Javascript jQuery

Impariamo JQuery

Giulio Angiani
I.I.S. "Blaise Pascal" - Reggio Emilia

JQuery and AJAX

Cos'è AJAX

AJAX è un acronimo che sta per

Asynchronous **J**avaScript **A**nd **X**ML.

AJAX NON E' UN NUOVO LINGUAGGIO

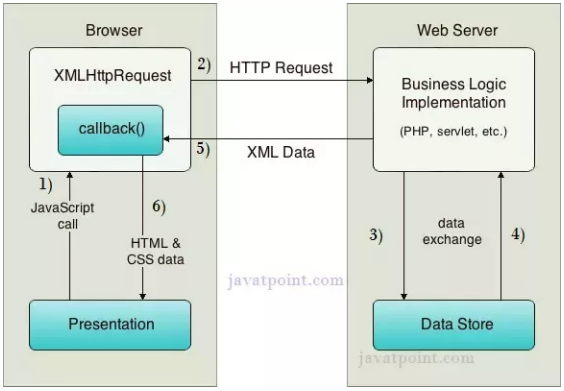
AJAX è l'unione di Javascript con l'architettura dei browser che permette di comunicare con il **server** in maniera **asincrona** e **dietro le quinte**

OBIETTIVO : permettere invio e ricezione di dati da e per il server **SENZA** dover ricaricare la pagina

- Modificare contenuti di parti della pagina HTML
- Inviare e ricevere dati in vari formati (JSON, XML, TXT, HTML)
- Migliorare le performance dell'applicazione e l'esperienza utente

AJAX - architettura

L'architettura di AJAX è riportata in figura



Rispetto all'architettura classica del web possiamo

- recuperare dati dal server con una **funzione javascript**
- controllare lo **stato** della comunicazione (ovvero che tutto sia andato bene o se c'è stato qualche problema gestirlo in maniera corretta)
- modificare (se necessario) il **contenuto** della pagina HTML
- gestire le risposte in maniera **asincrona** (ovvero nel momento in cui arrivano, senza sapere quando sarà)

AJAX - architettura

Utilizza gli stessi metodi della form e dei link per comunicare col server

- POST
  - es: invio dei dati di una form
- GET

Intercetta lo stato della comunicazione

- pagina non trovata
- errore sul server
- tutto ok

Permette di effettuare operazioni sulla pagina HTML

- prima dell'invio dei dati
- alla ricezione dei dati
  - in caso di successo
  - in caso di errore

AJAX - casi d'uso

- Login con effetto di shaking in caso di errore
  - shake login!
- conferma di memorizzazione dati senza fare "Salva"
  - saving data!
- suggerimenti di compilazione
  - ricerca
- caricamento di parti della pagina in maniera asicrona
  - chat

AJAX - struttura base

in jQuery l'architettura AJAX è mappata sulla funzione **$.ajax** il codice base è:

JQUERY

```
$.ajax({
      type: "POST",                    // definisco il tipo della chiamata POST o GET
      url: "search.php",               // specifico la URL della risorsa da contattare
      data: {                          // passo dei dati alla risorsa remota
            "nome": n,                 // non obbligatori
            "cognome": c,              // questa sezione puo' anche non esserci
      },
      beforeSend: function() {         // imposto un'azione da fare prima di inviare i dati - non obbligatoria
        $("#box_risultati").html("loading...");
      },
      success: function(risposta) {    // un'azione per il caso di successo
        $("#box_risultati").html(risposta);
      },
      error: function(){               // ed una per il caso di fallimento
        $("#box_risultati").html("Oops! si sono verificati problemi di collegamento...<br>Riprova...");
      }
    });
```

Questa operazione avviene **dietro le quinte**, ovvero **senza** ricaricamento della pagina...

Alla ricezione dei dati (sezione success o error) javascript eseguirà le operazioni sui dati ricevuti dalla pagina php chiamata

AJAX - struttura base

Vediamo le sezioni nel dettaglio:

type: sostituisce method della form e può valere GET o POST. Solitamente in Ajax si usa sempre POST

url: OBBLIGATORIA. Sostituisce l'action delle form. Indica la pagina che server che dovrà gestire i dati

data: non obbligatoria. Sostituisce i campi presenti nella form. Usa il formato JSON per l'invio dei dati che sono pertanto indicati nella forma nome:valore

beforeSend: non obbligatoria. Permette di eseguire alcune operazioni prima che i dati vengano inviati. Simile alle funzioni javascript standard effettuate prima del submit

success: formalmente non obbligatoria ma praticamente si. Permette di eseguire alcune operazioni dopo che i dati sono stati restituiti dalla pagina chiamata.

error: non obbligatoria. Permette di eseguire alcune operazioni se si sono verificati errori nella chiamata al server (pagina non trovata, errore del server, etc...).
Molto utile nell'interazione con l'utente

AJAX - Login asincrono

Come primo esempio eseguiamo il login in asicrono via AJAX

Passiamo da:

```HTML
<form method=POST action='check.php'>
  <input type='text' name='usr' placeholder='username'>
  <input type='password' name='pwd' placeholder='password'>
  <br>
  <input type='submit' value='Log In'>
</form>
```

a

```HTML
<form>
  <input type='text' id='usr' placeholder='username'>
  <input type='password' id='pwd' placeholder='password'>
  <br>
  <input type='button' id='button' value='Log In'>
</form>
```

Eliminando submit, action e method

AJAX - Login asincrono

La gestione del login sarà fatta da ajax in asincrono

JQ+AJAX

```
$("#button").click(function(){          // al click del bottone
    $.ajax({
      type: "POST",                 // il method
      url: "check.php",             // la action
      data: {
          usr:  $("#usr").val(),    // passaggio parametri via POST
          pwd:  $("#pwd").val(),
      },
      // imposto un'azione per il caso di successo, ovvero il server non ha dato errore
      success: function(risposta) {
          if (risposta == 'OK') {           // ho scelto che se il login va a buon fine restituisco OK
                location = 'home.php';
          }
          else {                            // altrimenti restituisco KO
                alert("Login fallito! Riprova");
          }
      },
      // ed una per il caso di fallimento
      error: function(){
        alert("Ooops! Qualcosa è andato storto...");
      }
    });
});
```

Giulio Angiani
I.I.S. "Blaise Pascal" - Reggio Emilia