
Un approccio moderno alle pagine Web

Massimiliano Filacchioni <m.filacchioni AT caspur DOT it>

Articolo disponibile online in formato HTML alla URL <http://www.mfila.it/articoli/un-approccio-moderno-alle-pagine-web/> ed in formato PDF alla URL <http://www.mfila.it/articoli/un-approccio-moderno-alle-pagine-web.pdf>

Copyright (c) 2005, Massimiliano Filacchioni. Materiale rilasciato secondo i termini della licenza Creative Commons Attribution-Sharealike 2.0 [<http://creativecommons.org/licenses/by-sa/2.0/>].

	2005-10-28	
	Diario delle Revisioni	
Revisione 1.3a	2005-10-28	MF
	Corretti vari refusi	
Revisione 1.3	2005-10-27	MF
Cambiato il titolo dell'articolo perché ambiguo, semplificati e chiariti alcuni concetti, aggiunta sezione Conclusioni, ampliata sezione Risorse e corretti vari refusi		
Revisione 1.2a	2005-10-24	MF
Modificato il numero della versione di Firefox successiva alla 1.0 da 1.1 a 1.5 e menzionata la prossima versione di Internet Explorer (IE7)		
Revisione 1.2	2005-08-05	MF
Aggiunta la sezione JavaScript non intrusivo, corretti vari refusi, aggiornate alcune immagini e la sezione Risorse		
Revisione 1.1a	2005-06-13	MF
Modificata la sezione XHTML semantico e microformati ed ulteriormente ampliata la sezione Risorse		
Revisione 1.1	2005-06-10	MF
Aggiunta la sezione XHTML semantico e microformati, ampliata la sezione Risorse ed approfonditi vari argomenti		
Revisione 1.0	2005-06-03	MF
	Versione iniziale	

Estratto

Quest'articolo propone un approccio alla realizzazione delle pagine Web basato sull'esplicita separazione degli aspetti che le caratterizzano. Tale approccio presenta molteplici vantaggi anche in termini di accessibilità e usabilità.

Accessibilità e usabilità: un'introduzione

L'accessibilità e l'usabilità sono due delle caratteristiche più importanti di un sito Web (e quindi delle singole pagine che lo costituiscono). In questo articolo tali caratteristiche verranno spesso menzionate, di conseguenza di seguito ne viene data una breve definizione:

accessibilità capacità di un sito di essere fruibile da qualsiasi utente indipendentemente dalle sue eventuali disabilità e dalla tipologia di browser utilizzato. Tale caratteristica comporta dei benefici non solo per gli utenti portatori di handicap, ma anche per quelli che si trovano nella necessità di utilizzare dispositivi diversi da un personal computer (tipicamente dispositivi mobili quali PDA e smart phone)

usabilità capacità di un sito di essere utile all'utente e di venire incontro alle sue esigenze nel miglior modo possibile. Tale caratteristica è influenzata da un numero di fattori decisamente elevato, alcuni comuni a tutti i siti e gli utenti, altri specifici per le singole tipologie di siti ed utenti

L'accessibilità è un argomento che sta particolarmente a cuore al W3C [1] (World Wide Web Consortium), l'organizzazione internazionale che si occupa della standardizzazione delle tecnologie alla base del Web e negli ultimi anni è stato recepito dalle legislazioni di molti paesi. Il W3C, attraverso l'iniziativa WAI [2] (Web Accessibility Initiative), ha prodotto varie linee guida per promuovere l'accessibilità (rivolte agli autori delle pagine Web, ai produttori di strumenti per la loro creazione, ai produttori dei browser e a quelli dei riproduttori multimediali). Tali linee guida sono alla base della maggior parte delle leggi sull'accessibilità, almeno per ciò che riguarda il Web (esempi di tali leggi sono la cosiddetta Section 508 [3] promossa dal Congresso degli Stati Uniti d'America e la legge Stanca [4] del governo italiano).

Le linee guida per l'accessibilità rivolte agli autori delle pagine Web, WCAG [5] (Web Content Accessibility Guidelines), contengono un insieme di punti di controlli (checkpoint), a cui sono associate tre diverse priorità, che consentono di verificare l'accessibilità delle pagine. In base al soddisfacimento di tali punti di controllo le WCAG definiscono i tre seguenti livelli di accessibilità:

- livello A, che richiede il soddisfacimento di tutti i punti di controllo di priorità 1
- livello AA, che richiede il soddisfacimento di tutti i punti di controllo di priorità 1 e 2
- livello AAA, che richiede il soddisfacimento dei punti di controllo di tutte e tre le priorità

Le suddette linee guida, a causa della loro importanza, sono state tradotte in molte lingue. La traduzione italiana [6] è curata dall'AIB (Associazione Italiana Biblioteche).

Per quanto riguarda l'usabilità, invece, non esistono delle guide di riferimento equivalenti a quelle per l'accessibilità. Soprattutto perché essa è influenzata da un elevatissimo numero di fattori. In letteratura, comunque, esistono innumerevoli pubblicazioni che si occupano dell'argomento, molte delle quali sono il prodotto di studi relativi al comportamento degli utenti.

Accessibilità contro usabilità

Un sito accessibile è per definizione un sito che risulta essere più proficuamente utilizzabile, almeno per una certa categoria di utenti (quella costituita dagli utenti che usano tecnologie assistive, come screen reader, display braille e ingranditori di schermo oppure browser alternativi a quelli grafici per personal computer, che sono i più diffusi). Di conseguenza è facile rendersi conto che l'accessibilità di un sito ne aumenta l'usabilità complessiva.

Il contrario però non è sempre vero, anzi in alcuni casi ci si può trovare nella condizione in cui nel tentativo di aumentare l'usabilità di un sito se ne pregiudichi l'accessibilità. Ciò si verifica soprattutto quando si interviene in modo scriteriato sugli aspetti di presentazione e di comportamento delle pagine Web (aspetti che verranno di seguito approfonditi).

In generale nella realizzazione di un sito è opportuno cercare di ottenere, sempre e comunque, il giusto equilibrio tra accessibilità e usabilità.

Accessibilità e usabilità, in ogni caso, non sono caratteristiche totalmente indipendenti, esse infatti in parte si sovrappongono. Ad esempio la possibilità di selezionare i link e i campi di una form mediante tasti di scelta rapida, migliora sicuramente l'accessibilità delle pagine per gli utenti che soffrono di problemi motori che gli impediscono un uso efficace del mouse, ma anche l'usabilità per qualsiasi altro utente.

Aspetti caratteristici di una pagina Web

Gli aspetti che caratterizzano una pagina Web sono:

<i>il contenuto</i>	che rappresenta l'informazione che si vuole comunicare all'utente, che come è facile intuire è l'aspetto che maggiormente contribuisce a determinare il valore che l'utente attribuisce alla pagina
<i>la struttura</i>	intesa sia come struttura di navigazione, costituita dall'insieme di link che consentono all'utente di raggiungere le varie sezioni di un sito, eventuali form di ricerca e login, ecc., sia come struttura logica del contenuto, cioè l'organizzazione di questo in paragrafi, intestazioni, elenchi puntati e numerati, ecc.
<i>la presentazione</i>	che rappresenta la modalità con cui la struttura viene presentata all'utente. La presentazione può essere di tipo visuale (grafica) e non visuale, per gli utenti che usano tecnologie assistive ed alcuni browser alternativi
<i>il comportamento</i>	che consente di alterare la struttura e la presentazione in risposta ad eventi generati dall'utente (pressione di un tasto del mouse, movimento del puntatore, pressione di un tasto della tastiera, ecc.) o ad eventi e condizioni indipendenti dall'utente

I primi tre aspetti definiscono una pagina come entità statica, l'ultimo, invece, ne determina la dinamicità dal punto di vista del browser (o come si dice in gergo lato client).

Anche se il contenuto ed un minimo di struttura logica sono sempre presenti, molte pagine non prevedono nessuna struttura di navigazione, altre non prevedono nessun comportamento specifico ed altre ancora, per la verità piuttosto rare, non prevedono alcuna presentazione specifica.

Rispetto alla presentazione ed al comportamento, comunque, è importante sottolineare che anche quando una pagina non ne presenta di specifici, essa risulta caratterizzata da una presentazione predefinita determinata dalla tipologia di browser utilizzato per accedervi (i browser grafici e testuali, ad esempio, generalmente separano i paragrafi con uno spazio equivalente ad una linea di testo vuota e quelli grafici rappresentano il testo associato ai link in blu e sottolineato) e da un comportamento predefinito (la selezione di un link, infatti, provoca sempre l'accesso alla risorsa che ne rappresenta la destinazione, mentre l'invio di una form comporta sempre il trasferimento dei dati immessi ad un server Web).

I vari aspetti esaminati coinvolgono figure professionali decisamente diverse tra loro:

- il redattore, che è responsabile del contenuto e della struttura logica
- l'esperto di architettura dell'informazione, che è responsabile dell'organizzazione dell'intero sito e quindi della struttura di navigazione, ma anche della struttura logica di massima delle varie tipologie di pagine in esso presenti
- il designer, che è responsabile della presentazione
- lo sviluppatore, che è responsabile del comportamento

Spesso accade, comunque, soprattutto nei progetti Web di piccole e medie dimensioni, che i ruoli di più figure professionali collassino su una stessa persona (nel caso limite su una sola persona).

Gli standard per le pagine Web

Dal punto di vista tecnologico i vari aspetti caratteristici di una pagina vengono concretizzati utilizzando i seguenti standard:

- l'HTML [7] (HyperText Markup Language) che è il linguaggio di descrizione documentale alla base del Web, utilizzato per il contenuto, la struttura ed eventualmente la presentazione delle pagine (anche se questo suo ultimo utilizzo come vedremo è da evitare)
- l'XHTML [8] (eXtensible HyperText Markup Language) che approfondiremo in seguito
- i fogli di stile CSS [9] (Cascading Style Sheets) utilizzati esclusivamente per controllare la presentazione delle pagine su diverse tipologie di dispositivi (testuali, grafici, vocali, di stampa, ecc.), mediante apposite regole applicate agli elementi che le costituiscono
- il linguaggio di programmazione JavaScript [10] utilizzato per l'implementazione del comportamento
- il DOM [11] (Document Object Model) un modello che rappresenta le pagine Web mediante una struttura ad albero (che evidenzia la relazione di contenimento degli elementi HTML che la costituiscono) e fornisce delle API (Application Programming Interface) per vari linguaggi, tra cui JavaScript, che consentono di navigare e modificare tale albero

Tutti questi standard ad eccezione di JavaScript, sono promossi dal W3C. JavaScript, invece, è stato inizialmente introdotto da Netscape Communications nella versione 2 del suo browser Netscape Navigator, successivamente adottato in diversi altri browser ed infine diventato uno standard promosso dall'ECMA (European Computer Manufacturers Association) con il nome di ECMAScript.

Nonostante i molti sforzi relativi alla standardizzazione delle tecnologie alla base del Web, però, spesso accade che i vari browser (tra cui purtroppo quello più utilizzato dagli utenti, Internet Explorer di Microsoft) implementino i nuovi standard e le nuove versioni di quelli già supportati piuttosto lentamente, ne interpretino alcune direttive in modo non corretto ed introducano delle estensioni proprietarie (alcune volte riassorbite dagli standard ufficiali). Ciò fa sì che per poter realizzare pagine che siano fruibili con tutti i browser, pur rispettando gli standard, sia spesso necessario far ricorso a vari accorgimenti (tecnicamente denominati *hack* o *workaround*) per livellarne le differenze.

Negli ultimi anni, comunque, tale situazione sta lentamente migliorando. Ciò nonostante la maggior parte dei browser grafici attuali prevedono due modalità di funzionamento alternative: una in cui tentano (con diversi livelli di successo) di aderire agli standard ed un'altra, denominata *quirks mode*, in cui per garantire la "compatibilità con il passato" vengono mantenute le interpretazioni errate degli standard introdotte da altri browser o (paradossalmente) da loro versioni precedenti.

Tutto ciò, come è facile intuire, rende molto più complicata del necessario la realizzazione delle pagine Web.

Infine è interessante notare che accanto agli standard promossi dalle varie organizzazioni internazionali, alcune volte si affermano anche standard inizialmente promossi dai singoli produttori di browser e successivamente accettati dai loro diretti concorrenti. Due esempi di tali standard sono costituiti dall'estensione al DOM che consente di realizzare editor HTML visuali all'interno delle pagine Web e dall'oggetto XMLHttpRequest [12] che consente a programmi scritti in JavaScript ed in altri linguaggi di richiedere risorse ai server Web, entrambi introdotti da Microsoft e successivamente implementati (con piccole differenze) da Mozilla Foundation e da altri produttori di browser.

Relazione tra aspetti caratteristici, standard, accessibilità e usabilità

I vari aspetti caratteristici di una pagina Web influenzano l'accessibilità e l'usabilità come segue:

- il contenuto influenza sia l'accessibilità (un contenuto scritto in modo semplice e lineare, infatti, risulta più facilmente fruibile da parte degli utenti con difficoltà cognitive), sia l'usabilità (la qualità del contenuto, infatti, contribuisce a determinare il valore che l'utente attribuisce alla pagina)
- la struttura influenza l'usabilità e soprattutto l'accessibilità (una buona struttura, infatti, rende da sola una pagina altamente accessibile, in quanto consente ai vari browser di fornire per essa la presentazione predefinita più adatta ai loro utilizzatori)
- la presentazione influenza sia l'accessibilità (ad esempio impostando in modo opportuno la dimensione dei caratteri per gli utenti ipovedenti o la giusta combinazione cromatica per gli utenti che hanno problemi nella visione dei colori), sia l'usabilità (ad esempio mettendo in evidenza in vari modi le parti più importanti di una pagina)
- il comportamento influenza esclusivamente l'usabilità

Per quanto riguarda gli standard, invece, è importante sottolineare che il loro "semplice" utilizzo non comporta automaticamente il perseguimento dell'accessibilità e dell'usabilità. Per ottenere risultati significativi in termini di accessibilità e usabilità, infatti, è indispensabile fare ricorso a pratiche di utilizzo degli standard diligenti e consolidate.

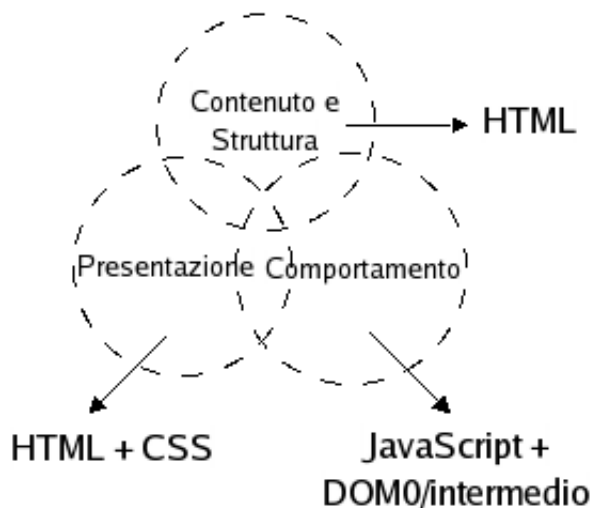
Il vecchio modo di concepire le pagine Web

A partire dall'affermazione del Web come fenomeno di massa, avvenuta a pochi anni dalla sua creazione grazie alla diffusione dei primi browser grafici (Mosaic dell'NCSA, Netscape Navigator della Netscape Communications ed Internet Explorer di Microsoft) e successivamente degli Editor HTML visuali (Netscape Composer, FrontPage di Microsoft, DreamWeaver di Macromedia, ecc.), sono state prodotte centinaia di milioni di pagine Web.

La stragrande maggioranza di queste pagine erano (ed in parte sono tuttora) caratterizzate da uno spiccato mescolamento degli aspetti che le caratterizzano, da una struttura logica minimale (in molti casi quasi inesistente) e da una presentazione predominante (utilizzata anche per colmare la carenza di struttura logica). Si noti, comunque, che il mescolamento del contenuto e della struttura è inevitabile, essendo questi aspetti per loro natura fortemente dipendenti.

La figura che segue schematizza l'architettura di tali pagine mettendo in evidenza gli standard in essa utilizzati:

Figura 1. Il vecchio modo di concepire le pagine Web



Il vecchio modo di concepire le pagine Web presenta i seguenti svantaggi:

- pregiudica l'accessibilità
- non rende indipendenti le figure professionali associate ai vari aspetti delle pagine
- rende difficile la modifica dei singoli aspetti delle pagine successivamente alla loro creazione, complicando, tra le altre cose:
 - l'aggiornamento del contenuto
 - il restyling dei siti
 - il miglioramento dell'usabilità
- costringe a duplicare il contenuto e la struttura qualora se ne volessero realizzare presentazioni alternative
- rende difficile l'estrazione del contenuto da parte di procedure automatiche

Tutto ciò è stato provocato da:

- una scriteriata esasperazione della presentazione grafica da parte dei committenti dei siti e dei designer (che soprattutto in passato non avevano una formazione specifica sul Web)
- le estensioni degli standard da parte dei produttori dei browser
- la lentezza con cui tali produttori hanno implementato (ed in parte continuano ad implementare) i vari standard
- il fatto che gli editor HTML visuali spesso non incoraggiano l'uso appropriato degli standard

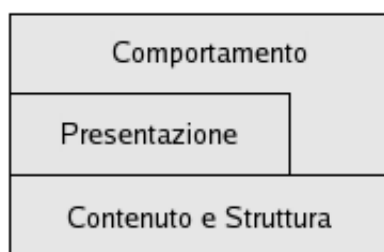
Il nuovo modo di concepire le pagine Web: architettura a strati

Grazie ad un maggiore supporto degli standard da parte dei browser e ad una generale presa di coscienza dei problemi precedentemente esaminati, negli ultimi anni si sta lentamente affermando un nuovo modo di concepire le pagine Web.

Questo nuovo approccio propone un'architettura in cui i vari aspetti caratteristici delle pagine sono nettamente separati (dal punto di vista logico e fisico), in modo tale che ad ognuno di essi corrisponda uno strato (layer) o livello indipendente, fatta eccezione per il contenuto e la struttura ai quali, essendo fortemente dipendenti, corrisponde un unico strato.

La figura che segue schematizza tale architettura:

Figura 2. Architettura a strati



Lo strato di comportamento poggia sia su quello di presentazione, sia su quello di contenuto e struttura in quanto agisce su entrambi, mentre quello di presentazione poggia unicamente su quello di contenuto e struttura. Ogni strato agisce esclusivamente su quello sottostante (quelli sottostanti nel caso del comportamento) ed ignora totalmente la presenza di quelli sovrastanti.

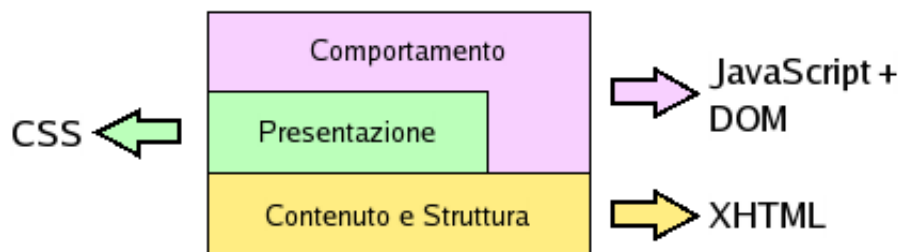
Lo strato di contenuto e struttura deve essere il più **semplice e lineare** possibile ed esaltare il significato (semantica) delle informazioni. Gli hack necessari per livellare le incompatibilità dei varie browser, invece, devono riguardare **unicamente** gli strati di presentazione e comportamento.

Le tecnologie utilizzate dai vari strati sono:

- l'XHTML per lo strato di contenuto e struttura
- i fogli di stile CSS per lo strato di presentazione
- JavaScript e DOM per lo strato di comportamento

La figura che segue ripropone la schematizzazione dell'architettura a strati mettendo in evidenza gli standard in essa utilizzati:

Figura 3. Standard utilizzati nell'architettura a strati



L'architettura a strati se realizzata con criterio risolve tutti i problemi esaminati nella sezione Il vecchio modo di concepire le pagine Web. Essa, inoltre, consente all'utente di personalizzare le pagine a suo piacimento sostituendo integralmente o in parte gli strati di presentazione e di comportamento mediante fogli di stile CSS e script JavaScript creati sulla sua postazione di lavoro (e quindi utilizzabili solo da lui). Tale possibilità, vista da alcuni come sgradevole o addirittura pericolosa, secondo molti rappresenta una vera e propria rivoluzione (positiva) del modo di utilizzare il Web.

Firefox [13] (il browser di Mozilla Foundation, che attualmente è tra quelli più conformi agli standard), ad esempio, consente all'utente, con l'aiuto di un'apposita estensione, denominata URId [14], prima della versione 1.5 e nativamente da questa versione in poi, di assegnare fogli di stile personalizzati ai singoli siti e, mediante un'altra estensione, denominata Greasemonkey [15], di effettuare la stessa operazione con gli script JavaScript.

Due ottimi esempi dell'utilizzo dell'architettura a strati sono rappresentati da S5 [16] e HTML Slidy [17], due sistemi che consentono di realizzare presentazioni fruibili mediante un browser Web. L'autore del primo è Eric Meyer (una delle massime autorità nel campo dei CSS), mentre l'autore del secondo è Dave Raggett del W3C.

Le possibili viste di una pagina Web

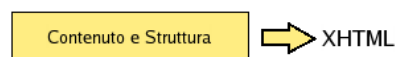
Come accennato in precedenza in una pagina Web non devono necessariamente essere presenti tutti gli aspetti che la caratterizzano e di conseguenza tutti gli strati ad essi corrispondenti.

In base all'insieme di strati presenti si possono distinguere varie viste (view) di una pagina:

vista semplice (plain view)

comprendente solo lo strato di contenuto e struttura che può essere così schematizzata:

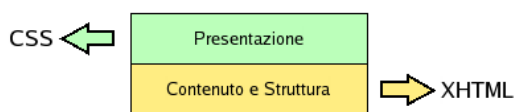
Figura 4. Vista semplice



vista di presentazione (presentation view)

comprendente lo strato di contenuto e struttura e quello di presentazione che può essere così schematizzata:

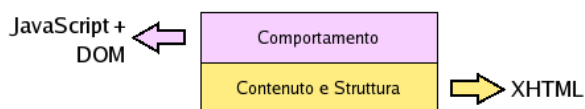
Figura 5. Vista di presentazione



vista semplice con comportamento
(*simple behavior view*)

comprendente lo strato di contenuto e struttura e quello di comportamento che può essere così schematizzata:

Figura 6. Vista semplice con comportamento



vista completa (full view)

comprendente tutti gli strati

Le suddette viste sono state proposte da Peter-Paul Koch nel suo articolo [The Behavior Layer] [18].

Perché l'XHTML

L'HTML è un linguaggio di descrizione documentale che deriva da un altro linguaggio, l'SGML (Standard Generalized Markup Language) promosso dall'ISO (International Organization for Standardization). L'SGML è in realtà un metalinguaggio di descrizione documentale, cioè un linguaggio per la definizione di altri linguaggi di descrizione documentale. I linguaggi definiti mediante l'SGML vengono denominati applicazioni SGML, l'HTML di conseguenza è un'applicazione SGML.

L'SGML è un linguaggio estremamente generico e flessibile, ma proprio a causa di queste sue caratteristiche i documenti descritti con i linguaggi da questo derivati non hanno una struttura facilmente elaborabile in modo automatico, inoltre tali linguaggi non risultano facilmente "estendibili".

Per ovviare a questi problemi il W3C ha creato un nuovo metalinguaggio di descrizione documentale, l'XML [19] (eXtensible Markup Language), che può essere considerato, un po' grossolanamente, una versione ridotta dell'SGML (infatti un qualsiasi documento XML valido è anche un documento SGML valido). L'XML è ormai alla base della maggior parte degli standard Web ed ha avuto un enorme successo anche in molti ambiti applicativi estranei al Web, come linguaggio universale per la rappresentazione di dati strutturati e semi strutturati.

L'XHTML non è altro che la riformulazione dell'HTML in XML. I vantaggi dell'XHTML rispetto all'HTML vanno ricercati nelle differenze tra XML e SGML. Più in dettaglio le pagine XHTML hanno una struttura più facilmente elaborabile di quella delle pagine HTML ed inoltre sono estendibili, in quanto l'XHTML può essere combinato con altri linguaggi derivati dall'XML, come ad esempio MathML [20] (Mathematical Markup Language) che consente la rappresentazione di formule matematiche anche molto complesse e SVG [21] (Scalable Vector Graphics) che consente la descrizione di immagini vettoriali statiche e animate.

Inoltre è importante sottolineare che lo sviluppo dell'HTML è ormai stato interrotto a favore dell'XHTML, che quindi può essere considerato di fatto come la sua naturale evoluzione. Per questo motivo nelle sezioni che seguono il termine HTML verrà utilizzato per riferirsi indifferentemente ad entrambi i linguaggi.

CSS e DOM sono ovviamente utilizzabili anche con l'XHTML (e più in generale con XML).

Di seguito vengono riportate le varie differenze a livello di sintassi tra le pagine XHTML e HTML:

- anche se non è obbligatorio, come prima riga di un documento XHTML andrebbe sempre inserita la così detta dichiarazione XML (mediante la quale si indica la versione XML di riferimento ed altre informazioni come la codifica dei caratteri utilizzata nel documento). Di seguito viene riportato un esempio di tale dichiarazione:

```
<?xml version="1.0" encoding="UTF-8"?>
```

- in XHTML è obbligatoria la dichiarazione del tipo di documento, che specifica la versione del linguaggio utilizzata
- in XHTML nel tag iniziale dell'elemento radice (che è ancora html) è obbligatorio inserire il namespace associato a tale linguaggio (i namespace hanno la funzione di consentire l'utilizzo contemporaneo di più linguaggi derivanti da XML in uno stesso documento). Un esempio di tale tag è il seguente:

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="it" lang="it">
```

- poiché in XML viene fatta distinzione tra le lettere maiuscole e minuscole per i nomi degli elementi, in XHTML questi devono essere sempre scritti in minuscolo
- i tag iniziali e finali sono sempre obbligatori
- gli elementi vuoti, tipo `br` e `img`, devono essere scritti utilizzando entrambi i tag (iniziali e finali), ad esempio:

```
<br></br>
```

oppure utilizzando la sintassi speciale `<nome_elemento />` prevista dall'XML per questo tipo di elementi, ad esempio:

```
<br />
```

si noti che lo spazio tra il nome dell'elemento e il carattere `'` è opzionale, ma per compatibilità con i browser che non supportano XHTML è opportuno inserirlo comunque

- poiché in XML viene fatta distinzione tra le lettere maiuscole e minuscole anche per i nomi degli attributi, in XHTML questi devono essere sempre scritti in minuscolo
- stesso discorso vale per i valori degli attributi che riconoscono un numero limitato di valori (attributi enumerati), ad esempio l'attributo `type` dell'elemento `input`
- i valori degli attributi devono essere sempre racchiusi tra apici singoli o doppi
- gli attributi che in HTML vengono identificati solo con il loro nome (senza quindi avere nessun valore), sono in realtà attributi speciali caratterizzati da un unico valore coincidente col loro nome per i quali viene consentita una rappresentazione abbreviata (minimizzazione degli attributi). Poiché in XML non è consentita una tale rappresentazione in XHTML deve essere sempre specificato un valore per ogni attributo, ad esempio l'attributo `checked` di `input` dovrà essere scritto come segue:

```
checked="checked"
```

- in XHTML è obbligatorio utilizzare il riferimento ad entità `&`; al posto di `&` anche nel valore degli attributi (ad esempio nel valore di `href` quando si specifica una URL contenente più parametri)

L'XHTML 1.0 mette a disposizione le seguenti dichiarazioni di tipo di documento:

- dichiarazione stretta, nella quale sono esclusi la maggior parte degli elementi e degli attributi di presentazione

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

- dichiarazione transitoria, nella quale sono compresi tutti gli elementi e gli attributi di presentazione (la più utilizzata)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

- dichiarazione con supporto ai frame e a tutti gli elementi e gli attributi di presentazione

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

Di seguito, infine, viene riportata la struttura base di una pagina XHTML che utilizza la dichiarazione stretta ed in cui è stata specificata la codifica UTF-8 dei caratteri e la lingua italiana per il contenuto:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="it" lang="it">
<head>
<title>Titolo</title>
. . .
</head>
<body>
. . .
</body>
</html>
```

Separazione del contenuto e della struttura dalla presentazione

Per separare il contenuto e la struttura dalla presentazione si deve:

- evitare di utilizzare nelle pagine HTML elementi di presentazione quali `font`, `center`, ecc. ed attributi di presentazione quali `align`, `border`, `color`, ecc.
- enfatizzare la struttura logica delle pagine mediante elementi quali `p`, `ul`, `li`, ecc. ed attributi quali `alt`, `id`, `class`, ecc.
- inserire le regole CSS in file separati e referenziarli mediante l'elemento `link` utilizzando una sintassi del tipo:

```
<link rel="stylesheet" href="file.css" type="text/css" />
```

è anche possibile specificare la tipologia del dispositivo a cui il foglio di stile si riferisce mediante l'attributo `media` e fornire all'utente un insieme di fogli di stile alternativi (dandogli la possibilità di cambiare la presentazione della pagina) specificando "alternate stylesheet" come valore dell'attributo `rel`

- evitare di conseguenza di specificare tali regole mediante l'elemento `style` o l'omonimo attributo

Isolamento della struttura

Affinché una pagina sia caratterizzata da una buona struttura (e di conseguenza risulti altamente accessibile e facilmente elaborabile in modo automatico) si deve:

- evitare di utilizzare le tabelle per controllare il layout della pagina, esaltando piuttosto la struttura logica delle sue componenti mediante gli elementi `div` (per le componenti che costituiscono dei blocchi, come ad esempio header e footer) e `span` (per le componenti in linea, come ad esempio parole e frasi)
- utilizzare le tabelle unicamente per rappresentare informazioni in formato tabellare
- rappresentare i menu e le barre di navigazione mediante elenchi puntati (elementi `ul` e `li`) essendo tali strutture di navigazione, dal punto di vista logico, degli elenchi di link
- esaltare l'eventuale strutturazione in sezioni del contenuto facendo iniziare ogni sezione con un'intestazione (elementi `h1`, `h2`, `h3`, ecc.), l'elemento `h1` è anche il miglior candidato per rappresentare all'interno del contenuto il titolo della pagina
- utilizzare le intestazioni a partire da `h1` avendo cura di non saltare più di un livello alla volta
- evitare di utilizzare immagini contenenti del testo (tipicamente impiegate per rappresentare bottoni o in sostituzione delle intestazioni), utilizzando al loro posto semplicemente il relativo testo
- evitare di utilizzare immagini per controllare la spaziatura tra le varie componenti della pagina (tali immagini, tipicamente di un solo pixel del colore dello sfondo o trasparenti ed opportunamente ridimensionate, sono generalmente note con il nome di `spacer`)
- evitare di utilizzare i ritorni a capo (elemento `br`) e le sequenze di spazi per controllare la spaziatura tra le varie componenti della pagina
- evitare di simulare gli elenchi puntati mediante la combinazione di immagini e testo unicamente per utilizzare simboli grafici personalizzati e/o per controllarne l'indentazione
- più in generale evitare di utilizzare immagini il cui scopo è puramente decorativo
- utilizzare le immagini solo quando contribuiscono a formare il contenuto (ad esempio foto, diagrammi, screenshot, ecc) e per quelle componenti che si vogliono rendere disponibili in tutte le possibili viste della pagina (ad esempio loghi e banner pubblicitari)
- utilizzare le mappe sensibili in modo oculato, preferendo quelle gestite lato client (elemento `map`) a quelle gestite lato server
- evitare di creare mappe sensibili contenenti un solo link con lo scopo di rendere "attiva" una singola porzione di un'immagine (ad esempio un bottone all'interno di un'immagine che contiene anche parte dello sfondo della pagina)
- preferire per quanto riguarda la formattazione del testo gli elementi che ne mettono in evidenza il significato logico (ad esempio `em` e `strong`, che consentono di esprimere enfasi rispettivamente normale e forte), piuttosto che quelli che agiscono sulla presentazione dei caratteri (ad esempio `i` e `b`, che consentono rispettivamente di ottenere testo in corsivo e grassetto)
- associare in modo esplicito le etichette ai campi delle form mediante l'elemento `label` e raggruppare logicamente i campi mediante l'elemento `fieldset`
- più in generale scegliere di volta in volta quale elemento HTML utilizzare in base al significato logico del contenuto

- utilizzare per i metadati (cioè i dati che non costituiscono il contenuto vero e proprio, ma informazioni aggiuntive su di esso) gli appositi elementi (`title` per il titolo della pagina, `link` per i link che collegano la pagina nella sua interezza ad altre risorse e `meta` per i metadati generici) ed attributi (`lang` per indicare la lingua utilizzata nel testo contenuto in un elemento, `rel` per indicare la categoria di un qualsiasi link, `class` per indicare la tipologia del contenuto di un qualsiasi elemento, ecc.)

Per migliorare l'accessibilità e l'usabilità di una pagina, inoltre, risulta opportuno:

- fornire una breve descrizione del contenuto di ogni tabella mediante l'attributo `summary` dell'elemento `table`
- avere cura di fornire per le immagini e gli altri oggetti non testuali una rappresentazione testuale alternativa (mediante, a seconda dei casi, l'attributo `alt`, `longdesc` o un link ad un documento testuale contenente una descrizione dettagliata)
- associare dei tasti di scelta rapida ai link di navigazione e ai campi delle form mediante l'attributo `accesskey`
- evitare l'uso dei frame in quanto creano problemi ai motori di ricerca, a molti browser alternativi e tecnologie assistive
- considerare l'ipotesi di inserire all'inizio della pagina dei link per facilitare l'accesso alle componenti principali della stessa (link tipo "salta il menu di navigazione", se questo precede il contenuto vero e proprio o "vai al menu di navigazione", in caso contrario), tali link risultano molto utili per gli utilizzatori di browser alternativi e tecnologie assistive

XHTML semantico e microformati

Con il termine Web semantico [22] si indica un modo di concepire il Web che prevede di fornire insieme ai contenuti le informazioni sul loro significato (la semantica appunto) al fine di consentire agli utenti e alle applicazioni di trarre il massimo beneficio dal loro utilizzo.

Il Web semantico è un altro argomento che sta molto a cuore al W3C. Gli standard nati per il suo perseguimento sono molto potenti, ma allo stesso tempo piuttosto complessi. Ciò ne ha ostacolato l'adozione di massa da parte di chi realizza le pagine Web. Attualmente infatti essi vengono utilizzati soprattutto nei progetti di dimensioni medio-grandi.

Accanto a tali standard, comunque, si è affermato un approccio alternativo al Web semantico che fa uso del solo XHTML per evidenziare il significato dei contenuti, mediante pratiche di utilizzo che in parte e senza entrare troppo nei dettagli sono state esaminate nella sezione Isolamento della struttura. Questo approccio è stato denominato XHTML semantico [23]. Esso ovviamente non si vuole sostituire ai suddetti standard, ma essere semplicemente un'alternativa alla portata di chiunque realizzi pagine Web.

L'XHTML semantico è anche alla base dei cosiddetti microformati [24], il cui scopo è quello di rappresentare varie tipologie di informazioni in modo tale che risultino non solo immediatamente fruibili dagli utenti (essendo inserite all'interno di normali pagine Web), ma allo stesso tempo anche elaborabili da procedure automatiche. Tali formati possono essere creati ad hoc oppure essere la trasposizione in XHTML di formati preesistenti (scelta, che come è facile intuire, è consigliabile ogni volta che risulti essere praticabile). Le informazioni rappresentate mediante i microformati possono essere facilmente estratte dalle pagine Web, convertite in altri formati (ad esempio quelli di cui essi rappresentano l'eventuale trasposizione) e quindi riutilizzate.

Presentazione visuale mediante i CSS

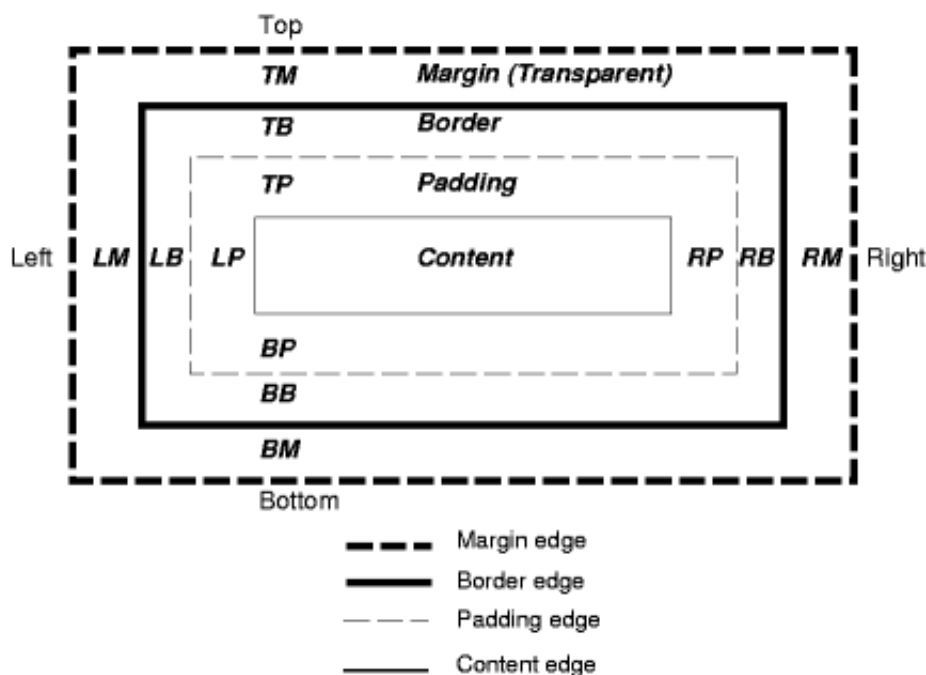
La presentazione visuale mediante i CSS si basa sul cosiddetto box model. Secondo tale modello ad ogni elemento del corpo di una pagina, sia esso un elemento che costituisce un blocco o un elemento

in linea, è associato un box avente le seguenti caratteristiche:

- quattro margini (superiore, destro, inferiore e sinistro) controllabili in modo indipendente, che rappresentano lo spazio che separa il box dell'elemento da quelli degli altri elementi
- un bordo, le cui componenti (superiore, destra, inferiore e sinistra) sono controllabili in modo indipendenti
- un contenuto, costituito a seconda dell'elemento, da testo e/o altri elementi
- uno spazio interno, tra il bordo e il contenuto, denominato padding, le cui componenti (superiore, destra, inferiore e sinistra) sono controllabili in modo indipendenti
- la larghezza complessiva di un box è costituita dalla somma dei margini sinistro e destro, delle componenti laterali del bordo, delle componenti laterali del padding e della larghezza del contenuto; discorso analogo vale per l'altezza

Nella seguente figura, tratta dalle specifiche del W3C, viene schematizzato il box model:

Figura 7. Schematizzazione del box model



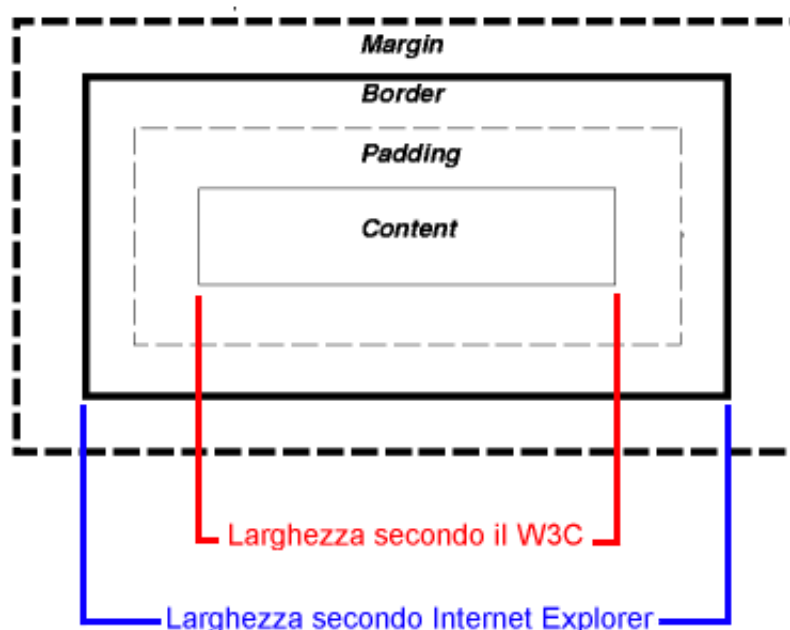
Di ogni box è possibile controllare varie proprietà, tra cui:

- lo spazio che lo separa dagli altri box (intervenendo sui margini)
- il colore, lo spessore e l'aspetto delle varie componenti del bordo, che può essere eliminato ponendo il suo spessore a 0
- lo spazio che separa il contenuto dal bordo (intervenendo sulle componenti del padding)
- lo sfondo dell'area compresa all'interno del bordo (contenuto e padding), che può essere trasparente o di colore uniforme e può contenere un'immagine (fissa, ripetuta orizzontalmente e/o verticalmente e posizionata in un punto preciso dell'area in questione)

- la larghezza e l'altezza (minima e normale) del suo contenuto
- la visualizzazione o meno dell'intero box o del solo contenuto
- il posizionamento che può essere relativo alla sua naturale collocazione all'interno della pagina, assoluto rispetto al box che lo contiene o fisso rispetto all'area di visualizzazione del dispositivo utilizzato (nel caso di un browser grafico ciò consente di realizzare layout simili a quelli tipici dei frame, ma senza questi ultimi)
- la sua profondità in modo da controllare l'ordine di visualizzazione dei box che si sovrappongono
- l'allineamento (a destra o a sinistra) rispetto ai box che lo circondano

Relativamente alla larghezza e all'altezza dei box Internet Explorer per Windows (fino alla versione 5.5 e nella versione 6 se utilizzato in modalità quirks mode) soffre di un problema (bug) nell'interpretazione del box model. Secondo lo standard del W3C, infatti, la larghezza e l'altezza assegnate ad un elemento si riferiscono al solo contenuto del relativo box, mentre Internet Explorer le interpreta comprendendovi anche il bordo e il padding. Ciò fa sì che utilizzando tale browser, agli elementi per cui viene specificata la larghezza e/o l'altezza e almeno una dimensione non nulla per il bordo e il padding corrispondano dei box più stretti e/o corti rispetto a quelli visualizzati dai browser conformi allo standard. Il suddetto bug rispetto alla larghezza degli elementi viene evidenziato dalla seguente figura:

Figura 8. Il box model bug di Internet Explorer rispetto alla larghezza degli elementi



Il box model bug non è comunque l'unico problema che le suddette versioni di Internet Explorer hanno rispetto ai CSS. Ciò, considerando il fatto che Internet Explorer è il browser Web più diffuso (soprattutto grazie al fatto che è integrato in Windows), ostacola in parte ancora oggi lo sfruttamento a pieno dello standard. A quanto sembra, comunque, la versione 7 del browser di Microsoft correggerà molti di questi problemi.

Altro aspetto molto importante dei CSS è rappresentato dal fatto che consentono di utilizzare unità di misura sia assolute, sia relative per i valori delle proprietà che rappresentano delle dimensioni. In genere l'uso di unità di misura relative è sicuramente da preferire perché consente di ottenere una

presentazione che risulta essere indipendente dalle dimensioni dell'area di visualizzazione a disposizione dei browser sui vari dispositivi e di adattarsi al ridimensionamento da parte dell'utente del testo e di tale area (ovviamente quando possibile, ad esempio nel caso si disponga di un personal computer dotato di una scheda video e di un monitor con una risoluzione sufficientemente alta).

Di seguito vengono riportati una minima parte degli effetti di presentazione che è possibile ottenere sfruttando il box model ed altre proprietà visuali dei CSS:

- controllare la dimensione e il posizionamento delle varie componenti delle pagine, facendo addirittura in modo di fornire loro un ordine di presentazione diverso da quello di scrittura
- controllare la spaziatura, il colore di primo piano e lo sfondo di tali componenti
- utilizzare immagini di sfondo anche complesse (senza però esagerare per evitare di influire negativamente sui tempi di caricamento delle pagine) che non dovranno necessariamente ripetersi e che potranno essere collocate in un qualsiasi punto dell'area delimitata dal bordo di un box
- impostare la dimensione, l'allineamento, lo stile, l'interlinea, l'indentazione e i capolinea del testo
- simulare bottoni grafici (se non eccessivamente complessi) mediante del comune testo
- utilizzare una qualsiasi immagine come simbolo associato agli elementi degli elenchi puntati
- controllare l'indentazione degli elementi dei vari tipi di elenchi
- inserire immagini decorative come sfondo fisso (non ripetuto) del box associato ad un elemento (agendo sul padding in modo che il relativo contenuto non le copra)
- fornire presentazioni diverse per i link che fanno parte del contenuto e quelli che fanno parte della struttura di navigazione (mettendo in questo modo in evidenza che svolgono funzioni diverse)
- alterare la presentazione di un qualsiasi elemento quando l'utente posiziona il puntatore del mouse su di esso (effetto noto col nome di rollover ed in passato solitamente ottenuto mediante JavaScript), si noti comunque che Internet Explorer fino alla versione 6 supporta tale funzionalità esclusivamente per i link (elemento a)
- se lo si ritiene opportuno inibire la visualizzazione del contenuto testuale di un elemento (tipicamente un'intestazione) sostituendolo con un'immagine posta come sfondo del relativo box (senza intaccare la struttura e compromettere l'accessibilità della pagina)
- inibire la visualizzazione dei link che consentono l'accesso diretto alle varie componenti della pagina, spesso inutili nel caso di browser grafici

A titolo di esempio della potenza dei CSS vengono riportate tre viste di presentazione alternative della home page del sito CSS Zen Garden [25] (nato per promuovere l'uso dei CSS) corrispondenti a tre diversi fogli di stile:

Figura 9. CSS Zen Garden utilizzando la presentazione di default



Figura 10. CSS Zen Garden utilizzando la presentazione Zenfandel

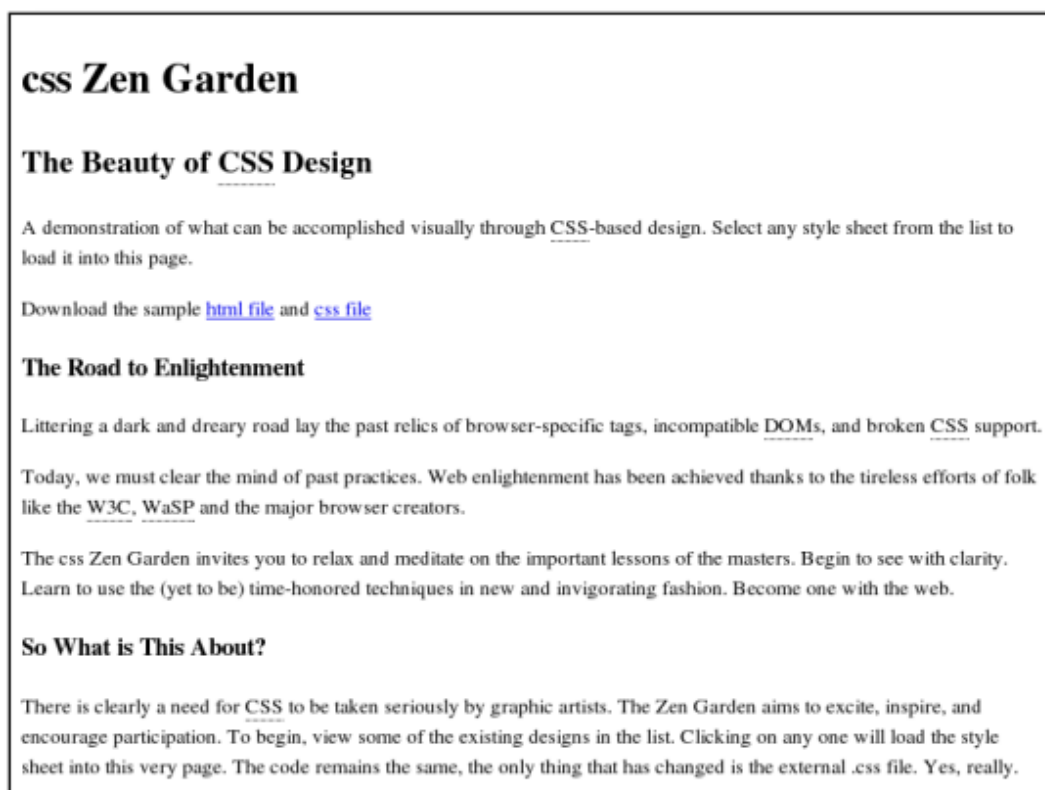


Figura 11. CSS Zen Garden utilizzando la presentazione Bugs



Infine viene riportata la vista semplice di tale pagina:

Figura 12. CSS Zen Garden senza presentazione



Separazione del contenuto e della struttura dal comportamento

Per separare il contenuto e la struttura dal comportamento si deve:

- evitare di scrivere direttamente nelle pagine HTML il codice JavaScript all'interno dell'elemento `script`
- inserire il codice JavaScript in file separati e referenziarli mediante l'attributo `src` dell'elemento `script` utilizzando una sintassi del tipo:

```
<script src="file.js" type="text/javascript"></script>
```

- evitare di associare i gestori degli eventi ai singoli elementi HTML (corrispondenti ai link, ai campi delle form, ecc.) mediante gli attributi `onClick`, `onSubmit`, `onLoad`, ecc.
- effettuare tale associazione in modo programmatico all'interno dei file contenenti il codice JavaScript
- evitare di far generare il contenuto iniziale delle pagine a JavaScript, al fine di fornire agli utenti che non possono o non vogliono utilizzare tale linguaggio pagine che risultino essere comunque fruibili (JavaScript ovviamente potrà essere utilizzato per alterare dinamicamente tale contenuto)

Separazione della presentazione dal comportamento

Per separare la presentazione dal comportamento si deve essenzialmente evitare di impostare la presentazione iniziale delle pagine mediante JavaScript, al fine di fornire agli utenti che non possono o non vogliono utilizzare tale linguaggio pagine che risultino essere comunque fruibili (JavaScript ovviamente potrà essere utilizzato per alterare dinamicamente tale presentazione)

JavaScript non intrusivo

Con il termine JavaScript non intrusivo [26] (Unobtrusive JavaScript) vengono indicate le varie tecniche di utilizzo di tale linguaggio (relativamente al Web) volte alla netta separazione del contenuto e della struttura dal comportamento e della presentazione dal comportamento.

Alcuni utilizzi dello strato di comportamento

Di seguito vengono elencati alcuni tra i possibili utilizzi dello strato di comportamento:

- validare il contenuto delle form prima che venga inviato al server Web (tale validazione comunque non può sostituire quella lato server, che dovrà comunque essere prevista e applicata, per evitare problemi di sicurezza e non solo che potrebbero avere effetti catastrofici)
- alterare la struttura, complicandola più del necessario, senza modificare il codice HTML vero e proprio solo per poter ottenere mediante le regole CSS effetti visivi particolarmente complessi (mantenendo di conseguenza lo strato di contenuto e struttura il più semplice e lineare possibile)
- alterare il contenuto sostituendo alcune porzioni di testo e immagini con componenti multimediali equivalenti (ad esempio filmati Flash)
- compensare i diversi livelli di supporto dei CSS da parte dei browser (simulando, ad esempio, l'effetto di rollover su qualsiasi elemento nel caso di Internet Explorer)
- fornire all'utente la possibilità (mediante form o link) di cambiare foglio di stile qualora la pagina ne fornisse di alternativi (non tutti i browser infatti supportano direttamente tale funzionalità)

Lo strato di comportamento e le applicazioni Ajax

Nelle applicazioni Web tradizionali ogni volta che l'utente segue un link o invia il contenuto di una form si verifica il caricamento di una nuova pagina dal server.

Ajax, che sta per Asynchronous JavaScript + XML, rappresenta un nuovo modo di concepire le applicazioni Web che fa uso di JavaScript e dell'oggetto XMLHttpRequest per consentire allo strato di comportamento di richiedere informazioni ad un server con le quali aggiornare singole porzioni di una pagina. Il termine Ajax è stato inventato da Jesse James Garrett nel suo articolo [Ajax: A New Approach to Web Applications] [27], in cui viene formalizzato un approccio alle applicazioni Web che comunque si stava affermando già da qualche tempo.

I vantaggi delle applicazioni Ajax sono:

- un accesso più veloce alle informazioni
- un minore spreco di banda
- un numero complessivo di pagine minore

- un comportamento più simile alle applicazioni native (non Web)

Ajax può essere utilizzato sia per migliorare applicazioni Web esistenti, sia per creare nuove applicazioni altamente dinamiche che fanno un pesante uso di JavaScript. Il pericolo principale legato ad Ajax è relativo all'accessibilità delle applicazioni da parte dei browser che non supportano le tecnologie necessarie. Per risolvere tale problema si deve cercare di utilizzare Ajax in modo degradabile, cioè facendo sì che le applicazioni che se ne avvalgono siano in grado di comportarsi come quelle tradizionali rispetto ai browser che non supportano le tecnologie necessarie.

Tra gli esempi di applicazioni Ajax disponibili va sicuramente citata Google Suggest [28], che tra l'altro è stata creata prima della pubblicazione del suddetto articolo. Tale applicazione rappresenta un'evoluzione della classica interfaccia del motore di ricerca più famoso del mondo la quale, man mano che l'utente digita caratteri interroga una procedura lato server e visualizza in tempo reale i termini di ricerca più significativi che iniziano con il testo immesso fino a quel momento (insieme al numero di documenti che li soddisfano) in un menu a tendina sotto il campo della form preposto all'inserimento dei termini. La seguente figura mostra un esempio di utilizzo di Google Suggest:

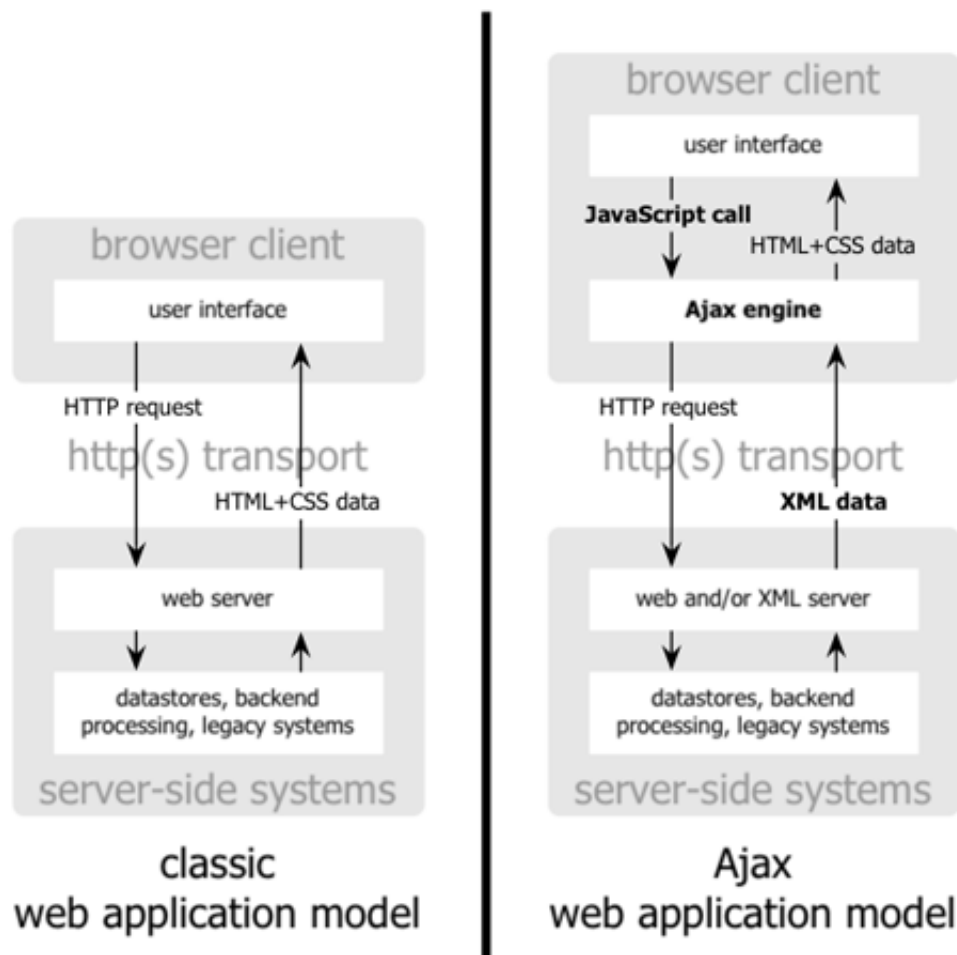
Figura 13. Google Suggest all'opera



Altri esempi di applicazioni Ajax sono: Google Maps [29] e Gmail [30].

La figura che segue, tratta dall'articolo di Jesse James Garrett, mette a confronto le applicazioni Web tradizionali e quelle Ajax:

Figura 14. Applicazioni Web tradizionali e Ajax a confronto



Si noti infine che l'XML viene proposto come formato ideale per lo scambio di dati tra applicazioni Ajax e server Web, ma al suo posto è possibile utilizzare un qualsiasi altro formato.

Validatori ed altri strumenti utili

I validatori sono programmi che data una pagina Web consentono di verificarne l'aderenza agli standard in essa utilizzati, la conformità alle indicazioni contenute nelle linee guida sull'accessibilità (si noti comunque che non tutte le indicazioni sono verificabili in modo automatico), la correttezza dei link in essa presenti e varie altre caratteristiche (come ad esempio il rapporto tra testo e immagini ed altre componenti multimediali, la dimensione delle varie componenti e quindi la velocità di trasferimento alla pagina rispetto a diverse modalità di accesso alla rete, ecc.).

Tali programmi, in grado di operare su pagine presenti sul computer dell'utente o pubblicate in un sito Web, possono essere programmi a se stanti o applicazioni Web (utilizzabili quindi mediante un comune browser). Tra i validatori utilizzabili via Web vanno sicuramente menzionati:

- il Markup Validation Service [31] del W3C, che consente la validazione dell'HTML e dell'XHTML

- il CSS Validator [32] del W3C, che consente la validazione dei CSS
- il Link Checker [33] del W3C, che consente la verifica dei link
- il portale Cynthia Says [34], che consente la validazione dell'accessibilità
- il Web Page Analyzer [35] del sito [WebSiteOptimization.com](http://www.WebSiteOptimization.com), che consente la verifica del livello di ottimizzazione delle pagine pagina

I browser testuali (come Lynx [36] e Links [37]), inoltre, costituiscono uno strumento molto utile per verificare l'accessibilità delle pagine. Le pagine correttamente fruibili mediante tali browser, infatti, in genere lo sono anche utilizzando altri browser alternativi e la maggior parte delle tecnologie assistive.

Per quanto riguarda la verifica dello strato di comportamento, invece, tale operazione richiede l'utilizzo di strumenti tipici dello sviluppo del software (ad esempio i debugger), molti dei quali si integrano con i vari browser.

A causa delle incompatibilità tra i browser grafici più diffusi risulta inoltre piuttosto utile verificare la fruibilità delle pagine almeno con quelli più diffusi.

Sempre più browser, infine, forniscono strumenti addizionali utili a chi realizza pagine Web. Firefox ad esempio ne offre molti, sia direttamente integrati nel browser, sia sotto forma di estensioni (tra le quali meritano sicuramente di essere citate la Web Developer Extension [38] e ColorZilla [39]). Opera [40], un altro browser piuttosto diffuso, invece, fornisce una modalità di visualizzazione che consente di simulare la fruizione delle pagine mediante dispositivi mobili (dotati di un display di dimensioni estremamente ridotte rispetto a quelle di un comune monitor).

Conclusioni

La netta separazione degli aspetti che caratterizzano le pagine Web, attraverso un uso diligente e consolidato degli standard, consente di risolvere molti dei problemi che affliggono le pagine Web attuali, da grandi benefici in termini di accessibilità ed usabilità e facilita l'adozione di eventuali innovazioni tecnologiche future. Tale separazione, però, non è sempre banale, ma risulta comunque semplificata dall'uso di vari strumenti ormai largamente diffusi.

Risorse

- [1] W3C: <http://www.w3.org/> .
- [2] WAI: <http://www.w3.org/WAI/> .
- [3] Section 508: <http://www.section508.gov/> .
- [4] Legge Stanca sull'accessibilità: <http://www.innovazione.gov.it/ita/intervento/accessibilita.shtml> .
- [5] WCAG (versione di riferimento): <http://www.w3.org/TR/WAI-WEBCONTENT/> .
- [6] WCAG (traduzione italiana): <http://www.aib.it/aib/cwai/WAI-trad.htm> .
- [7] HTML: <http://www.w3.org/MarkUp/> .
- [8] XHTML: <http://www.w3.org/MarkUp/> .
- [9] CSS: <http://www.w3.org/Style/CSS/> .
- [10] JavaScript: <http://en.wikipedia.org/wiki/JavaScript> .
- [11] DOM: <http://www.w3.org/DOM/> .

- [12] *XMLHttpRequest*: <http://www.xml.com/pub/a/2005/02/09/xml-http-request.html> .
- [13] *Firefox*: <http://www.mozilla.org/products/firefox/> .
- [14] *URLid*: <http://extensionroom.mozdev.org/more-info/uriid> .
- [15] *Greasemonkey*: <http://greasemonkey.mozdev.org/> .
- [16] *S5*: <http://www.meyerweb.com/eric/tools/s5/> .
- [17] *HTML Slidy*: <http://www.w3.org/2005/03/slideshow.html> .
- [18] *The Behavior Layer*: http://www.digital-web.com/articles/the_behavior_layer/ .
- [19] *XNL*: <http://www.w3.org/XML/> .
- [20] *MathML*: <http://www.w3.org/Math/> .
- [21] *SVG*: <http://www.w3.org/Graphics/SVG/> .
- [22] *Web semantico*: <http://www.w3.org/2001/sw/> .
- [23] *XHTML semantico*: <http://developers.technorati.com/wiki/SemanticXHTMLDesignPrinciples> .
- [24] *Microformati*: <http://microformats.org/> .
- [25] *CSS Zen Garden*: <http://www.csszengarden.com/> .
- [26] *JavaScript non intrusivo*: <http://www.onlinetools.org/articles/unobtrusivejavascript/> .
- [27] *Ajax*: <http://www.adaptivepath.com/publications/essays/archives/000385.php> .
- [28] *Google Suggest*: <http://www.google.com/webhp?hl=en&complete=1> .
- [29] *Google Maps*: <http://maps.google.com/> .
- [30] *Gmail*: <http://www.gmail.com/> .
- [31] *Markup Validation Service*: <http://validator.w3.org/> .
- [32] *CSS Validator*: <http://jigsaw.w3.org/css-validator/> .
- [33] *Link Checker*: <http://validator.w3.org/checklink> .
- [34] *Cynthia Says*: <http://www.contentquality.com/> .
- [35] *Web Page Analyzer*: <http://www.websiteoptimization.com/services/analyze/index.html> .
- [36] *Lynx*: <http://lynx.browser.org/> .
- [37] *Links*: <http://links.sourceforge.net/> .
- [38] *Web Developer Extension*: <http://chrispederick.com/work/firefox/webdeveloper/> .
- [39] *ColorZilla*: <http://www.iosart.com/firefox/colorzilla/> .
- [40] *Opera*: <http://www.opera.com/> .