Capire JavaScript: gli oggetti

Gli oggetti in JavaScript

Paolo Caramanica

Sommario

- Gli oggetti
  - Che cos'è un oggetto?
  - Creare un oggetto in JavaScript
  - Proprietà e metodi
  - Live demo
- Costruttori e classi
  - I costruttori e l'operatore new
  - Introduzione alle classi
  - Differenza con gli altri linguaggi
  - Live demo
- Gli oggetti in memoria
  - Salvataggio per riferimento
  - Copia di oggetti
  - Il Garbage Collector
  - Live demo
- JSON
  - Rappresentazione di oggetti in JSON
  - Live demo

Gli oggetti

Introduzione agli oggetti in JavaScript

Che cos'è un oggetto?

- In JavaScript, i valori assegnabili ad una variabile possono essere dati di tipo:
  - **Primitivo**, come numeri e stringhe, che rappresentano entità elementari
  - **Oggetti**, che rappresentano entità più complesse
- Gli oggetti:
  - Contengono collezioni di dati di diverso tipo (stringhe, numeri o anche altri oggetti)
  - Sono nella forma chiave-valore

Creare un oggetto in JavaScript

- In JavaScript, gli oggetti sono rappresentati tra parentesi graffe
  - La seguente istruzione crea un oggetto vuoto di nome pluto

```
let pluto = {}
```

  - La seguente istruzione crea un oggetto di nome rettangolo che contiene due dati (**proprietà**) di tipo numerico (base e altezza):

```
let rettangolo = {
    base: 15,
    altezza: 8,
}
```

- Accesso alle proprietà di un oggetto:
  - Dot notation: rettangolo.base
  - Square-bracket notation: rettangolo['altezza']
- Dopo la creazione, possono essere aggiunte (o eliminate) proprietà ad un oggetto

Proprietà e metodi

- In JavaScript, qualunque valore assegnabile ad una variabile può essere assegnato alla proprietà di un oggetto:
  - Anche una funzione può essere assegnata ad una variabile
  - Se viene assegnata alla proprietà di un oggetto, questa prende anche il nome di **metodo**
- Aggiungiamo un metodo all'oggetto rettangolo:

```
let rettangolo = {
   base: 15,
   altezza: 8,
   calcolaArea: function() {
      return this.base * this.altezza;
   },
}
```

- In un metodo, this rappresenta l'oggetto stesso

Live demo

Costruttori e classi

Creare oggetti «simili» in JavaScript

I costruttori e l'operatore new

- Se vogliamo creare diversi oggetti rettangolo, con diversi valori di base e altezza, utilizziamo un costruttore:

```
function Rettangolo(b,h) {
    this.base = b;
    this.altezza = h;
    this.calcolaArea = function() {
        return this.base * this.altezza;
    }
}
```

- Il costruttore deve essere chiamato con l'operatore **new:**

```
let rettangolo1 = new Rettangolo(4,2)
```

Introduzione alle classi

- A partire dalle ultime versioni, anche in JavaScript sono state introdotte le classi.
- Esempio:

```
class Rettangolo {
    constructor(b,h) {
        this.base = b;
        this.altezza = h;
    }
    calcolaArea() {
        return this.base*this.altezza;
    }
}
```

- Utilizzo:

```
let rettangolo2 = new Rettangolo(5,3);
```

Differenze con gli altri linguaggi

- Rispetto ai più comuni linguaggi di programmazione che supportano il paradigma OOP, in JavaScript:
  - Gli oggetti possono essere definiti anche senza le classi
  - Anche se creato tramite una classe, può essere modificato (con aggiunta e cancellazione di proprietà)
  - La differenza tra proprietà e metodi è «sfumata»

Live demo

Gli oggetti in memoria

Copiare e clonare oggetti

Salvataggio per riferimento

- In JavaScript, le variabili contenenti dati di tipo primitivo sono salvate **per valore**
  - Esempio:
    ```
    let str = 'Ciao';
    ```
    Nella variabile str è contenuta la stringa 'Ciao'
- Gli oggetti sono salvati **per riferimento**
  - Esempio:
    ```
    let obj = { a: 24 }
    ```
    Nella variabile obj non è contenuto l'oggetto, ma l'indirizzo di memoria in cui è salvato l'oggetto, cioè il riferimento.

Copia di oggetti (1)

- Le variabili contenenti dati di tipo primitivo sono anche copiate **per valore**
  - Esempio:

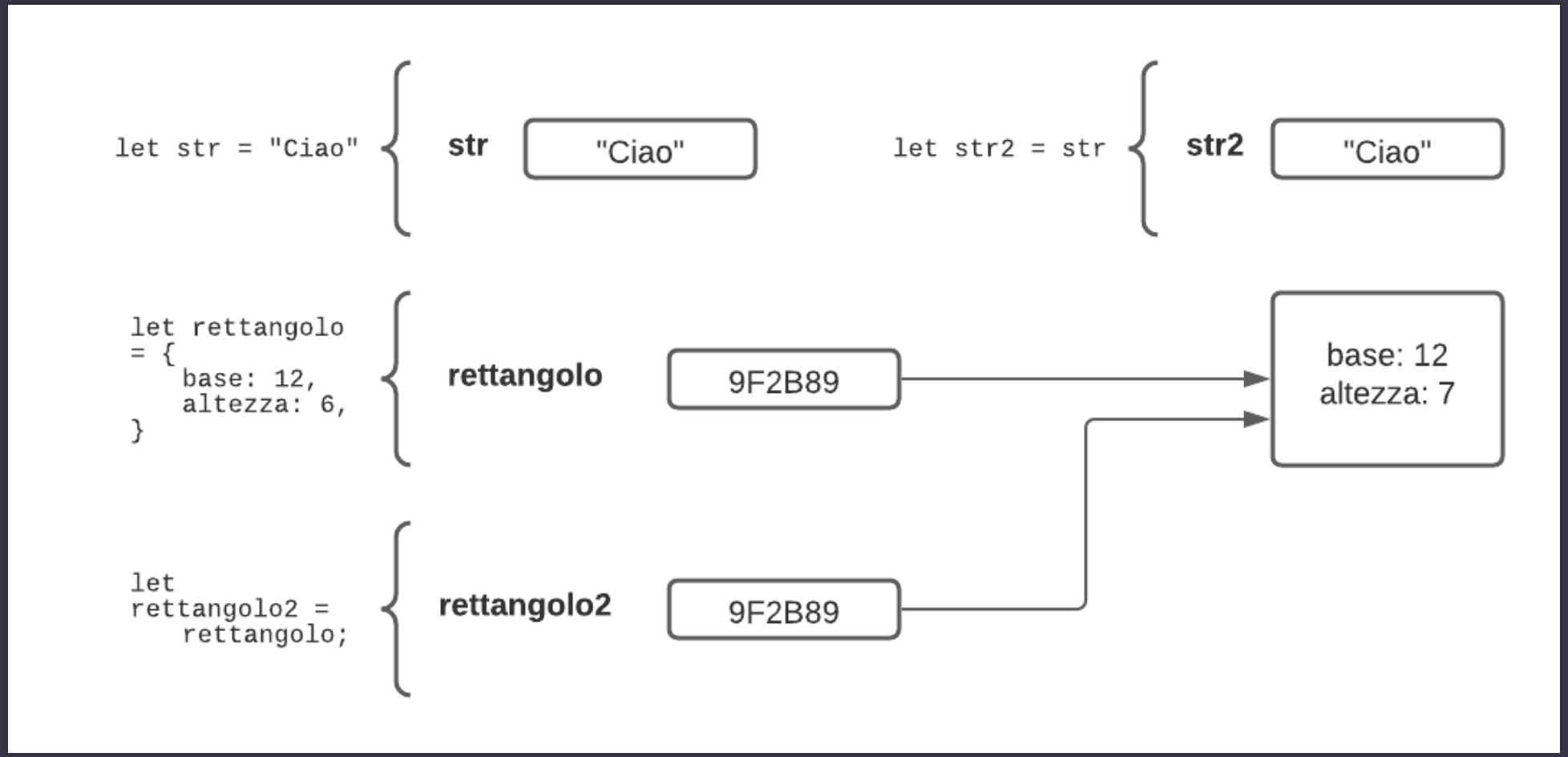
```
let str = 'Ciao';
let str2 = str;
```

    Nella variabile str è contenuta la stringa 'Ciao', nella variabile str2 una copia della stessa stringa (la stringa viene duplicata).
- Gli oggetti sono copiati **per riferimento**
  - Esempio:

```
let obj = { a: 24 };
let obj2 = obj;
```

    Nella variabile obj è contenuto l'indirizzo di memoria in cui è salvato l'oggetto, in obj2 una copia del contenuto di obj, cioè lo stesso riferimento (l'oggetto non viene duplicato).

Copia di oggetti (2)



Il Garbage Collector

- Ricapitolando, se creiamo un oggetto

```
let obj = {
    a: 29,
    b: 9,
}
```

  l'oggetto viene salvato da qualche parte in memoria, mentre l'indirizzo della locazione iniziale di tale area di memoria viene salvato nella variabile obj.

- Che succede se eseguiamo la seguente istruzione?

```
obj = null;
```

  - Viene eliminato l'indirizzo dell'oggetto dalla variabile obj
  - L'oggetto in memoria non è più accessibile in quanto non abbiamo più l'indirizzo

- Il **Garbage Collector**, in automatico, elimina dalla memoria tutti gli oggetti non più raggiungibili

Live demo

JSON

Un formato per lo scambio di dati

Rappresentazione di oggetti in JSON

- **JSON** (JavaScript Object Notation): nato inizialmente per JavaScript, è un formato per la rappresentazione, sotto forma di stringa, di valori e oggetti complessi e costituisce uno standard diffuso per lo scambio di dati.
- Metodi JavaScript:
  - JSON.stringify: converte un oggetto JavaScript in JSON
  - JSON.parse: converte un JSON in oggetto JavaScript

Live demo

Contatti

Email: paolocaramanica@gmail.com

Linkedin: https://it.linkedin.com/in/paolo-caramanica-436942a/it-it

Facebook: https://it-it.facebook.com/paolo.caramanica

Twitter: https://twitter.com/PaoloCaramanica

Sito web: http://www.paolocaramanica.net

Q&A