JAVASCRIPT

Cristian Carrino

ENGIM PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

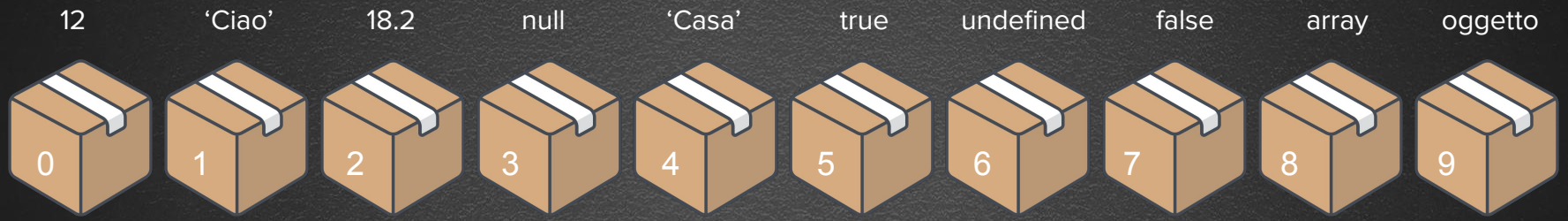
GLI ARRAY

(I VETTORI)

Cristian Carrino

ENGIM
PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

Gli array sono delle variabili particolari che possono contenere più dati.
Sono come dei contenitori numerati da 0 in poi, al cui interno posso mettere quello che voglio.



Per definire un array, si usa la notazione con le parentesi quadre [ ]
I valori al suo interno dovranno essere separati da virgola ,

```
> var numeri = [3, 9, 8, 17, 10000, 4, 2, 32];
```

```
> var elementi = [12, 'Ciao', 18.2, null, 'Casa', true, undefined, false, [1, 2], { total: 2 }];
```

Per definire un array vale ancora la sintassi new Array( ) che però ormai è in disuso.

```
> var numeri = new Array(3, 9, 8, 17, 10000, 4, 2, 32);
```

Gli array in realtà sono degli oggetti e come tali hanno delle proprietà e dei metodi:

- .length (esempio: numeri.length)
  (proprietà, mi restituisce la lunghezza dell'array)
- .push(...) (esempio: numeri.push(7))
  (metodo, aggiunge l'elemento in coda)
- .pop() (esempio numeri.pop())
  (metodo, toglie l'elemento dalla coda e lo ritorna)
- .unshift(...) (esempio numeri.unshift(8))
  (metodo, aggiunge l'elemento in testa)
- .shift() (esempio numeri.shift())
  (metodo, toglie l'elemento dalla testa e lo ritorna)

Esistono molti altri metodi:

- .sort (esempio: numeri.sort())
  (ordina gli elementi dell'array in ordine decrescente)
- .reverse() (esempio: numeri.reverse())
  (gira l'array al contrario)
- .concat([...]) (esempio numeri.concat(numeri2))
  (concatena il secondo array al primo in un unico array finale)
- .join(...) (esempio numeri.join(' - '))
  (unisce tutti gli elementi dell'array in una stringa sola
  separandoli attraverso la stringa passata come argomento)

Per prendere uno degli elementi dell'array, si usa la sua posizione (chiamata indice).

```
> var numeri = [3, 9, 8, 17, 10000, 4, 2, 32];
  console.log(numeri[0]);
  console.log(numeri[4]);
  3
  10000
```

Attraverso gli indici si può anche modificare (o aggiungere) un elemento dell'array

```
> numeri[9] = 20;
```

Per ciclare un array (cioè scansionare tutto l'array) si utilizzano un ciclo for e la proprietà length dell'array.

```
> for (var i = 0; i < numeri.length; i++) {
      console.log(numeri[i]);
  }
  3
  9
  8
  17
  10000
  4
  2
  32
  undefined
  20
```

GLI OGGETTI

Cristian Carrino

ENGIM
PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

Gli oggetti assomigliano agli array, ma per accedere ad un elemento non si utilizza il suo indice ma una label associata ad esso (chiamata chiave).

Quindi:

- gli array sono posizionali (cioè si accede agli elementi tramite la loro posizione)
- gli oggetti sono coppie chiave-valore.

Gli elementi di un oggetto sono chiamate proprietà (se sono delle variabili) o metodi (se sono delle funzioni).

Esempio

```
> var person = {
      firstname: 'Cristian',
      lastname: 'Carrino',
      age: 34,
      children: null
  };

  console.log(person);
```

```
▶ {firstname: "Cristian", lastname: "Carrino", age: 34, children: null}
```

NB: le chiavi possono anche essere messe tra virgolette. Questo perché, in questo modo, si possono mettere anche degli spazi all'interno della chiave

```
> var person = {
      'first name': 'Cristian',
      'last name': 'Carrino',
      'age': 34,
      'children': null
  };

  console.log(person);
```

```
▶ {first name: "Cristian", last name: "Carrino", age: 34, children: null}
```

Per accedere ad una proprietà dell'oggetto si può usare la notazione con il punto . o quella con le parentesi quadre [ ].

```
> person.age;
<· 34
> person['age'];
<· 34
> person["age"];
<· 34
```

NB: ovviamente per chiavi che hanno nomi che contengono spazi, si è obbligati ad usare la notazione con parentesi quadre.

```
> person['first name'];
<· "Cristian"
> person.first name;
✕ Uncaught SyntaxError: Unexpected identifier
```

NB: in ogni caso è altamente consigliato usare chiavi che non contengano spazi!

NB: un altro caso in cui siamo obbligati ad usare la notazione con le parentesi quadre è quando la chiave che vogliamo prendere, ci viene passata in una variabile.

```
> var chiave = 'age';
  person[chiave];
<· 34
> var chiave2 = 'children';
  person[chiave2];
<· null
```

Come abbiamo visto prima, un oggetto può avere anche dei metodi, cioè delle funzioni.
Esempio:

```
> var person = {
      'firstname': 'Cristian',
      'lastname': 'Carrino',
      'age': 34,
      'children': null
  };

  person.fullName = function() {
      return this.firstname + ' ' + this.lastname;
  }
```

Il this. che vediamo all'interno della funzione, vuol dire che il firstname e il lastname che sto richiamando sono riferiti allo stesso oggetto da cui sto chiamando fullName()

```
person.fullName = function() {
    return this.firstname + ' ' + this.lastname;
}
```

Per poter richiamare quel metodo dovrò fare:

```
> person.fullName();
< "Cristian Carrino"
```

Altro esempio

```
> person.increaseAge = function(number) {
      this.age = this.age + number;
  }
```

In questo caso, il metodo vuole un parametro in ingresso e quello che farà sarà modificare in modo permanente la proprietà dell'oggetto.

```
> person.age;
<· 34
> person.increaseAge(10);
> person.age;
<· 44
```

GLI OGGETTI NATIVI DI JAVASCRIPT

Cristian Carrino

ENGIM
PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

WINDOW

E' l'oggetto principale di Javascript ed è rappresentato dall'intera "finestra" del browser. Se la finestra è composta da una pluralità di frames (ad esempio perchè nella pagina sono presenti degli <iframe>) il browser creerà un oggetto window per la finestra e tanti altri oggetti window per quanti sono i frame attivi nella pagina.

L'oggetto window è ricco di proprietà e metodi, vediamo di seguito i più utilizzati

Proprietà dell'oggetto window

- frames - (GET) restituisce un array con tutti i frames presenti nella pagina;
- innerHeight e innerWidth - (GET) restituisce l'altezza e la larghezza interna della finestra del browser;
- length - (GET) restituisce il numero dei frames presenti nella finestra;
- opener - (GET) restituisce il riferimento alla finestra madre che ha aperto la finestra corrente (se esiste);
- outerHeight e outerWidth - (GET) restituisce l'altezza e la larghezza esterna della finestra del browser (comprende toolbar e scrollbar);
- pageXOffset e pageYOffset - (GET) restituisce il valore (in pixel) dell'eventuale scroll sull'asse orizzontale e verticale;
- parent - (GET) restituisce il riferimento alla finestra con il frame corrente;
- self - (GET) restituisce un riferimento alla finestra corrente;
- top - (GET) restituisce un riferimento alla finestra principale;

Metodi dell'oggetto window

- alert() - mostra una finestra di avviso;
- close() - chiude la finestra corrente;
- confirm() - mostra una finestra di dialogo con un messaggio di conferma;
- open() - apre una nuova finestra;
- print() - stampa (invia alla stampante) il contenuto della finestra corrente;
- prompt() - apre una finestra di dialogo dove l'utente è invitato a scrivere qualcosa in un campo testo;
- scrollTo() - esegue uno scrolling automatico del documento verso le coordinate specificate.
- setInterval() e clearInterval() - imposta (ed annulla) una serie di istruzioni da eseguirsi ad intervalli di tempo prestabiliti;
- setTimeout() e clearTimeout() - imposta (ed annulla) una serie di istruzioni da eseguirsi una volta dopo un dato intervallo di tempo;

LOCATION

Attraverso l'oggetto location è possibile accedere ad informazioni relativamente alla URL corrente, cioè la URL attualmente caricata nella finestra del browser.

L'oggetto location, come history, è figlio dell'oggetto window da cui discende direttamente.

Proprietà dell'oggetto location

- hash - (GET/SET) restituisce o setta eventuale ancora (#) presente nella URL;
- hostname - (GET/SET) restituisce o setta l'hostname della URL;
- href - (GET/SET) restituisce o setta l'intera URL;
- pathname - (GET/SET) restituisce o setta il path (cartella / file) della URL;
- protocol - (GET/SET) restituisce o setta il protocollo della URL;

Metodi dell'oggetto location

- assign() - carica una nuova URL all'interno della finestra
- reload() - ricarica la pagina corrente;
- replace() - sostituisce la URL corrente con una nuova;

NB: A prima vista i metodi asign() e replace() potrebbero apparire identici in realtà vi è una differenza rilevante: con il primo viene caricata una nuova URL (e quella precedente resta nella cronologia) mentre col secondo la URL corrente viene sostituita (e pertanto non è più raggiungibile attraverso il tasto "back").

HISTORY

L'oggetto history fa parte dell'oggetto window e contiene informazioni relative alla cronologia delle URL visitate all'interno della finestra corrente.

Proprietà dell'oggetto history

- lenght che restituisce il numero delle URL presenti nella cronologia.

Metodi dell'oggetto location

- back() - carica la URL precedente a quella corrente (corrisponde al tasto back del browser);
- forward() - carica la URL successiva a quella corrente (corrisponde al tasto next del browser);
- go() - carica una specifica URL tra quelle presenti in cronologia;

NAVIGATOR

Permette di accedere ad una serie di informazioni sul browser utilizzato.

Proprietà dell'oggetto navigator

- appVersion - (GET) restituisce informazioni sulla versione del browser in uso;
- cookieEnabled - (GET) determina se il browser accetta o meno i cookie;
- geolocation - (GET) restituisce un oggetto geolocation che può essere utilizzato per determinare la posizione geografica dell'utente;
- language - (GET) restituisce il linguaggio di default impostato nel browser;
- userAgent - (GET) restituisce l'user-agent che il browser ha inviato al server;

Metodi dell'oggetto navigator

- javaEnabled() - per verificare se il browser supporta o meno Java

SCREEN

L'oggetto screen contiene una serie di informazioni relative allo schermo.

Proprietà dell'oggetto screen

- availWidth e availHeight - (GET) restituisce le dimensioni della finestra del browser, esclusa la taskbar;
- colorDepth - (GET) restituisce la profondità del colore (bits x pixel);
- pixelDepth - (GET) restituisce la risoluzione colore dello schermo (bits x pixel);
- width e height - (GET) restituisce le dimensioni dello schermo;

DOCUMENT

Il Document Object Model (o, più brevemente, DOM) è uno standard ufficiale del W3C attraverso il quale la struttura di un documento (come ad esempio una pagina HTML) è rappresentata sotto forma di un modello orientato agli oggetti.

In pratica ogni elemento della struttura del documento è rappresentato sotto forma di nodo di un elemento padre creando una sorta di "albero".

Attraverso Javascript è possibile manipolare il DOM della pagina. Ogni elemento della pagina viene rappresentato come un oggetto il quale può essere manipolato attraverso i suoi metodi e le sue proprietà.

L'oggetto di base per la manipolazione del DOM è document il quale rappresenta la radice del documento (cioè la pagina web).

L'oggetto document è il genitore di tutti gli altri elementi della pagina web.

Altri oggetti di Javascript: Date, Math, Array, RegExp

```
var data = new Date();
var vettore = new Array();
var re = new RegExp();
var i = Math.ceil(0.97);
```

SCREEN

L'oggetto screen contiene una serie di informazioni relative allo schermo.

Proprietà dell'oggetto screen

- availWidth e availHeight - (GET) restituisce le dimensioni della finestra del browser, esclusa la taskbar;
- colorDepth - (GET) restituisce la profondità del colore (bits x pixel);
- pixelDepth - (GET) restituisce la risoluzione colore dello schermo (bits x pixel);
- width e height - (GET) restituisce le dimensioni dello schermo;

IL FORMATO JSON

Cristian Carrino

ENGIM
PIEMONTE
Formazione Orientamento
Cooperazione Lavoro

```
var auto = { "marca": "Audi", "modello": "A4", "colore": "Nero" };
```