SEMINARIO INTRODUZIONE A JQUERY

Linguaggi per il Web

Ingegneria Informatica, Ingegneria dell'Informazione,
Sapienza Università di Roma, sede di Latina

17 Maggio 2023

Corrado Di Benedetto

ARGOMENTI

- Introduzione
- Selettori
- Eventi
- Effetti
- Modificare il DOM
- jQuery per AJAX

PERCHÉ JQUERY ?

- Semplifica la programmazione *JS* nelle applicazioni web
- Motto jQuery: "***write less, do more***"
- Semplice da imparare
- Attenua i problemi di compatibilità dei browser
- Compatibile con altre librerie *JS*
- Molto popolare ed estensibile

CARATTERISTICHE

- ***Manipolazione del DOM*** – semplifica la selezione, la navigazione e la modifica degli elementi DOM
- **Gestione degli eventi** – elegante metodo per catturare gli eventi senza inserire nell'HTML i gestori di eventi
- **AJAX** – semplifica molto l'utilizzo
- **Animazioni** – contiene molti effetti di animazione
- **Leggera** – circa 95 KB
- **Cross Browser** – compatibile con IE 6.0+, FF 2.0+, Safari 3.0+, Chrome e Opera 9.0+

COME USARE JQUERY?

- Download (https://jquery.com/download/) dell'ultima versione disponibile (produzione e sviluppo)
- Includere il riferimento a jQuery nel codice HTML

Locale

```
<head>
<script src="jquery-1.12.0.min.js"></script>
</head>
```

Rete

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js"></script>
</head>
```

```
<head>
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-1.12.0.min.js"></script>
</head>
```

SINTASSI JQUERY

```
$(selettore).azione()
```

- Il simbolo `$` definisce l'uso della libreria jQuery (`jQuery` alias `$`)
- Un `selettore` serve per selezionare uno o più elementi HTML usando la sintassi dei CSS
- Una `azione()` è l'azione da applicare su uno o più elementi HTML

Esempi

- `$(this).hide()` - nasconde l'elemento corrente
- `$("p").hide()` - nasconde tutti gli elementi paragrafi `p`
- `$(".test").hide()` - nasconde tutti gli elementi con classe uguale a `test`
- `$("#test").hide()` - nasconde l'elemente con `id` uguale a `test`

SINTASSI JQUERY

- E' buona pratica di programmazione far eseguire il codice jQuery dopo che la pagina sia stata caricata:

```
$(document).ready(function(){

   // jQuery methods go here...

});
```

oppure

```
$(function(){

   // jQuery methods go here...

});
```

ARGOMENTI

- Introduzione
- Selettori
- Eventi
- Effetti
- Modificare il DOM
- jQuery per AJAX

SELETTORI JQUERY

- I selettori permettono di ***selezionare*** *e* ***manipolare*** gli elementi HTML
- Gli elementi possono essere identificati attraverso l'*id*, la *classe*, gli *attributi*, i *valori degli attributi* e *altro*
- Confrontiamo il codice JS con quello jQuery:

```
// JavaScript
document.getElementById("idElement");

// jQuery
$("idElement");
```

- *Sintassi dei selettori*: CSS + selettori specifici

SELETTORI CSS: *$(ELEMENTO)*

- Permette di selezionare gli elementi HTML passati come parametro
  - `$("p")` tutti i paragrafi della pagina
  - `$("a")` tutti i link (e ancore) della pagina
  - `$("a,p")` tutti i link e i paragrafi della pagina
- **Esempio**

```
$(document).ready(function(){
    $("button").click(function(){
        $("p").hide();
    });
});
```

SELETTORI CSS: *$(#IDELEMENTO)*

- Permette di selezionare l'elemento della pagina che ha l'*id* passato come parametro

  - `$("#test")` **elemento della pagina con attributo** `id` **è uguale a** `test`
  - `$("div#test")` `div` **della pagina con attributo** `id` **è uguale a** `test`, **essendo l'**`id` **univoco all'interno della pagina la selezione del** `div` **poteva essere omessa**

- **Esempio**

```
$(document).ready(function(){
    $("button").click(function(){
        $("#test").hide();
    });
});
```

SELETTORI CSS: *$(.NOMECLASSE)*

- Permette di selezionare gli elementi della pagina che hanno la classe passata come parametro
  - `$(".test")` **elementi della pagina con attributo** `class` **è uguale a** `test`
  - `$("div.test")` **tutti i** `div` **della pagina con attributo** `class` **è uguale a** `test`
- **Esempio**

```
$(document).ready(function(){
    $("button").click(function(){
        $(".test").hide();
    });
});
```

SELETTORI CSS: *$(ELEMENTO[ATTRIBUTO])*

- Permette di selezionare gli elementi della pagina che hanno l'attributo passato come parametro
  - `$("img[title]")` **tutte le immagini in cui è specificato l'attributo** `title`
  - `$("input[value]")` **tutti gli elementi** `input` **in cui è specificato l'attributo** `value`
- **Esempio**

```
$(document).ready(function(){
   $("button").click(function(){
      $("img[title]") .hide();
    });
 });
```

SELETTORI CSS: *$(ELEMENTO[ATTRIBUTO=X])*

- Permette di selezionare gli elementi della pagina che hanno l'attributo passato come parametro uguale al valore *x*
  - `$("img[title='logo']")` **tutte le immagini in cui l'attributo** `title` **è uguale a** `logo`
  - `$("input[type='radio']")` **tutti gli elementi** `input` **di tipo** `radio`
- **Esempio**

```
$(document).ready(function(){
   $("button").click(function(){
      $("img[title='logo']") .hide();
    });
 });
```

SELETTORI CSS: *$(ELEMENTO[ATTRIBUTO$=X])*

- Permette di selezionare gli elementi della pagina che hanno l'attributo passato come parametro che finisce per *x*
  - `$("img[src$='.png']")` **tutte le immagini con estensione** `png`
  - `$("a[href$='.pdf']")` **tutti i link a file** `pdf`

- **Esempio**

```
$(document).ready(function(){
   $("button").click(function(){
      $("img[src$='.png']") .hide();
    });
 });
```

SELETTORI DI POSIZIONE: *$(:FIRST / :LAST)*

- Permettono di selezionare il primo `(:first)` e l'ultimo `(:last)` elemento di un insieme
  - `$("a:first")` il primo link della pagina
  - `$("p.evidenziato:first")` il primo paragrafo con classe uguale a `evidenziato`
  - `$("img[src*=logo]:last")` ultima immagine che contenga nell'indirizzo la parola `logo`
- **Esempio**

```
$(document).ready(function(){
   $("button").click(function(){
      $("a:first") .hide();
    });
 });
```

SELETTORI DI POSIZIONE: *$(:EVEN / :ODD)*

- Permettono di selezionare gli elementi in posizioni pari `(:even)` e dispari `(:odd)`
  - `$("table.test tr:even")` le righe delle tabelle in posizione pari che hanno classe `test`
  - `$("li:odd")` gli elementi delle liste in posizione dispari

- **Esempio**

```
$(document).ready(function(){
    $("button").click(function(){
        $("table.test tr:even").hide();
    });
});
```

SELETTORI DI POSIZIONE: *$(:EQ(N) / :LT(N) / :GT(N))*

- Permettono di selezionare gli elementi in posizioni uguale a n `(:eq(n))`, maggiore di *n* `(:gt(n))` e minore di *n* `(:lt(n))`
  - `$("tr:eq(4)")` il quinto elemento `tr`
  - `$("tr:lt(4)")` i primi cinque elementi `tr`
  - `$("tr:gt(4)")` dal sesto elemento `tr` in poi

- **Esempio**

```
$(document).ready(function(){
    $("button").click(function(){
        $("tr:eq(4)").hide();
    });
});
```

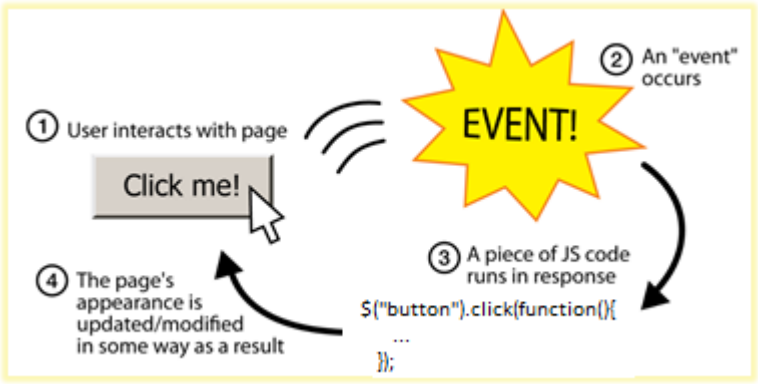
ARGOMENTI

- Introduzione
- Selettori
- Eventi
- Effetti
- Modificare il DOM
- jQuery per AJAX

EVENTI

- Gli oggetti HTML possono generare eventi ai quali corrisponderà un'azione, se il relativo codice jQuery collega all'evento un **gestore di eventi (event handler)**

```
<html>
    <head><script src="jquery-1.12.3.min.js"></script>
     <script>$(document).ready(function(){
     $("button").click(function(){
         // jQuery code go here ...
     });});</script>
    </head>
     <body>
        <input type="button" value="Click me!"/>
    </body>
</html>
```



METODO BIND

- Permette di collegare gli eventi ad un selettore e di eseguire la funzione associata (gestore di eventi)

```
$(selettore).bind(evento,[dati],function(evento){…})
```

- **Esempio**

```
$(":text").bind('focusin', function(){
        this.addClass('inserimento');
}).bind('focusout',function(){
        this.removeClass('inserimento');
})
```

Per ogni casella di testo aggiunge la classe `inserimento` quando il *focus* è sull'oggetto e rimuove la stessa classe quando il *focus* non è più nell'oggetto

SINTASSI ALTERNATIVA

```
$(selettore).bind(evento,[dati],function(evento){…})
```

equivalente

```
$(selettore).evento([dati],function(evento){…})
```

- **Esempio**

```
$("li").click(function(){$(this).hide()})
$("#mostra").mouseover(function(){$("li:hidden").show();})
```

Al click su una lista `li` nasconde gli elementi e quando il mouse passa sopra l'elemento con `id="mostra"` vengano visualizzati gli elementi `li` nascosti

ESEMPI BIND

- `$(selettore).mouseover(function(evento){…});`

si attiva quando l'utente passa sopra all'elemento selezionato con il cursore del mouse

- `$(selettore).focus(function(evento){…});`

si attiva quando l'elemento ha il focus

- `$(selettore).dblclick(function(evento){…});`

si attiva quando l'utente fa doppio click sull'elemento

- ***Passaggio dati*** alle funzioni di callback degli eventi

```
var messaggio="ciao mondo";
$("li").click({msg:messaggio},function(evento){
alert(evento.data.msg)});
```

ARGOMENTI

- Introduzione
- Selettori
- Eventi
- Effetti
- Modificare il DOM
- jQuery per AJAX

EFFETTI

Gli effetti jQuery sono legati a:

- ***Visibilità*** **(Hide, Show, Toggle)**
- ***Scorrimento*** **(Slide)**
- ***Dissolvenza*** **(Fade)**
- ***Animazione*** **(Animate)**

degli elementi HTML selezionati

EFFETTI VISIBILITÀ

- `$(selettore).hide(velocità, callback);`
- `$(selettore).show(velocità, callback);`
- `$(selettore).toggle(velocità, callback);` **alterna hide() e show()**

- `velocità` **può assumere i valori:** *fast*, *slow* **o** *millisecondi*

- **Esempio**

```
$("button").click(function(){
    $("p").hide(1000);
});

$("button").click(function(){
    $("p").show();
});

$("button").click(function(){
    $("p").toggle();
});
```

EFFETTI DISSOLVENZA

- `$(selettore).fadeIn(velocità, callback);` **nasconde con dissolvenza**
- `$(selettore).fadeOut(velocità, callback);` **rende visibile con dissolvenza**
- `$(selettore).fadeToggle(velocità, callback);` **alterna fadein() e fadeout()**
- `$(selettore).fadeTo(velocità, opacità, callback);` **applica opacità**

- `velocità` **è obbligatorio e valori possibili:** *fast*, *slow* **o** *millisecondi*
- `opacità` **è obbligatorio e specifica il valore di opacità a cui si ferma (tra 0 e 1)**

- **Esempio**

```
$("button").click(function(){
    $("#div3").fadeIn(3000);

    $("#div1").fadeOut();
    $("#div2").fadeOut("fast");

    $("#div2").fadeToggle("slow");
    $("#div3").fadeToggle(2000);

    $("#div1").fadeTo("slow", 0.15);
});
```

EFFETTI SCORRIMENTO

- `$(selettore).slideDown(velocità, callback);` **fa scendere**
- `$(selettore).slideUp(velocità, callback);` **fa salire**
- `$(selettore).slideToggle(velocità, callback);` **alterna slideDown() e slideUp()**

- `velocità` **è obbligatorio e valori possibili:** *fast*, *slow* **o** *millisecondi*

- **Esempio**

```
$("button").click(function(){
    $("#panel").slideDown(13000);

    $("#panel").slideUp("fast");

    $("#panel").slideToggle("slow");
});
```

EFFETTI ANIMAZIONE

- `$(selettore).animate({params}, velocità, callback);` fa partire
- `$(selettore).stop(stopAll, goToEnd);` fa finire

- `velocità` è obbligatorio e valori possibili: *fast*, *slow* o *millisecondi*
- `stopAll` è opzionale e specifica se le animazioni accodate devono essere ripulite. Il default è *false*, fermerà solo l'animazione attiva
- `goToEnd` è opzionale e specifica se completare immediatamente la corrente animazione. Il default è *false*

- **Esempio**

```
$("button").click(function(){
    var div = $("div");
    div.animate({height: '300px', opacity: '0.4'}, "slow");
    div.animate({width: '300px', opacity: '0.8'}, "slow");
    div.animate({height: '100px', opacity: '0.4'}, "slow");
    div.animate({width: '100px', opacity: '0.8'}, "slow");
});
```

CALLBACK E CHAINING

- Una funzione di ***callback*** sarà eseguita dopo che il corrente effetto è teminato

- `$(selettore).hide(velocità, callback);`

- Ci sono tecniche chiamate ***chaining***, che permettono l'esecuzione di più comandi jQuery uno dopo l'altro su uno stesso elemento/i

- `$("#p1").css("color", "red").slideUp(2000).slideDown(2000);`

ARGOMENTI

- Introduzione
- Selettori
- Eventi
- Effetti
- Modificare il DOM
- jQuery per AJAX

MANIPOLARE IL CONTENUTO

- `text()` - Imposta o restituisce il testo contenuto nell'elemento selezionato
- `html()` - Imposta o restituisce l'HTML contenuto nell'elemento selezionato
- `val()` - Imposta o restituisce il *value* di un elemento selezionato in una FORM HTML

- **Esempio**

```jquery
$("#btn1").click(function(){
    alert("Text: " + $("#test").text());
    alert("HTML: " + $("#test").html());
    alert("Value: " + $("#inputText").val());
    $("#test").text("Hello world!");
    $("#test").html("<b>Hello world!</b>");
    $("#inputText").val("Dolly Duck");

});
```
jQuery

```html
<p id="test">This is some <b>bold</b> text in a paragraph.</p>
<p>Name: <input type="text" id="inputText" value="Mickey Mouse"></p>
<button id="btn1">Test</button>
```
HTML

MODIFICARE GLI ATTRIBUTI

- `attr()` - Imposta e restituisce i valori di un attributo

- **Esempio**

```jQuery
$("button").click(function(){
    alert($("#w3s").attr("href"));
    $("#w3s").attr({
        "href" : "http://www.w3schools.com/jquery",
        "title" : "W3Schools jQuery Tutorial"
    });
    alert($("#w3s").attr("href"));
});
```

```HTML
<p>
   <a href="http://www.w3schools.com" id="w3s">W3Schools.com</a>
</p>
```

FUNZIONE DI CALLBACK

- `text()`, `html()`, `val()` e `attr()` possono usare la funzione di *callback*

- La funzione di *callback* prende in *input* 2 parametri:
  - **Indice** dell'elemento corrente nella lista degli elementi selezionati
  - **Valore originale**

  e restituisce in *output* una **stringa** che può essere usata dalla funzione stessa come nuovo valore

- **Esempio**

```
$("#btn1").click(function(){
    $("#test1").text(function(i, origText){
        return "Old text: " + origText + " New text: Hello world!
        (index: " + i + ")";
    });
});

$("button").click(function(){
    $("#w3s").attr("href", function(i, origValue){
        return origValue + "/jquery";
    });
});
```

AGGIUNGERE AL DOM

- E' possibile aggiungere elementi e contenuti al DOM utilizzando i seguenti metodi:
  - `append()` - Alla fine dell'elemento selezionato
  - `prepend()` - All'inizio dell'elemento selezionato
  - `after()` - Dopo l'elemento selezionato
  - `before()` - Prima dell'elemento selezionato

- **Esempio**

```
$("p").append("Some appended text.");
$("p").prepend("Some prepended text.");
$("img").after("Some text after");
$("img").before("Some text before");

var txt1="<b>I </b>";   // elemento creato con HTML
var txt2=$("<i></i>").text("love ");   // con jQuery
var txt3=document.createElement("big");  // con Javascript
txt3.innerHTML="jQuery!";   // con DOM
$("img").after(txt1,txt2,txt3); // inserisce nuovi elementi dopo img
```

RIMUOVERE DAL DOM

- `remove()` - Rimuove l'elemento selezionato e i suoi figli
- `empty()` - Rimuove i figli dell'elemento selezionato

- **Esempio**

```
$("#div1").remove();
$("#div1").empty();
$("p").remove(".test");
$("p").remove(".test, .demo");
```

MANIPOLARE LE PROPRIETÀ CSS

- `addClass()` - Aggiunge una o più classi all'elemento selezionato
- `removeClass()` - Rimuove una o più classi all'elemento selezionato
- `toggleClass()` - Alterna addClass e removeClass sull'elemento selezionato
- `css()` - Imposta o restituisce l'attributo di stile

- **Esempio**

```
$("button").click(function(){
  $("#div1").addClass("background-color");
  $("#div1").addClass("important background-color");
  $("#div1").removeClass("background-color");
  $("h1, h2, p").toggleClass("background-color");
  $("p").css("background-color");
  $("p").css("background-color", "yellow");
  $("p").css({"background-color": "yellow", "font-size": "200%"});
});
```

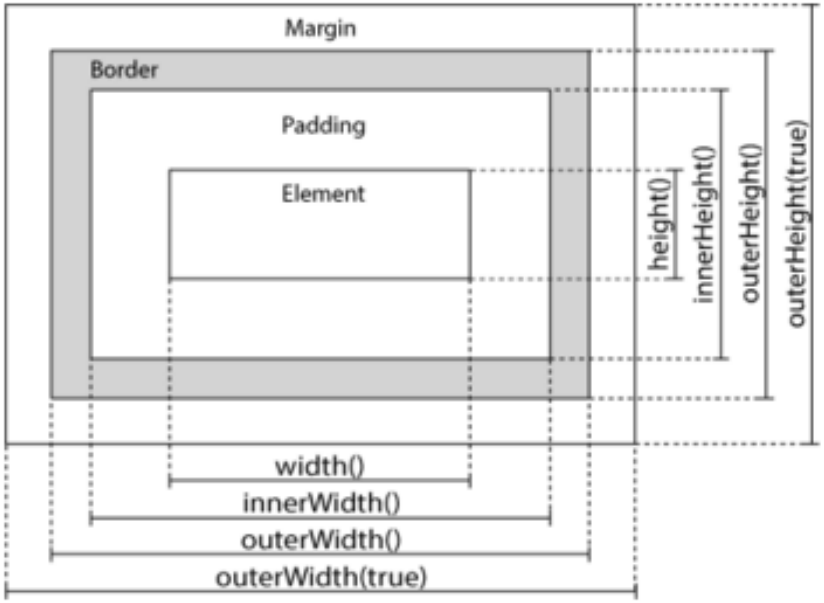
jQuery

```
.important {
    font-weight: bold;
    font-size: xx-large;
}
.background-color {
    color: blue;
}
```

CSS

DIMENSIONI DEGLI ELEMENTI

- `width()`
- `height()`
- `innerWidth()`
- `innerHeight()`
- `outerWidth()`
- `outerHeight()`



- **Esempio**

```
$("button").click(function(){
    var txt = "";
    txt += "Width: " + $("#div1").width() + "</br>";
    txt += "Height: " + $("#div1").height();
    txt += "Inner width: " + $("#div1").innerWidth() + "</br>";
    txt += "Inner height: " + $("#div1").innerHeight();
    txt += "Outer width: " + $("#div1").outerWidth() + "</br>";
    txt += "Outer height: " + $("#div1").outerHeight();
    $("#div1").html(txt);
});
```

NAVIGARE IL DOM

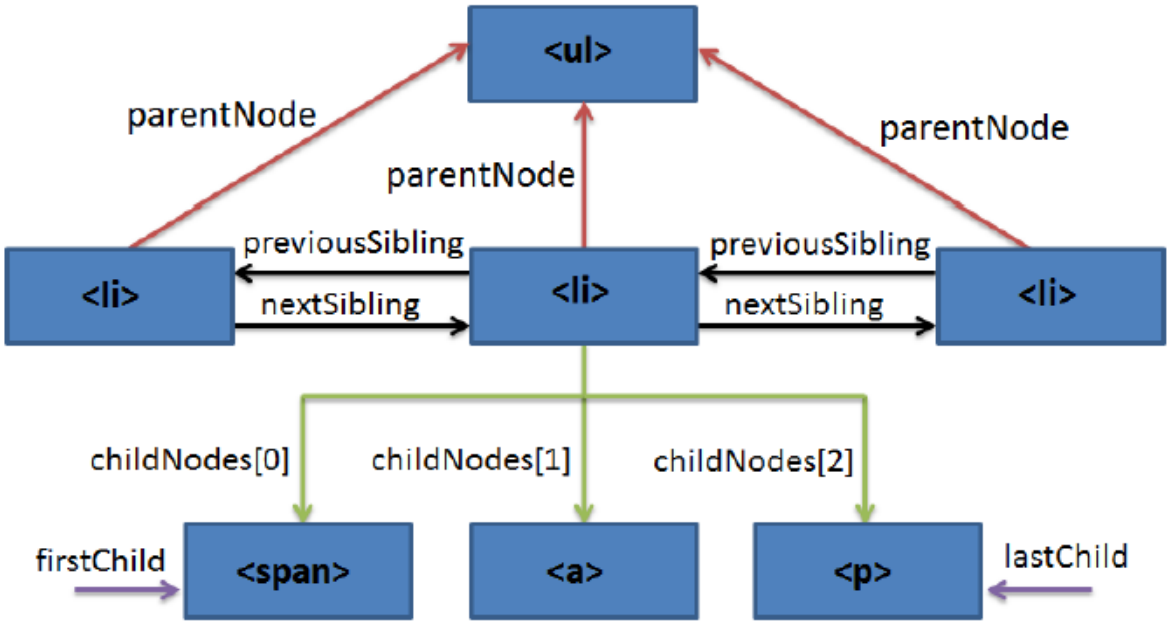
- `parent()`
- `parents()`
- `parentsUntil()`

- `children()`
- `find()`

- `siblings()`

- `next()`
- `nextAll()`
- `nextUntil()`

- `prev()`
- `prevAll()`
- `prevUntil()`



ARGOMENTI

- Introduzione
- Selettori
- Eventi
- Effetti
- Modificare il DOM
- jQuery per AJAX

JQUERY PER AJAX

- I principali metodi jQuey per AJAX sono:
  - `load()` - carica i dati da un server e li inserisce all'interno dell'elemento selezionato
  - `get()` e `post()` - caricano i dati dalla risorsa specificata, ma senza inserirli nell'elemento, corrispondono alle richieste *HTTP* di tipo *GET* e *POST*
  - `getJSON()` - carica i dati JSON dalla risorsa specificata usando una richiesta *HTTP* di tipo *GET*
  - `ajax()` - è il metodo più generico per effettuare una chiamata AJAX
  - `serialize()` - estrae tutti i valori da una *FORM HTML* inviata

METODO LOAD()

- `$(selector).load(URL,data,callback);`

carica i dati da un server e li inserisce all'interno dell'elemento selezionato

Parametri:

- `URL` è obbligatorio e specifica `URL` da cui caricare i dati
- `data` è facoltativo e specifica le coppie `chiave:valore` da inviare nella richiesta
- `callback` è facoltativo e rappresenta il nome della funzione da eseguire dopo che il metodo load() è completato

- **Esempio**

```
<body>
   <div id="div1"><h2>Let jQuery AJAX Change This Text</h2></div>
   <button>Get External Content</button>
</body>
```

HTML

```
$(document).ready(function(){
    $("button").click(function(){
        $("#div1").load("demo_test.txt", function(responseTxt, statusTxt, xhr){
            if(statusTxt == "success")
                alert("External content loaded successfully!");
            if(statusTxt == "error")
                alert("Error: " + xhr.status + ": " + xhr.statusText);
        });
    });
});
```

jQuery

METODO GET()

- `$.get(URL,callback)`

carica i dati da una risorsa specificata, senza inserirli nell'elemento, e corrisponde a una richiesta *HTTP* di tipo *GET*

Parametri:

- `URL` è obbligatorio e specifica `URL` a cui inviare la richiesta *HTTP*
- `callback` è facoltativo e rappresenta il nome della funzione da eseguire se la richiesta ha successo

- **Esempio**

```php
<?php
   echo "This is some text from an external php file.";
?>
```
PHP

```jquery
$(document).ready(function(){
    $("button").click(function(){
        $.get("demo_test.php", function(data, status){
            alert("Data: " + data + "\nStatus: " + status);
        });
    });
});
```
jQuery

METODO POST()

- `$.post(URL,data,callback)`

carica i dati da una risorsa specificata, senza inserirli nell'elemento, e corrisponde a una richiesta *HTTP* di tipo POST

Parametri:

- `URL` è obbligatorio e specifica `URL` a cui inviare la richiesta *HTTP*
- `data` è facoltativo e specifica le coppie `chiave:valore` da inviare nella richiesta
- `callback` è facoltativo e rappresenta il nome della funzione da eseguire se la richiesta ha successo

- **Esempio**

```jquery
$(document).ready(function(){
    $("button").click(function(){
        $.post("demo_test_post.php",
        {
          name: "Donald Duck",
          city: "Duckburg"
        },
        function(data,status){
          alert("Data: " + data + "\nStatus: " + status);
        });
    });
});
```

jQuery

```php
<?php
if( $_REQUEST["name"] && $_REQUEST["city"]){
   $name = $_REQUEST['name'];
   $city = $_REQUEST['city'];
   echo "Dear ". $name . ".";
   echo "Hope you live well in ". $city . ".";
}
?>
```

PHP

METODO GETJSON()

- `$.getJSON(URL,callback)`

carica i dati JSON dalla risorsa specificata usando una richiesta *HTTP* di tipo *GET*

Parametri:

- `URL` è obbligatorio e specifica `URL` a cui inviare la richiesta *HTTP*
- `callback` è facoltativo e rappresenta il nome della funzione da eseguire se la richiesta ha successo

- **Esempio**

```
<body>
   <p>Click on the button to load result.json file</p>
   <div id = "stage" style = "background-color:#eee;">
   <input type = "button" id = "driver" value = "Load Data" />
</body>
```

HTML

```
$(document).ready(function() {
    $("#driver").click(function(event){
       $.getJSON('/jquery/result.json', function(jd) {
          $('#stage').html('<p> Name: ' + jd.name + '</p>');
          $('#stage').append('<p>Age : ' + jd.age+ '</p>');
          $('#stage').append('<p> Sex: ' + jd.sex+ '</p>');
       });
    });
 });
```

jQuery

```
{
  "name": "Zara Ali",
  "age" : "67",
  "sex": "female"
}
```

result.json

METODO AJAX()

- `$.ajax(opzioni)`

è il metodo più generico per effettuare una chiamata AJAX, restituisce l'oggetto `XMLHttpRequest` creato

Parametri:

- `opzioni = {nome:valore, nome:valore, ... }`

  configurano la richiesta AJAX, sono opzionali ad eccezione della coppia con nome `URL`

- **Esempio**

```html
<body>
   <div id="div1"><h2>Let jQuery AJAX Change This Text</h2></div>
   <button>Get External Content</button>
</body>
```

HTML

```jquery
$(document).ready(function(){
    $("button").click(function(){
        $.ajax({url: "demo_test.txt", success: function(result){
            $("#div1").html(result);
        }});
    });
});
```

jQuery

METODO SERIALIZE()

- `$.serialize()`

estrae tutti i valori da una *FORM* HTML

- **Esempio**

```jquery
$(document).ready(function(){
    $("button").click(function(){
        $("#results").text($("form").serialize());
    });
});
```

jQuery

```html
<form>
  <select name="single">
    <option>Single</option>
    <option>Single2</option>
  </select>
  <br>
  <select name="multiple" multiple="multiple">
    <option selected="selected">Multiple</option>
    <option>Multiple2</option>
    <option selected="selected">Multiple3</option>
  </select>
  <br>
  <input type="checkbox" name="check" value="check1" id="ch1">
  <label for="ch1">check1</label>
  <input type="checkbox" name="check" value="check2" checked="checked" id="ch2">
  <label for="ch2">check2</label>
  <br>
  <input type="radio" name="radio" value="radio1" checked="checked" id="r1">
  <label for="r1">radio1</label>
  <input type="radio" name="radio" value="radio2" id="r2">
  <label for="r2">radio2</label>
</form>

 <p id="results">results</p>
 <button>Serialize</button>
```

HTML

RISORSE

- https://jquery.com/
- https://api.jquery.com/
- https://learn.jquery.com/
- https://it.wikibooks.org/wiki/JQuery/Introduzione
- https://www.w3schools.com/jquery/
- https://www.tutorialspoint.com/jquery/index.htm

- https://jqueryui.com/
- https://jqueryui.com/demos/
- https://learn.jquery.com/jquery-ui/getting-started/
- https://www.tutorialspoint.com/jqueryui/index.htm