Capitolo 1

Il layer strutturale: HTML5

Il World Wide Web è diventato oggi, più che in passato, uno spazio virtuale dove le persone vogliono fare qualsiasi cosa: shopping sicuro, leggere libri, socializzare con altre persone, guardare video e ascoltare musica, giocare con videogame di una certa complessità, editare documenti e così via, un lungo elenco che potrebbe estendersi fino a dove arriva l'immaginazione.

Chiaramente, questa crescente e progressiva domanda di nuove feature, impone che sia i progettisti degli standard web sia gli sviluppatori delle piattaforme hardware/software che devono, poi, implementarle, siano al passo con i tempi rendendo utilizzabile e pratico ciò che è scritto in "freddi" documenti di specifiche tecniche.

Nel *mare magnum* delle tecnologie web esistenti o proposte, allo stato attuale il linguaggio di markup denominato HTML5 rappresenta quella emersa con maggior prepotenza e forza, entrando nelle vite, sia di semplici utenti che navigano in Internet sia di sviluppatori che progettano le pagine web, diventando, di fatto, lo strumento fondamentale attraverso il quale creare siti complessi e altamente interattivi.

Nel momento in cui scriviamo, tuttavia, HTML5, visto sia come mero linguaggio strutturale per la creazione delle pagine web sia come piattaforma che fornisce delle moderne API per lo sviluppo, si trova in una fase definita *Draft Standard*, ossia le sue specifiche non sono definitive e immodificabili ma, anzi, nel tempo potrebbero subire cambiamenti, aggiunte o addirittura eliminazioni di talune parti.

In questo capitolo

- **Breve storia della nascita di HTML5**
- **Elementi di base di una pagina web**
- **Inserimento e formattazione di base del testo**
- **Generazione di liste**
- **Utilizzo delle tabelle**
- **Navigazione tra i contenuti delle pagine web**
- **Visualizzazione di immagini e creazione di mappe**
- **Creazione di form e controlli utente**
- **Elementi per la strutturazione di una pagina HTML**
- **Elementi semantici per il testo**
- **Miscellanea di elementi**

In ogni caso, la situazione descritta non deve preoccupare chi desidera imparare tale tecnologia, perché essa è già parte integrante dei moderni browser, cioè la maggior parte delle caratteristiche è già stata in essi implementata.
In più, si può essere certi che le eventuali aggiunte o modifiche, man mano che verranno inserite nella specifica, saranno, in tempi relativamente brevi, subito prese in considerazione dai produttori dei principali browser, che anzi "lotteranno" tra loro per chi le avrà completamente implementate.
In definitiva, HTML5, e tutto l'ecosistema di API che vi ruota attorno, è una grossa opportunità che, sia i vendor dei browser sia gli sviluppatori di applicazioni web non si devono lasciar sfuggire, perché tale linguaggio rappresenta il presente e, certamente, il futuro del World Wide Web.

Breve storia della nascita di HTML5

HTML5, come oggi lo conosciamo e come poi vedremo meglio in seguito, è frutto di una gestazione che trae le sue origini, anche se per certi versi in modo indiretto, dal lavoro pionieristico iniziato nei lontani anni Novanta da un ricercatore di nome Tim Berners-Lee.

> **NOTA**
>
> Tim Berners-Lee è uno scienziato e informatico inglese nato a Londra l'8 giugno del 1955 che, insieme all'informatico belga, Robert Cailliau, ha inventato il World Wide Web. Nel 1980, al CERN, progettò un sistema software chiamato ENQUIRE, che si avvaleva degli ipertesti e che aveva come obiettivo quello di consentire, in modo standard e con estrema facilità, a tutti i ricercatori di scambiarsi delle informazioni. Nel 1989 adattò tale originaria idea in modo che potesse applicarsi a Internet, progettando il primo browser e server web che ne permettesse il funzionamento. Nasceva così il Web come noi tutti lo conosciamo.

Berners-Lee creò in quel periodo un linguaggio definito HTML (*HyperText Markup Language*) mediante il quale era possibile, principalmente, formattare dei documenti testuali con degli speciali marcatori che consentivano di creare dei collegamenti tra testi, che potevano risiedere anche su server localizzati in posizioni differenti dal PC dove si stava utilizzando la pagina principale, permettendo dunque di compiere una sorta di "navigazione" tra le informazioni che essi rappresentavano.
Successivamente, ne sottopose una bozza di specifiche all'importante istituto che si occupava (e si occupa tutt'ora) di creare gli standard di Internet, denominato IETF (*Internet Engineering Task Force*), dove nel 1994 fu creato un gruppo di lavoro (HTMLWG, *HTML Working Group*) che, a partire da quelle specifiche creò poi il relativo documento numerato come versione 2 di HTML.
In quello stesso anno, Tim Berners-Lee fondò un consorzio che doveva esplicitamente occuparsi della standardizzazione delle tecnologie web, denominato W3C (*World Wide Web Consortium*), che, tra le altre cose, causò nel 1996 lo scioglimento del gruppo HTMLWG, inserito nell'organico del W3C.
Nel 1997 il consorzio redasse e pubblicò un'altra specifica di HTML, che aveva come numero di versione il 3.2, per poi aggiornarla nuovamente alla fine dello stesso anno con la versione 4.0.

Arriviamo al 1998, anno in cui il W3C pubblicò una bozza di specifiche per la formulazione di un linguaggio che nel 1999 denominò XHTML 1.0. Nel 2000 divenne una raccomandazione e nel 2001 fu pubblicata la versione 1.1. XHTML differiva da HTML perché basato sul linguaggio XML (*Extensible Markup Language*) e non, come il primo, sul linguaggio SGML (*Standard Generalized Markup Language*).
Nel 1999, inoltre, lo stesso consorzio pubblicò la versione 4.01 di HTML e anche la bozza di una specifica, denominata XHTML Extended Forms, rinominata poi XForms 1.0, con cui intendeva riprogettare i web form e diventata raccomandazione nel 2003.
Dopo un po' di anni, siamo nel giugno del 2004, il W3C organizzò un convegno dal titolo "The W3C Workshop on Web Applications and Compound Documents", con lo scopo di raccogliere idee e suggerimenti sul futuro dello sviluppo web.
A tale convegno parteciparono tutti i principali protagonisti del mondo legato all'implementazione delle tecnologie del Web; tra le tante indicazioni e proposte di carattere generale, furono ritenute interessanti le seguenti, manifestate dai membri di Mozilla Foundation e Opera Software.

- Le specifiche e le caratteristiche di una tecnologia devono sempre essere aperte, visibili e pubblicamente commentabili.
- Le caratteristiche di una tecnologia che una specifica intende descrivere devono essere il meno astratte possibili, ossia devono essere descritte avvalendosi di pratici casi di uso.
- Se si progetta una nuova specifica la stessa deve avere una piena compatibilità con le tecnologie che intende modificare o aggiornare, in modo che sia gli sviluppatori web sia gli implementatori dei browser non vengano, improvvisamente e radicalmente, "investiti" da una nuova tecnologia che renda obsoleto e non funzionante tutto il loro lavoro pregresso.
- Se un documento contiene degli errori di authoring, gli stessi non dovrebbero interrompere il normale flusso di esecuzione della pagina web così come non dovrebbero essere visualizzati.
- Vi dovrebbe essere una gestione degli errori progettata con maggiore correttezza e con un livello di dettaglio analitico.
- Bisogna cercare di definire degli elementi di markup che siano in grado, dichiarativamente, di sostituire quello che, programmaticamente, un linguaggio di scripting, in equivalenza, è in grado di fare.
- Bisogna privilegiare la creazione di profili indipendenti dal dispositivo in uso.

Al termine degli interventi si decise di effettuare una votazione per decidere se era giusto intraprendere una nuova strada per lo sviluppo dei futuri standard web seguendo le indicazioni che erano state formulate.
Purtroppo, per i proponenti, vi fu una sconfitta (11 voti contro e 8 a favore), che però non li demoralizzò affatto ma, anzi, li indusse a unirsi e a creare un nuovo gruppo di lavoro separato dal W3C che denominarono WHATWG (*Web Hypertext Application Technology Working Group*).
Tra le motivazioni che addussero contro il consorzio e che causarono una rottura con esso, è interessante citare le seguenti, ossia che il W3C:

- era sempre stato distante dalle reali esigenze degli sviluppatori web avendo sempre proposto standard teorici e astratti e, dunque, poco pratici;

- non si era mai impegnato più di tanto a creare o ad aggiornare gli standard in modo incrementale in modo che gli "attori" dello sviluppo del web potessero avere il tempo di adeguarsi di conseguenza e in modo indolore;
- non aveva più aggiornato la specifica su HTML dal lontano 1999, anno in cui uscì la versione 4.01;
- non aveva tenuto in considerazione il fattore di compatibilità quando aveva progettato il linguaggio XHTML e XForms. Per esempio, se si desiderava progettare con rigore delle pagine web aderenti a XHTML si doveva, primariamente, cambiare il MIME Type da `text/html`, proprio di HTML, a `application/xhtml+xml` e scrivere i relativi elementi di markup secondo le nuove rigide regole sintattiche imposte da XHTML. Come conseguenza di ciò, se si scrivevano delle pagine XHTML con degli errori, l'elaborazione delle stesse sarebbe stata interrotta, e tale comportamento era, a loro avviso, ritenuto draconiano (*draconian error handling*), laddove il comportamento di HTML rispetto agli errori erra flessibile e senza avvisi.

NOTA

Il termine *errore draconiano* trae la sua semantica dal legislatore dell'antica Grecia Dracone che, all'epoca in cui visse, scrisse un codice di norme notevolmente rigido e duro.

Nel 2004 nacque, dunque, il citato gruppo WHATWG, fondato da alcuni membri di Mozilla Foundation, Apple e Opera Software, che aveva come obiettivo o *mission* principale quella di sviluppare e promuovere nuove tecnologie che potevano migliorare e far avanzare il mondo del Web.
Esso iniziò subito a lavorare su delle specifiche denominate Web Applications 1.0 e Web Forms 2.0, che avevano come scopo quello di aggiornare in maniera significativa l'attuale HTML ma in modo incrementale e compatibile.
Nel 2006, il padre del primigenio HTML, il già citato Tim Berners-Lee, si rese conto dell'importanza del lavoro che si stava svolgendo in seno al WHATWG e propose l'istituzione di un altro gruppo di lavoro in seno al W3C (che nacque ufficialmente nel 2007), che doveva con essi collaborare per la creazione degli standard web del futuro.
In seguito alla nascita di tale gruppo di lavoro, e come primo atto di collaborazione con il gruppo WHATWG, si decise di cambiare il nome della specifica da Web Applications 1.0 in HTML5.
Nel 2009, dato che HTML5 era divenuto la principale e più importante specifica che trattava del futuro dello sviluppo web, si decise che la specifica XHTML 2.0 dovesse cessare.
Oggi, anno 2012, se vogliamo effettuare un rigido paragone con i livelli delle specifiche come statuite dal W3C, nonostante la specifica di HTML5 sia ancora in uno stato di *Working Draft* (ma dovrebbe presto raggiungere lo stato di *Candidate Recommendation* per divenire una *Recommendation* nel 2022), i principali vendor di browser stanno continuando senza sosta a implementare tutte le sue eccitanti e nuove caratteristiche perché tale linguaggio rappresenta un'innovazione davvero importante per lo sviluppo delle moderne applicazioni web.

NOTA

Quando parla di HTML5, il gruppo WHATWG preferisce riferirsi a esso semplicemente come *HTML*, senza attribuirgli un numero di versione, e lo considera semplicemente come "l'ultimo lavoro su HTML". In effetti, nella specifica è precisato che il termine *HTML5* altro non è che una *buzzword* che si riferisce alle moderne tecnologie di sviluppo per il Web, la maggior parte delle quali sono sviluppate dal WHATWG, in autonomia o in collaborazione con il W3C e lo IETF.

Le specifiche secondo il W3C

Per meglio orientarsi nell'oscuro labirinto dei termini addottati per le specifiche dal W3C, può essere utile conoscere i livelli, o stati di avanzamento, di una specifica.

- *Working Draft* (WD): stato in cui la specifica è disponibile pubblicamente a tutti per una sua analisi. In questa fase è consentito effettuare modifiche alle sue caratteristiche.
- *Last Call Working Draft*: stato in cui si specifica una *deadline* per eventuali commenti, si sollecitano recensioni pubbliche e dai working group dipendenti.
- *Candidate Recommendation* (CR): stato in cui le caratteristiche di maggior rilievo della specifica non possono subire cambiamenti, a meno che non vi siano particolari e importanti richieste da parte degli implementatori.
- *Proposed Recommendation* (PR): stato in cui la specifica è sottoposta all'approvazione del W3C Advisory Council. In questa fase è difficile che siano accettati ed effettuati dei cambiamenti alle sue caratteristiche.
- *W3C Recommendation* (REC): stato in cui la specifica è uno standard e può essere impiegato nei reali sistemi di produzione.

ATTENZIONE

Allo stato attuale, il gruppo di lavoro WHATWG non ritiene più importante considerare la specifica su HTML5 come un'unica entità sottoposta a un iter procedurale che ne sancisce il livello di maturità. Piuttosto, invece, la considera in termini pratici come scomposta in sezioni, dove ciascuna sezione può avere un differente livello di maturità e utilizzo. In pratica, per il gruppo di lavoro la fatidica data del 2022 in cui dovrà essere rilasciata la specifica in modo completo non ha più alcuna rilevanza perché per esso è importante concentrarsi e rendere chiaro cos'è già da oggi possibile utilizzare per le proprie applicazioni web, adottando quindi un modello di gestione della specifica non rigido, ma a sezioni e incrementale.

Elementi di base di una pagina web

Una pagina web è composta fondamentalmente da marcatori, o *tag*, che stabiliscono delle regole su come debba essere strutturato e formattato il relativo documento che intendono descrivere.

Questi tag possono essere di tipo *two-sided* (detti anche *normal elements*), espressi mediante l'indicazione di un tag di apertura e chiusura nella forma `<elem_name></elem_name>`, oppure di tipo *one-sided* (detti anche *void* o *empty elements*), espressi mediante l'indicazione del

solo tag di apertura nella forma <elem_name> o <elem_name />. Inoltre essi possono avere degli attributi, ossia del testo scritto al loro interno in una forma chiave/valore, che permettono di specificare opzioni o dettagli supplementari.

TERMINOLOGIA

Spesso, i termini *tag* ed *elemento HTML* si usano in modo intercambiabile. In realtà tra di essi vi è una differenza che può essere sintetizzata nel seguente modo: un elemento è quel nome informativo che indica la semantica di un oggetto, mentre un tag è l'elemento racchiuso tra le parentesi angolari <>. Per esempio, p è un elemento HTML che descrive un generico paragrafo, mentre ne rappresenta il relativo tag di apertura, così come ne rappresenta il relativo tag di chiusura. Inoltre, in via più estensiva e generalizzata, possiamo anche indicare un elemento HTML come un'entità composta da tre parti: un tag di apertura ( ), il contenuto (Ciao a tutti) e un tag di chiusura ( ).

Vediamo allora un primo esempio di definizione di una semplice pagina HTML, scritta secondo la sintassi propria di HTML5, che renderizza a video dei comuni paragrafi.

Listato 1.1 File PrimaPagina.html.

```
<!DOCTYPE html>
<html>
    <head>
        <title>La nostra prima pagina in HTML5</title>
        <meta name="keywords" content="web, html5, javascript, css" charset="utf-8">
    </head>
    <body>
        <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, <br /> sed do
        eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
        <p>Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut
        aliquip ex ea commodo consequat.</p>
        <p>Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur,
        adipisci velit...</p>
        <p>Non c'è nessuno a cui piaccia il dolore in sé, che lo ricerchi e che voglia
        riceverlo, semplicemente perché è dolore...</p>
    </body>
</html>
```

NOTA

Le regole per la scrittura dei tag in HTML5 sono molto flessibili. Infatti, per gli elementi vuoti, ossia per quelli che non possono avere un contenuto, come per esempio br, la presenza del carattere slash di chiusura (/) non è obbligatoria (si può scrivere
 o
). Inoltre, i tag e gli attributi possono essere scritti con caratteri in maiuscolo, minuscolo o addirittura in modo misto; per alcuni elementi i tag di apertura o chiusura possono essere omessi; per gli attributi booleani (per esempio checked) non è necessario definire il valore relativo, che è indicato automaticamente dalla presenza/assenza dell'attributo (checked = "checked" è ridondante), mentre per gli altri il valore può essere scritto tra singoli apici, doppi apici o senza apici. La flessibilità di HTML5 non dovrebbe però essere "abusata". È buona norma darsi delle regole e cercare di essere il più possibile rigorosi, come se si lavorasse su documenti XHTML; in questo modo si migliorano la leggibilità, la scalabilità e la semantica dei documenti prodotti.

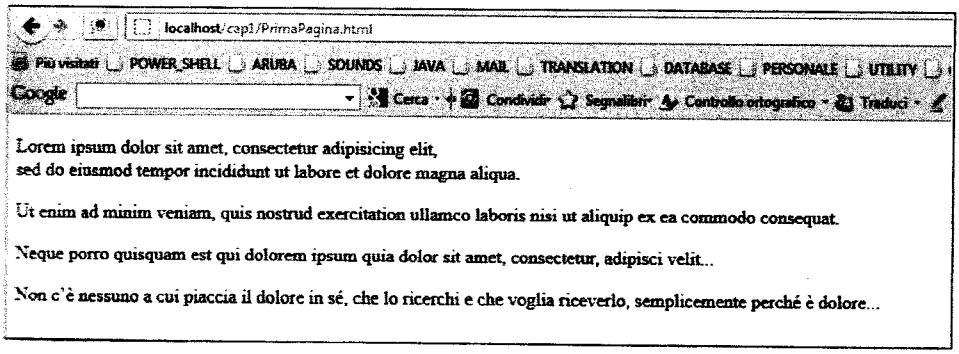

Figura 1.1 Output del file PrimaPagina.html (Firefox).

Nel Listato 1.1 notiamo la presenza, come primo elemento, del tag `<!DOCTYPE html>`, che rappresenta un'istruzione che dà al browser un'indicazione del tipo di codice di cui dovrà eseguire il parsing. In pratica, un documento HTML, tramite l'elemento `DOCTYPE`, dice al browser che il suo codice è conforme allo standard ivi indicato, e nel nostro caso esso asserisce che è conforme a HTML5.

Un approfondimento del tag DOCTYPE

Il `DOCTYPE` o *Document Type Declaration*, è un elemento HTML che deve essere posizionato come prima istruzione nell'ambito del documento che descrive e non deve avere altre istruzioni che lo precedono (anche righe vuote), altrimenti alcuni browser potrebbero trattare la pagina come se non ne avesse alcuno. Come già detto, il suo scopo principale è quello di indicare il linguaggio di markup che un browser deve utilizzare e le regole, stabilite in un apposito DTD (*Document Type Definition*), che deve eseguire per una corretta renderizzazione del documento. Se un documento non ha la definizione di un `DOCTYPE`, il browser assume che il codice è scritto in un modo detto *quirks mode*, mentre nel caso contrario assume che il codice è scritto in un modo detto *standards mode*. Di seguito riportiamo alcuni dei più comuni `DOCTYPE` utilizzabili.

- `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">`, per HTML 4.01 Strict.
- `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">`, per HTML 4.01 Transitional.
- `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">`, per HTML 4.01 Frameset.
- `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">`, per XHTML 1.0 Strict.
- `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">`, per XHTML 1.0 Transitional.
- `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">`, per XHTML 1.0 Frameset.
- `<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">`, per XHTML 1.1.
- `<!DOCTYPE html>`, per HTML5.

Abbiamo poi il tag <html>, il principale contenitore di un documento HTML, che è indicato anche con il temine di *elemento root* (radice). Al suo interno si inseriscono tutti gli altri elementi HTML. Segue il tag <head>, che rappresenta una sorta di intestazione (*header section*): qui si inseriscono dei tag che non sono interpretati come elementi che visualizzano o formattano qualcosa direttamente nell'area di renderizzazione del browser. Si tratta di tag come i seguenti.

- <title>: indica il titolo della pagina e ha un grande valore semantico. Viene visualizzato nella barra del titolo dei browser.
- <meta>: permette, tra le altre cose, di indicare delle parole inerenti alla tipologia o alle caratteristiche del sito web che si è progettato al fine di poterle utilizzare per l'indicizzazione dai motori di ricerca e di specificare la codifica di caratteri utilizzata nei propri documenti HTML.

> **NOTA**
>
> L'elemento meta viene utilizzato per fornire dei metadati per il documento HTML, principalmente attraverso l'attributo name, utilizzato per indicare il nome del metadato, e può avere come valori author (fornisce il nome dell'autore della pagina), description (fornisce una descrizione della pagina); keywords (fornisce una serie di parole rilevanti per la pagina) e così via. Abbiamo poi l'attributo content, che è utilizzato per indicare il valore del corrispondente attributo name.

- <script>: consente di inserire del codice di scripting nella pagina web.
- <link>: consente di collegare al documento altre risorse a esso esterne.

Infine, dopo il tag </head> abbiamo il tag <body>, che rappresenta la sezione più importante di una pagina web, poiché al suo interno sono definiti tutti quei tag che descrivono il contenuto da visualizzare nel browser. Nel nostro esempio, usiamo il tag <p>, che indica di formattare il testo in esso indicato come un paragrafo (da notare come il browser abbia posto degli spazi tra i vari paragrafi), e il tag
, che indica di spezzare una riga.

Inserimento e formattazione di base del testo

Il testo, visto come una mera sequenza di caratteri che hanno un qualche significato, è uno dei contenuti principali delle pagine web.
Esso viene inserito nei documenti HTML attraverso dei tag che hanno un preciso valore semantico e visuale.

> **NOTA**
>
> Lo scopo di questo capitolo è guidare il lettore nella costruzione di un documento HTML, partendo dalle basi. Nel Capitolo 2 vedremo invece la formattazione dei documenti HTML attraverso i CSS. In particolare nel paragrafo "Il box model" vedremo una ulteriore distinzione dei tag HTML dal punto di vista dei CSS.

Sinora, tra i vari tag utilizzabili per la suddetta finalità, abbiamo incontrato solamente il tag <p> che permette di definire un paragrafo il cui testo viene di norma visualizzato con l'aggiunta di una spaziatura superiore e inferiore.

Oltre a tale tag ne esistono altri, come quelli:

- per la definizione dei titoli (*heading*), dal più importante, visualizzato in grassetto con il font più grande (`<h1>`), al meno importante, visualizzato in grassetto con font più piccolo (`<h6>`), passando per titoli intermedi visualizzati con font di dimensioni progressivamente inferiori (`<h2>`, `<h3>`, `<h4>` e `<h5>`);
- per la definizione di testo in grassetto o con un'enfasi forte (`<b>` per *bold* o `<strong>` per *strongly emphasized*);
- per la definizione di testo in corsivo o con enfasi (`<i>` per *italico* o `<em>` per *emphasized*);
- per la definizione di testo in font monospaziato (`<kbd>` per *keyboard*, `<code>` e `<samp>` per *sample*);
- per la definizione di testo preformattato (`<pre>` per *preformatted*);
- per la definizione di un apice o un esponente (`<sup>` per *superscript*) o di un pedice (`<sub>` per *subscript*);
- per la definizione di una sezione il cui testo è una citazione (`<blockquote>`).

ATTENZIONE

Nell'elenco dei tag riportato ne abbiamo inseriti alcuni che, di fatto, formattano il testo dando una sorta di regola su come esso debba essere visualizzato. In effetti, li abbiamo qui elencati perché non abbiamo ancora affrontato i CSS (*Cascading Style Sheets*), che rappresentano il linguaggio principe per indicare delle regole di formattazione e visualizzazione degli elementi HTML. In ogni caso, per HTML5 i tag `<b>`, `<i>`, `<em>` e `<strong>` non sono solo di presentazione ma hanno anche uno scopo semantico, ossia danno un "significato" agli elementi che racchiudono o, per dirla in un altro modo, il loro nome riflette lo scopo per cui sono stati progettati. Così, per esempio, l'elemento `i` rappresenta una porzione di testo che indica, in modo descrittivo e significativo, una voce alternativa o uno stato d'animo il cui contenuto è tipicamente presentato in corsivo.

Listato 1.2 File Testo.html.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Tutto sul testo</title>
        <meta name="keywords" content="web, html5, javascript, css" charset="utf-8">
    </head>
    <body>
        <hgroup>
            <h1>Benvenuti nel fantastico mondo della programmazione del web</h1>
            <h3>Un libro per tutti dove imparerete come diventare dei web developer!!!</h3>
        </hgroup>
        <p>&copy; 2012 Pellegrino <b><i>~thp~</i></b> Principe</p>
        <p><strong>N.B.</strong> Ricordiamo di avere sempre a portata di mano la più
        recente specifica di HTML5<sup>1</sup></p>
        <h5><sup>1</sup> http://www.whatwg.org/specs/web-apps/current-work/multipage/ </h5>
        <p>Un pò di codice di esempio in JavaScript:</p>
        <pre>
var j=10;
var data = [];
for(var i=0; i < j; i++)
```