HTML 5

Introduzione

Storia

- HTML5 è il successore di HTML 4 e XHTML 1.0,
- standardizzati dal W3C rispettivamente nel 1999 e 2000
- Nel 2002 il W3C decide che la futura versione 2.0 di XHTML debba rimpiazzare HTML (questo implica la nonretrocompatibilità del futuro linguaggio con HTML 4), ed essere document-oriented e non application-oriented.
- Nel 2004 si forma il consorzio WHATWG (Web Hypertext Application Technology Working Group) in opposizione al W3C e alla sua decisione di abbandonare HTML per XHTML.
- WHATWG rilascia nel 2008 il primo draft del suo HTML5.

Storia

- Nel frattempo il W3C torna sui suoi passi, e finisce per sposare la visione WHATWG. Il progetto XTML 2.0 viene chiuso, e parte il lavoro per la standardizzazione di HTML5.
- Nel 2012 WHATWG rilascia l'"HTML5 living standard", ovvero una specifica di HTML5 che verrà lasciata evolvere continuamente (senza rilascio di versioni specifiche).
- Nel 2014 il W3C rilascia il suo standard HTML5.

HTML5 vs. HTML 4

- Nuove marcature strutturali e «semantiche»
- Altri nuovi tag
- Elementi HTML editabili
- Supporto ai microdati
- Nuovi tag e attributi per le form
- Nuove API

HTML5 vs. HTML 4

- Nuove API create per supportare:
  - Multimedialità
  - Geolocalizzazione
  - Esecuzione asincrona e parallela di script
  - Comunicazioni bidirezionali tra web client e web server
  - Drag and drop
  - Applicazioni web offline
  - Grafica
  - Cronologia della navigazione
  - …

Semantica

- Con semantica di un tag s’intende il significato che l’elemento attribuisce al suo contenuto.
- Con le nuove specifiche di HTML 5 gli attributi stilistici vengono gestiti solamente tramite fogli di stile in modo tale da separare il contenuto di un documento dalla sua visualizzazione.
- **Tutti i contenuti devono essere in un tag**

Doctype

- Il *doctype* è il primo elemento che dovrebbe comparire in un documento HTML.
- La sua funzione è di validare il documento e, per alcuni browser, serve ad abilitare la modalità di interpretazione standard.
- Inserire il *doctype* di HTML 4 non era per niente semplice.
- Fortunatamente con la nuova specifica la sintassi di questo elemento è molto più snella.

Doctype

- *Doctype HTML 5*
  - `<!DOCTYPE html>`
- *Doctype HTML 4.01 strict*
  - `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN «"http://www.w3.org/TR/html4/strict.dtd">`

Codifica dei caratteri

- Nell'intestazione di un documento HTML (all'interno dei tag `<head></head>`) sono definiti i tag `<meta>` tra i quali uno è utilizzato per definire la codifica dei caratteri.
- Anche in questo caso si può rappresentare lo stesso attributo con una sintassi più agile.
- *Doctype HTML 5*

```
<META charset=UTF-8">
```

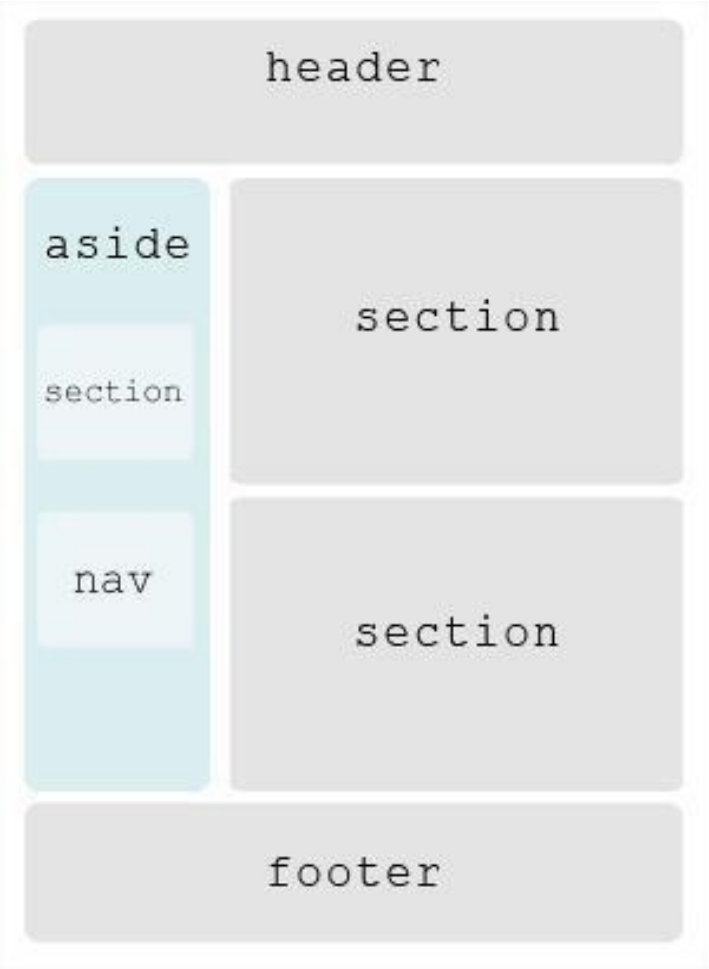
- *Doctype HTML 4.01*

```
<META http-equiv="Content-Type"  content="text/html;
charset=UTF-8">
```

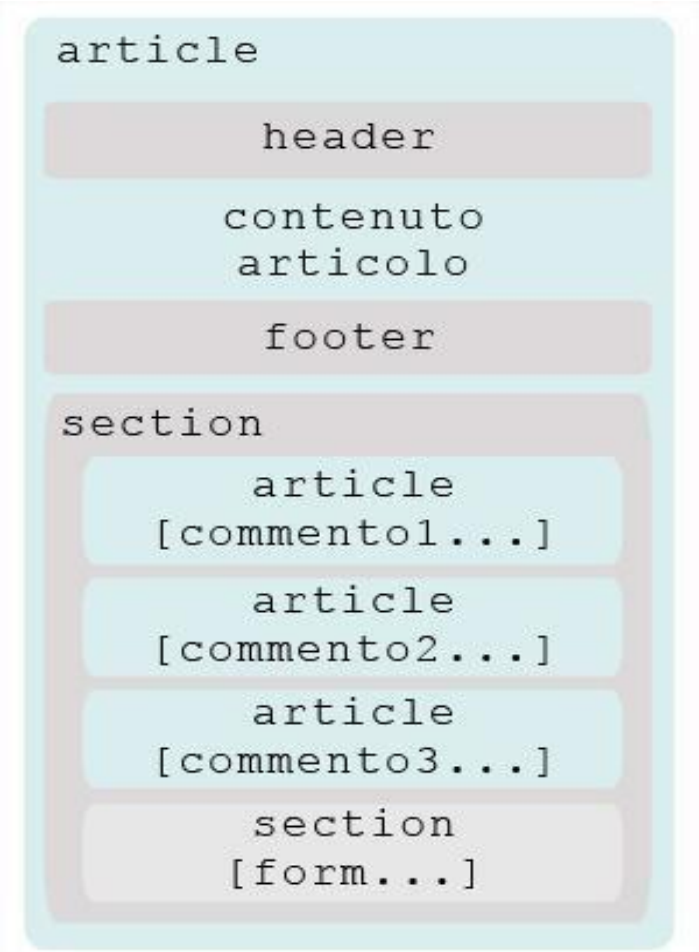
Tag strutturali e semantici

- <header>
- <footer>
- <section>
- <article>
- <nav>
- <aside>
- <hgroup>
- <mark>
- <time>
- <meter>
- <progress>
- <picture>

Esempi di uso dei nuovi tag



Struttura di una pagina



Suddivisione semantica degli articoli

Struttura

```
<!DOCTYPE html>
<html>
        <head>
            <meta charset = "utf-8">
            <title>...</title>
        </head>
        <body>
            <header>...</header>
            <nav>...</nav>
            <article>
                <section>
                </section>
            </article>
            <aside>...</aside>

            <footer>...</footer>
        </body>
</html>
```

Altri tag

- <figure>
- <figcaption>
- <ruby>
- <wbr>
- <command>
- <menu>
- <details>
- <summary>
- <keygen>
- <ouput>

Il tag <script>

- HTML 4

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

- HTML 5 semplifica la sintassi

<script src = "scriptfile.js"></script>

Il tag <link>

- HTML 4

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

- HTML 5 semplifica la sintassi
- <link rel = "stylesheet" href = "stylefile.css">

Supporto ai microdati

- I microdati servono a creare un tagging «semantico» di porzioni del documento
- Sono basati vocabolari che definiscono identificatori di proprietà relative ad un (micro-)dominio di interesse
- si utilizzano gli attributi HTML itemscope, itemtype e itemprop

Esempio: vocabolario Person

Property	Description
name (fn)	Name.
nickname	Nickname.
photo	An image link.
title	The person's title (for example, Financial Manager).
role	The person's role (for example, Accountant).
url	Link to a web page, such as the person's home page.
affiliation (org)	The name of an organization with which the person is associated (for example, an employer). If fn and org have the exact same value, Google will interpret the information as referring to a business or organization, not a person.
friend	Identifies a social relationship between the person described and another person.
contact	Identifies a social relationship between the person described and another person.
acquaintance	Identifies a social relationship between the person described and another person.
address (adr)	The location of the person. Can have the subproperties street-address, locality, region, postal-code, and country-name.

Microdati - esempio

<div **itemscope itemtype**="http://data-vocabulary.org/Person">il mio nome é

<span **itemprop**="name">Athos Ghiggi e il mio acronimo é

<span **itemprop**="nickname">GHA mi trovi sul sito della spai di Locarno

Questo é il link al sito:

Orario SPAI Insegno a Locarno come

<span **itemprop**="role">Docente informatica.

Modificare il contenuto di una pagina

- I seguenti nuovi attributi globali permettono di dichiarare elementi del documento HTML modificabili da utente:
  - Contenteditable
  - contextmenu
  - data-* (attributi definibili da utente)
  - Draggable
  - Hidden
  - spellcheck

Nuove API

- HTML5 aumenta significativamente le API messe a disposizione degli script
- Gli obiettivi sono:
  - In generale, accrescere le possibilità offerte alla programmazione lato client
  - Standardizzare alcuni tipi più comuni di operazioni effettuate lato client
  - Aggiornare le API alle necessità dei media agent più recenti (smartphone, tablet)
  - Gestione di risorse e flussi audio e video

Nuove API

- Accesso off-line alle applicazioni
- Estensione delle capacità di comunicazione, sia verso il web server che verso altre applicazioni
- Esecuzione di azioni in background
- Estensione del concetto di cookie (salvataggio di informazioni sul dispositivo dell'utente)
- Gestione della cronologia della navigazione
- Text editing
- Gestione del «drag and drop»
- Generazione di oggetti grafici 2D/3D
- Gestione di informazioni multimediali generate dall'utente (ad esempio mediante webcam e microfono)

Offline API

- Permettono di salvare copie locali di un insieme di risorse, allo scopo di permettere al browser di eseguire applicazioni anche in modalità offline
- L'elenco delle risorse da salvare è contenuto in un file chiamato manifest (MIME type 'text/cache-manifest')
- L'attributo manifest del tag html permette di dichiarare il file manifest associato al documento HTML
- La cache costituita dalle risorse elencate nel file manifest è gestita da apposite API (associate all'oggetto corrispondente alla proprietà applicationCache dell'oggetto window)

WebStorage API

- Estendono le capacità di memorizzazione dei cookies
- Oggetti localStorage e sessionStorage (accessibili come array associativi)
- Possono memorizzare solo stringhe (testo): per memorizzare oggetti arbitrari occorre serializzarli (ad esempio tramite JSON)

WebSocket API

- Permettono di instaurare una connessione dati bidirezionale tra web client e web server
- La creazione di un nuovo oggetto WebSocket crea una connessione con un server (la cui url va specificata nel costruttore)
- La funzione associata all'event handler onmessage viene eseguita quando dal server arriva un messaggio
- Il metodo send(x) invia il testo x al server

Canvas

- Elemento per dichiarare una zona della pagina su cui è possibile disegnare, tramite nuove API
- Si può disegnare una canvas in un contesto 2D o in un contesto 3D
- Le API per disegnare (in contesto 2D) si dividono in:
  - metodi path (linee, archi,…)
  - metodi modificatori (rotazioni, traslazioni,…)
  - metodo drawImage (disegna un’immagine)
  - metodi per scrivere testo
  - metodi per scrivere singoli pixel

Geolocation API

- Servono a gestire dati geospaziali, tipicamente la posizione (anche se questa è effettivamente disponibile solo su alcuni user agent)
- La proprietà geolocation dell'oggetto navigator ha due metodi per ottenere la posizione corrente:
  - **getCurrentPosition**
  - **watchPosition**
- (il secondo metodo differisce dal primo perché restituisce una nuova posizione ogni volta che questa cambia)

Audio/Video e nuove API

- HTML5 permette la gestione nativa (ovvero senza ricorrere a plug-in del browser) di contenuti multimediali
  - tag **<video>**: permette di inserire un contenuto video
  - tag **<audio>**: permette di inserire un contenuto audio
- il formato dei video e degli audio supportati dipende dai browser, tuttavia esistono dei formati di riferimento (mp4, webm, ogg per i video)
- esistono delle nuove API per video e audio che permettono la gestione di questo tipo di risorse da script

Form 2.0

- Nuovi attributi ed input types per le form sono stati introdotti in HTML5
- L’obiettivo principale è quello di permettere di definire le più comuni forme di validazione di form lato client direttamente nel documento HTML (cioè senza bisogno di aggiungere codice JavaScript)
- Sono stati inoltre aggiunti tipi di elementi utili soprattutto nei nuovi media agent (smartphone e tablet)

Autofocus, placeholder, form

- Nuovi attributi:
  - **autofocus** (booleano): all'apertura della form, il focus va sull'elemento che ha dichiarato autofocus
  - **placeholder** (per elementi input o textarea): valore che all'apertura della form compare sul campo editabile (N.B.: non corrisponde ad un valore (value) iniziale del campo, viene solo visualizzato all'inizio)
  - **form**: attributo che permette di associare un elemento ad una form. E' così possibile, ad esempio, dichiarare un campo che appartiene a più form, o dichiarare un campo all'esterno di un elemento form

Required, autocomplete

- **required** (booleano): rende obbligatoria la compilazione dell'elemento al momento del submit della form a cui l'elemento appartiene
- **autocomplete**: attributo che può assumere due valori:
  - **on** = il browser può effettuare l'autocompletion di questo campo, cioè completare il campo in maniera automatica usando i valori precedentemente inseriti in questo campo
  - **off** = il browser non può fare l'autocompletion
  - se autocomplete non viene assegnato, viene usato il default del browser (di solito è on)

Multiple, pattern

- **multiple** (booleano): permette al campo di accettare valori multipli
- esempio:

```
<form action='demo_form.asp'>
 sect images:<input type='file' name='img' multiple>
 <input type='submit'>
</form>
```

pattern: viene assegnato ad una espressione regolare; i valori del campo devono appartenere al linguaggio denotato dall'espressione regolare

Min, max, step

- **min** = valore minimo ammesso
- **max** = valore massimo ammesso
- **step** = intervallo tra un valore ammesso e il successivo
- **novalidate** (booleano): se dichiarato sull’elemento form, indica che non verrà effettuata la validazione degli elementi di quella form

Nuovi input types

- I seguenti input types permettono di gestire informazioni testuali di tipo specifico:
  - **tel**
  - **search**
  - **url**
  - **email**
  - **color**: permette la selezione di un colore
  - **number**: permette l'inserimento di un numero
  - **range**: permette l'inserimento di un numero attraverso uno slider (cursore orizzontale)

Nuovi input types

- Per gestire le date sono stati introdotti i seguenti input types:
  - **datetime**
  - **datetime-local**
  - **date**
  - **month**
  - **week**
  - **time**

Il tag <datalist>

- Permette di definire un campo testuale sul quale il browser può effettuare autocompletion usando un insieme predefinito di valori (l'utente però è libero di scrivere un valore al di fuori dell'elenco predefinito) esempio:

```
<input type='text' name='giudizio' list='listagiudizi'>
  <datalist id='listagiudizi'>
     <option value='insufficiente'>
     <option value='sufficiente'>
     <option value='discreto'>
     <option value='buono'>
     <option value='ottimo'>
  </datalist>
```