = 24;
  var result;

  result = (a == b && c < d);  // result sarà false
  result = (a != b && c < d);  // result sarà true
  result = (a > b || c == d);  // result sarà true
  result = (!(a > b));         // result sarà false
```