Tecnologie Web: JQuery & AJAX

Prof. Raffaele Montella, PhD
raffaele.montella@uniparthenope.it

Sommario

- Introduzione
- Selettori
- Operazioni sugli oggetti selezionati
- Eventi
- AJAX
- JSON
- jQuery + AJAX
- Conclusioni

Introduzione

- Libreria javascript
- The Write Less, Do More
- Serve a semplificare lo scripting client
- Semplificazione di operazioni frequenti
- Cross - browser

Introduzione

- Selezionare gli elementi HTML con i selettori CSS
- Gestire gli eventi provocati dalle interazioni utenti
- Animazioni (effects)
- Modificare la struttura ad albero (DOM)
- Selezionare gli elementi HTML in base alla loro posizione

Introduzione

- Gestire le comunicazioni Ajax con un server
- Gestione json

Aggiungere la libreria jQuery

- Come risorsa esterna:

```
<head>
    <script src="https://code.jquery.com/jquery-
2.1.4.js"></script>
</head>
```

- Come risorsa locale:

```
<head>
    <script src="lib/jquery-1.12.3.min.js"></script>
</head>
```

- Consultare http://jquery.com per la scelta della versione.

Aggiungere la libreria jQuery

- Come risorsa da Content Delivery Network (Google CDN):

```
<head>
    <script="https://ajax.googleapis.com/ajax/libs/jquery -
1.12.3.min.js"></script>
</head>
```

- Come risorsa da Content Delivery Network (Microsoft CDN):

```
<head>
    <script="http://ajax.aspnetcdn.com/ajax/jQuery/jquery -
1.12.3.min.js"></script>
</head>
```

- Questa soluzione ha il vantaggio che molti utenti lo hanno già scaricato per cui lo recuperano dalla cache.

Sintassi

- Il funzionamento di jQuery segue il modello seleziona gli elementi ed esegue un'azione su di essi.

```
$("css selector").action()
```

Alias per la funzione jQuery.

Seleziona alcuni elementi html della pagina attraverso id o classe di appartenenza.

Un metodo (funzione) da eseguire sugli elementi selezionati.

Esempi

```
/* Nasconde tutti gli elementi <p> */
$("p").hide()

/* Nasconde tutti gli elementi caratterizzati da una classe CSS "test" */
$(".test").hide()

/* Nasconde l'elemento il cui id è test */
$("#test").hide()
```

Caricamento del documento

- Non è possibile lavorare sul Document Object Model prima che non sia completamente definito (caricamento della pagina).

- Senza jQuery:
  ```
  window.onload = function() { codice }
  ```

- Con JQuery:
  ```
  $(document).ready(function() { codice })
  ```

- Con JQuery (versione compatta):
  ```
  $(function() { codice })
  ```

Funzioni di callback - funzioni come argomenti

- Una definizione di funzione anonima è passata come argomento ad un'altra funzione (click).

- Questa seconda funzione invocherà (callback) il codice passato al momento opportuno (click dell'utente)

```
$("#push").click(
   function(){
      alert("Hello!);
   }
);
```

- La callback function può accedere alle variabili della funzione contenitrice e alle variabili globali – vedere javascript closure

Selettore CSS

Element selector	`$("p")`	Seleziona tutti gli elementi p
#id selector	`$("#idValue")`	Seleziona un unico elemento con attributo id="idValue"
.class selector	`$(".classValue")`	Seleziona tutti gli elementi con attributo class="classValue"

Selettore CSS

Sintassi	Descrizione
`$("*")`	Seleziona tutti gli elementi
`$(this)`	Seleziona l'elemento HTML corrente
`$("p.intro")`	Selezione tutti gli elementi p con class="intro"
`$("p:first")`	Seleziona il primo elemento p
`$("ul li:first")`	Seleziona il primo del primo
`$("ul li:first-child")`	Seleziona il primo di ogni
`$("[href]")`	Seleziona tutti gli elementi con attributo href

Selettore CSS

Sintassi	Descrizione
`$("a[target='_blank']")`	Seleziona tutti gli <a> con un attributo target uguale a "_blank"
`$("a[target!='_blank']")`	Seleziona tutti gli <a> con un attributo target diverso da "_blank"
`$(":button")`	Seleziona tutti i <button> e gli elementi <input> con type="button"
`$("tr:even")`	Seleziona tutti gli elementi <tr> di posizione pari
`$("tr:odd")`	Seleziona tutti gli elementi <tr> di posizioni dispari

Confronto tra metodi DOM e jQuery

DOM	JQuery
`getElementById("idValue")`	`$("#idValue")`
`getElementsByTagName("tag")`	`$("tag")`
`getElementsByName("somename")`	`$("[name='somename']")`
`querySelector("css selector")`	`$("css selector")`
`querySelectorAll("css selector")`	`$("css selector")`

Eventi

- Un evento è una qualsiasi azione che un utente può effettuare su di una pagina web.
- Esempi:

Mouse	**Keyboard**	**Form**	**Document WIndow**
click	keypress	submit	load
dblclick	keydown	change	resize
mouseenter	keyup	focus	scroll
mouseleave		blur	unload

Eventi

- E'importante catturare un evento per avere un comportamento personalizzato rispetto a quello di default del browser.

- Sintassi per catturare e gestire gli eventi:

```
/* Gestire l'evento click su tutti gli elementi paragrafo */


$("p").click(function(){
   $(this).hide();
});
```

Effetti

- Esempi di effetti
  - hide(), show(), toggle()
  - fade()
  - slide()
  - animate()
  - chaining: possibilità di concatenare gli effects

Manipolazione

- Leggere/modificare il contenuto degli elementi/attributi selezionati
- Aggiungere e rimuovere elementi e/o attributi
- Leggere e modificare il CSS

Manipolazione

- .text() - Imposta o restituisce il contenuto testuale dell'elemento selezionato.

  .html() - Imposta o restituisce il contenuto HTML dell'elemento selezionato.

- .val() - Imposta o restituisce il valore di un campo di una form.

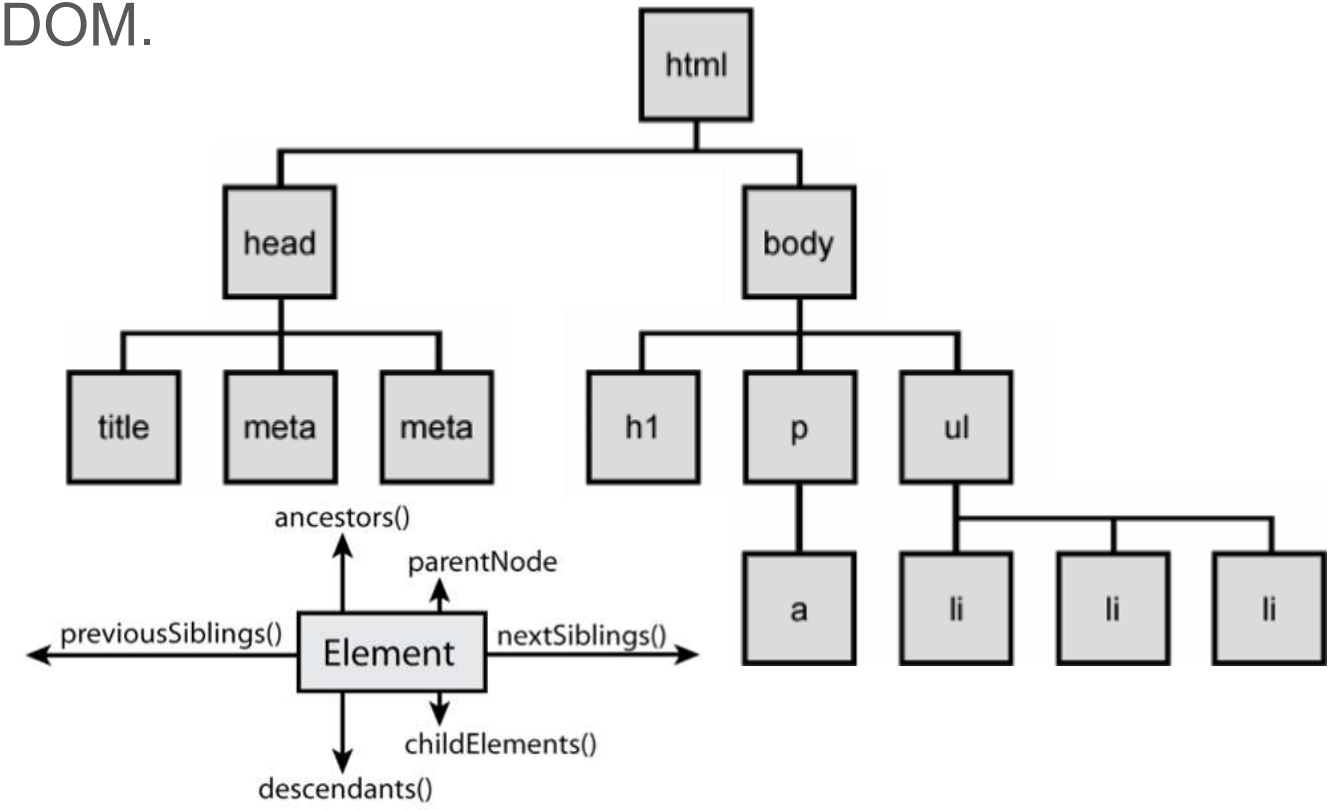
- .css() – Imposta o restituisce il valore di una proprietà CSS.

- append(), remove(), ...

Taversing

- Usato per localizzare gli elementi in base alla loro posizione nell'albero del DOM.

```
$(document).ready(
    function(){
        $("a").
            parent();
    }
);
```



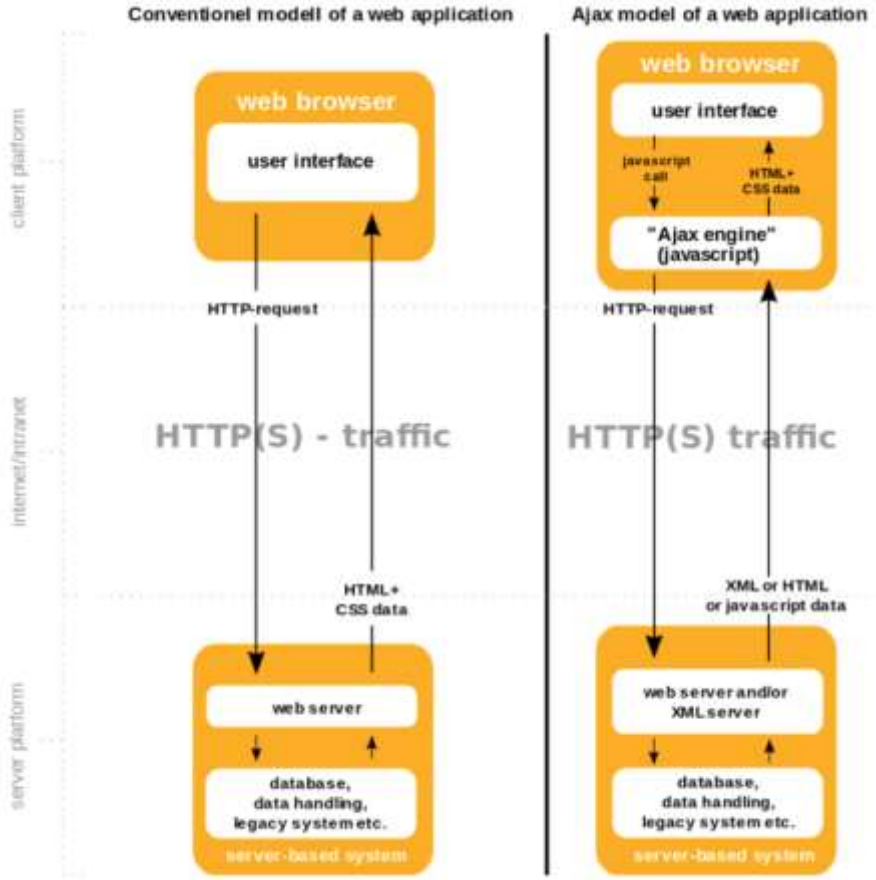
AJAX: Asynchronous Javascript and XML

- **ASINCRONO:**
  consente di caricare dati in background senza dover ricaricare l'intera pagina

- **JAVASCRIPT:**
  insieme di tecnologie web lato client

- **XML:**
  i dati restituiti possono essere XML, JSON, TXT, ...

AJAX: Asynchronous Javascript and XML



AJAX: Asynchronous Javascript and XML

- Ajax è basato sull'oggetto XMLHttpRequest.
- I vari tipi di browser gestiscono le chiamate Ajax in modo diverso.
- Questo significa che occorre aggiungere codice extra per capire su quale browser sta girando il codice.
- jQuery gestisce le differenze e consente di scrivere codice più compatto.

AJAX & jQuery

XMLHttpRequest

```
// 1 - Initialize the Http Request
Object. var xhr = new
XMLHttpRequest(); xhr.open('get',
'http://tour-
pedia.org/api/getPlaceDetails?id=1');
// 2 - Gestisci la risposta
xhr.onreadystatechange = function ()
{
if (xhr.readyState === 4)
if(xhr.status === 200)
document.getElementById("response").i
nnerHTML = xhr.response;
else alert('Error: ' + xhr.status +
"-"+ xhr.readyState); };
// 3 - Invia la richiesta a tour-
pedia xhr.send(null);
```

JQuery

```
// Invia la richiesta
$.get('http://tour-
pedia.org/api/getPlaceDetails?
id=1', function(data) {
// Gestisci la risposta
$("#response").text(data.name)
; });
```

jQuery + AJAX

- load()
  - Carica dati dal server e li pone nell'elemento selezionato
  - $(selector).load(URL,data,callback);

- $.get()
  - carica dati dal server con una richiesta HTTP GET
  - $.get(URL,callback);

jQuery + AJAX

- $.post()
  - richiede dati dal server con una richiesta HTTP POST
  - $.post(URL,data,callback);

- $.getJSON()
  - richiede dati JSON dal server con una richiesta HTTP GET
  - $.getJSON(url,data,success(data))

JSON - Javascript Object Notation

- Formato per scambio dati indipendente dal linguaggio (javascript, php, ...)

- Esempio di codice lato server che restituisce un Json

  - https://api.meteo.uniparthenope.it/products/wrf5/forecast/com63049

- Per vedere meglio i dati json su browser installare estensioni come jsonView

Esempio: getJSON

```
var url="http://tour-pedia.org/api/getPlacesStatistics";


$(document).ready(function(){
    $("#push").click(function(){
        $.getJSON(url,function(result){
            $("#result")
                .text("Le strutture ricettive in Amsterdam sono" +
                result.Amsterdam.accommodation);
        });
    });
});
```

Esempio: getJSON (con parametri)

```
var url="http://tour-pedia.org/api/getPlaceDetails";

$(document).ready(function(){
    $("#push").click(function(){
        var data={"id": 1};

        $.getJSON(url,data,function(result){
            $("#result").text(result.name);
        });
    });
});
```

Conclusioni

- jQuery è una libreria javascript.
- Permette di semplificare notevolmente il codice.
- È cross-browser, tutte le differenze sono appianate.
- AJAX consente di non ricaricare l’intera pagina ma di aggiornare solo alcuni elementi.

Conclusioni

- AJAX non è cross-browser, ma l’implementazione AJAX di jQuery si.
- JSON è un formato di scambio dati per webservice di tipo REST.
- getJSON permette di interagire con webservice di tipo REST in maniera semplice.