



# Introduzione a jQuery

# Che cosa è jQuery?

- The Write Less, Do More javascript library
- Serve a semplificare lo scripting client-side
  - Semplificazione di operazioni frequenti
  - multi - browser

# Bibliografia

- Tutorial on line
  - [jQuery w3cschools](#)
  - [jQuery code academy](#)
  - [jQuery Learning Centre](#)
- Documentazione
  - [Jquery web api reference](#)

# Sitografia

- [jQuery official site](#)
- [jQuery User Interface](#) : jQuery UI is a curated set of user interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript Library
- [jQuery Mobile](#) jQuery Mobile is a HTML5-based user interface system designed to make responsive web sites and apps that are accessible on all smartphone, tablet and desktop devices.

# Utilizzo di jQuery

- **Selezionare** gli elementi HTML con i selettori CSS
- Gestire gli **eventi** provocati dalle interazioni utenti
- Animazioni (effects)
- Modificare la struttura ad albero (DOM) dell'HTML
- Selezionare gli elementi HTML in base alla loro posizione
- Gestire le comunicazioni Ajax con un server
- Gestione json

# Javascript Coding Tips

- Un ambiente dove testare codice client HTML, Javascript
- Per il debugging del codice lato client usare l'ambiente sviluppatori (ctrl-shift-j)

# Aggiungere la libreria jQuery

- Agganciare direttamente la libreria jQuery

```
<head>
```

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

```
</head>
```

- Scaricare in locale l'ultima versione 1.x o 2.x (non compatibile con IE 6,7,8)

```
<head>
```

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

```
</head>
```

# Aggiungere la libreria jQuery

- Includerla da un Content Delivery Network (CDN)

- Google CDN

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

- 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





# jQuery Syntax

Il funzionamento di jQuery segue il modello **seleziona** gli elementi ed esegui un'**azione** su di essi

**\$("css selector").action()**

 **\$()** accedi a jQuery

 CSS selector seleziona alcuni elementi html della tua pagina

 Una jQuery action da essere eseguita sugli elementi selezionati

# Esempi

- `$("p").hide()` // hides all `<p>` elements.
- `$(".test").hide()` // hides all elements with `class="test"`.
- `$("#test").hide()` //hides the element with `id="test"`.

# Esempi

```
$("#p").hide();
```

nasconde tutti i paragrafi

```
$(".test").hide();
```

nasconde tutti gli elementi con attributo  
class="test"

```
$("#cap").hide();
```

nasconde l'elemento con attributo  
id="cap"

# Window.onload()

Non possiamo lavorare sul DOM prima che la pagina non sia completamente caricata

## 1. classical way

```
window.onload = function()  
  { // inserisci qui il codice che modifica il DOM }
```

## 2. jQuery

```
$(document).ready(function()  
  { // inserisci qui il codice che modifica il DOM });
```

## 3. jQuery simplified

```
$(function()  
  { // inserisci qui il codice che modifica il DOM });
```

# Callback Function – Funzioni come argomenti

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

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

Questa seconda funzione invocherà (**call back**) il codice passato al momento opportuno (click dell'utente)

N.B. La callback function può accedere alle variabili della funzione contenitrice e alle variabili globali – Per maggiori informazioni vedere javascript [closure](#)

# jQuery Css Selectors

Element selector	<code>\$("p")</code>	seleziona tutti gli elementi p
#id selector	<code>\$("#idValue")</code>	seleziona un unico elemento con attributo id="idValue"
.class selector	<code>\$(".classValue")</code>	seleziona tutti gli elementi con attributo class="classValue"

# jQuery Css Selectors

Syntax	Description
<code>\$("*")</code>	Selects all elements
<code>\$(this)</code>	Selects the current HTML element
<code>\$("p.intro")</code>	Selects all <p> elements with class="intro"
<code>\$("p:first")</code>	Selects the first <p> element
<code>\$("ul li:first")</code>	Selects the first <li> element of the first <ul>
<code>\$("ul li:first-child")</code>	Selects the first <li> element of every <ul>
<code>\$("[href]")</code>	Selects all elements with an href attribute
<code>\$("a[target='_blank']")</code>	Selects all <a> elements with a target attribute value equal to "_blank"
<code>\$("a[target!='_blank']")</code>	Selects all <a> elements with a target attribute value NOT equal to "_blank"
<code>\$(":button")</code>	Selects all <button> elements and <input> elements of type="button"
<code>\$("tr:even")</code>	Selects all even <tr> elements
<code>\$("tr:odd")</code>	Selects all odd <tr> elements

# Confronto metodi DOM e jQuery

DOM method	jQuery equivalent
<code>getElementById("idValue")</code>	<code>\$("#idValue")</code>
<code>getElementsByName("tag")</code>	<code>\$("tag")</code>
<code>getElementsByTagName("tag")</code>	<code>\$("tag")</code>
<code>getElementsByName("somename")</code>	<code>\$("[name='somename']")</code>
<code>querySelector("css selector")</code>	<code>\$("css selector")</code>
<code>querySelectorAll("css selector")</code>	<code>\$("css selector")</code>



# jQuery Events

- Cosa è un evento ?

*Un evento è una qualsiasi azione che un utente può effettuare su di una pagina web*

- Esempi di eventi

Mouse Events	Keyboard Events	Form Events	Document/Window Events
<b>click</b>	keypress	submit	<b>load</b>
dblclick	keydown	change	<b>resize</b>
<b>mouseenter</b>	keyup	<b>focus</b>	<b>scroll</b>
mouseleave		blur	unload

# jQuery Events

- E' importante catturare un evento per avere un comportamento personalizzato rispetto a quello di default del browser
- **Sintassi** per catturare/gestire gli eventi

```
/* Gestire l'evento click su  
tutti gli elementi paragrafo */  
$( "p" ).click(function() {  
    $(this).hide();  
});
```

# jQuery Effects

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

# jQuery Manipolazione

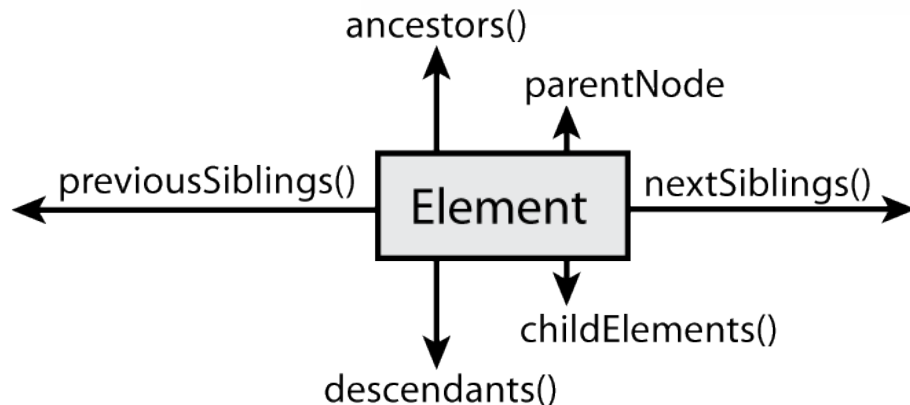
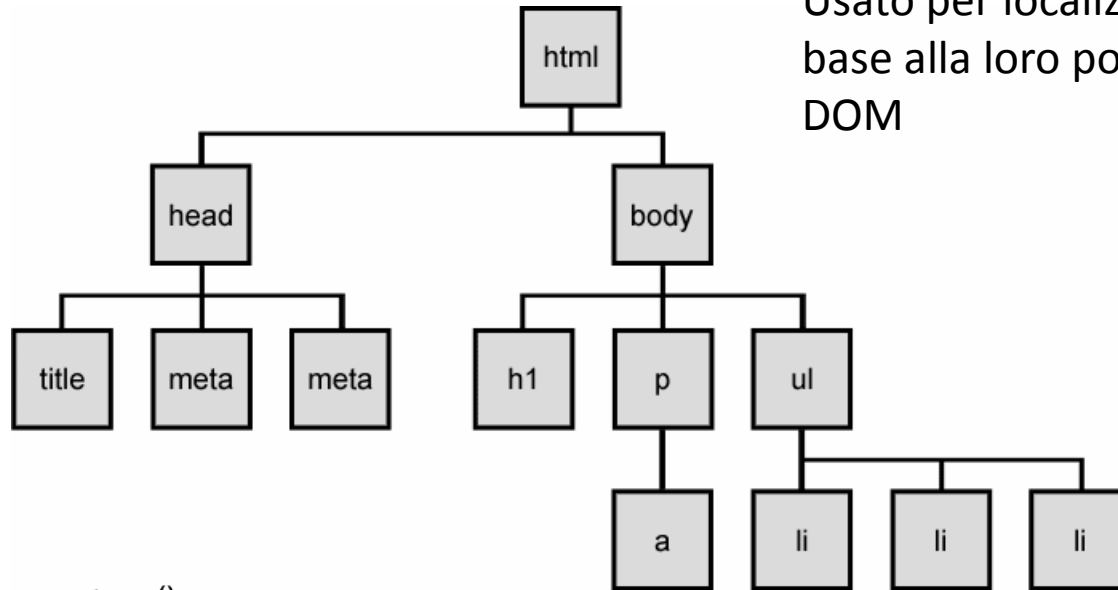
- Leggere/modificare il contenuto degli elementi/attributi selezionati
- Aggiungere/rimuovere elementi/attributi
- Leggere/Modificare il CSS

# jQuery Manipolazione

- `.text()` - Sets or returns the text content of selected elements
- `.html()` - Sets or returns the content of selected elements (including HTML markup)
- `.val()` - Sets or returns the value of form fields
- `.css()` – Sets or returns the value of CSS properties
- `append()`, `remove()`, ...

# jQuery Traversing

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



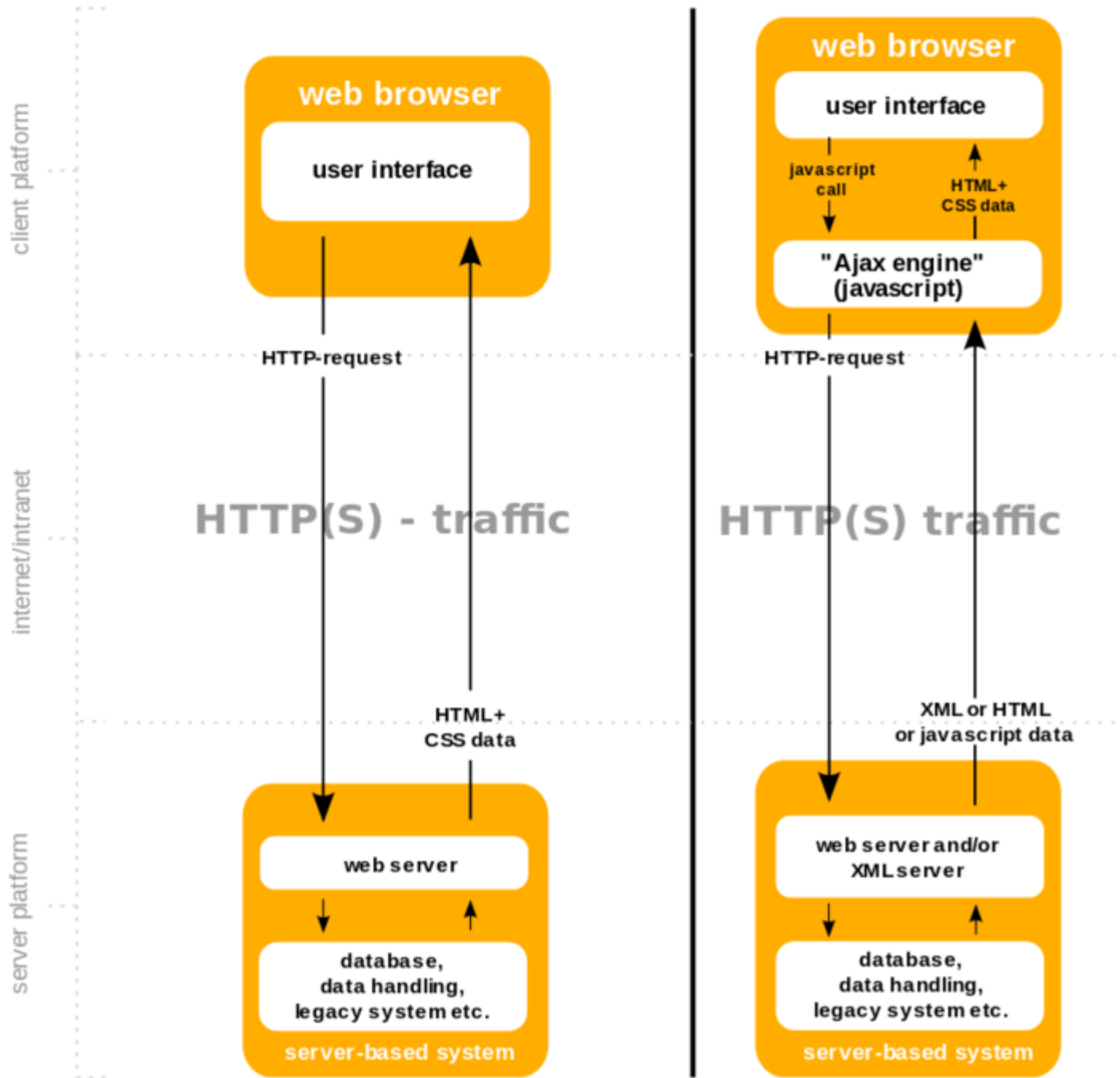
```
$(document).ready(function() {  
    $("a").parent();  
});
```

# Ajax

- 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, ...

Conventional model of a web application

Ajax model of a web application





# Ajax

- 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

## XMLHttpRequest

```
<script>
// 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").innerHTML =
xhr.response;
        else alert('Error: ' + xhr.status + "-" + xhr.readyState);
};

// 3 - Invia la richiesta a tour-pedia
xhr.send(null);
```

## jQuery

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

# 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
  - `$.get(URL,callback);`
- `$.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
  - <http://tour-pedia.org/api/getPlacesStatistics>
- Per vedere meglio i dati json su browser installare [estensioni](#) come jsonView

# Esempio di 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 di 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);
        });
    });
});
```