



# Client Side Programming HTML & CSS

A. Ferracani - [andreaFerracani@gmail.com](mailto:andreaFerracani@gmail.com)



<http://www.micc.unifi.it/people/andrea-ferracani>

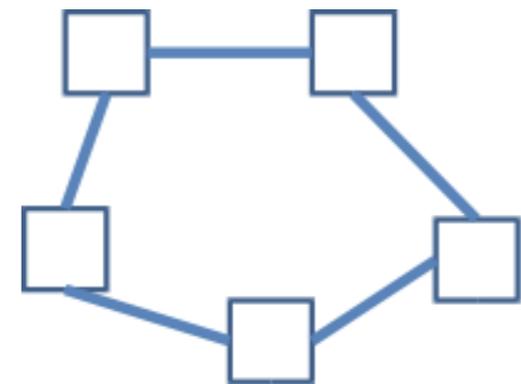
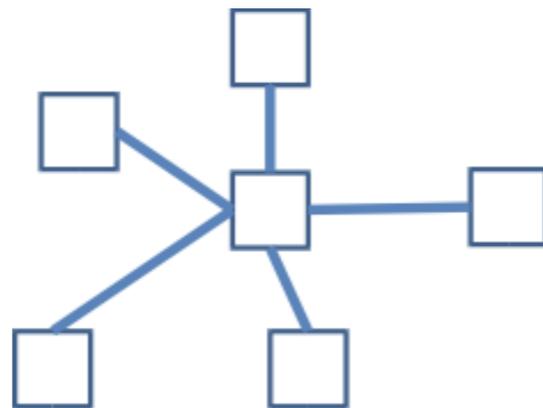
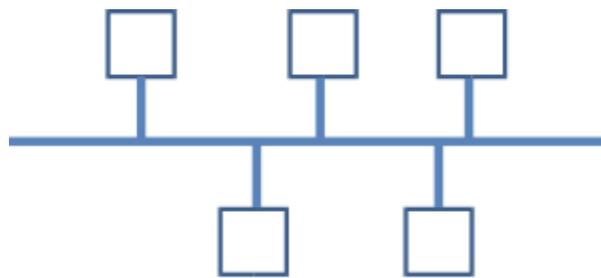
# Introduzione

---

Cos'è il web:

Internet e il Web sono due cose diverse:

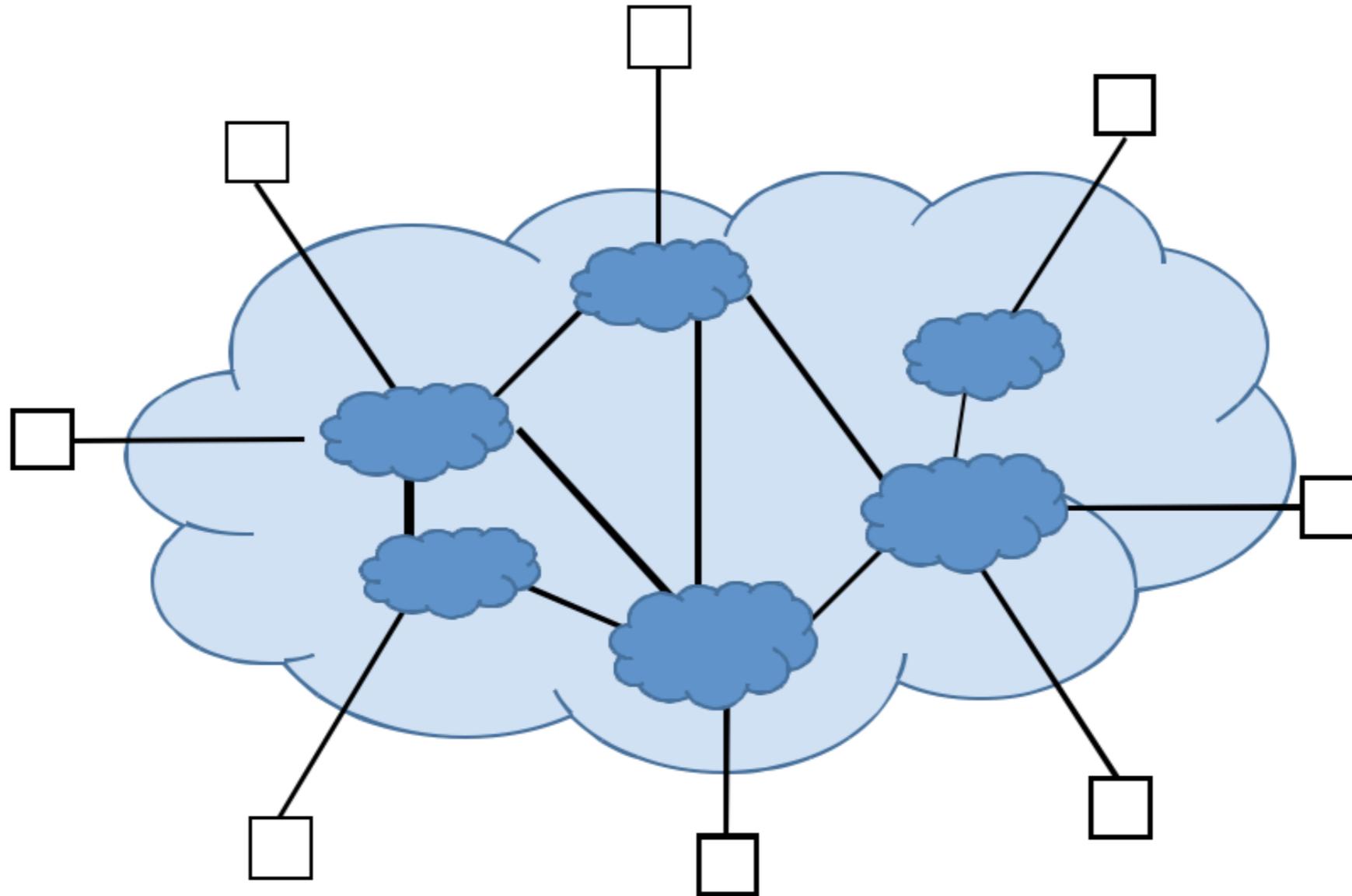
- ▶ Il **world wide web** è una rete mondiale di ipertesti che vengono trasmessi tra computer diversi attraverso il protocollo http. È una applicazione di Internet, ma ne esistono altre (reti di telefonia, mail)
- ▶ **Internet** è una rete mondiale di computer comunicanti tra loro attraverso connessioni fisiche.



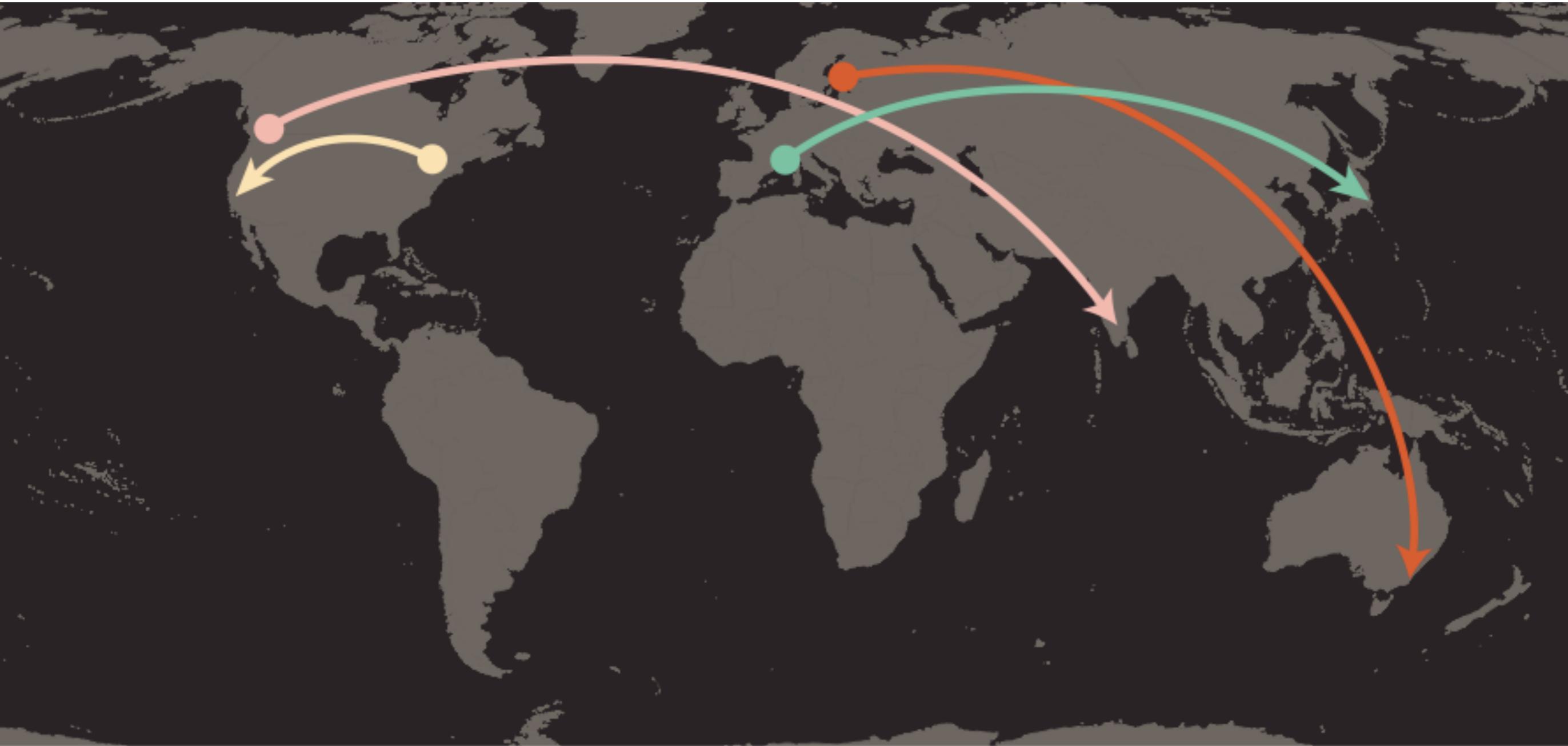
# Introduzione

---

Tutti noi percepiamo Internet come un'**unica rete virtuale**.



Come funziona il web: diversi **web clients** contattano attraverso **DNS** (Domain Name System) i **web servers**.



# Introduzione

---

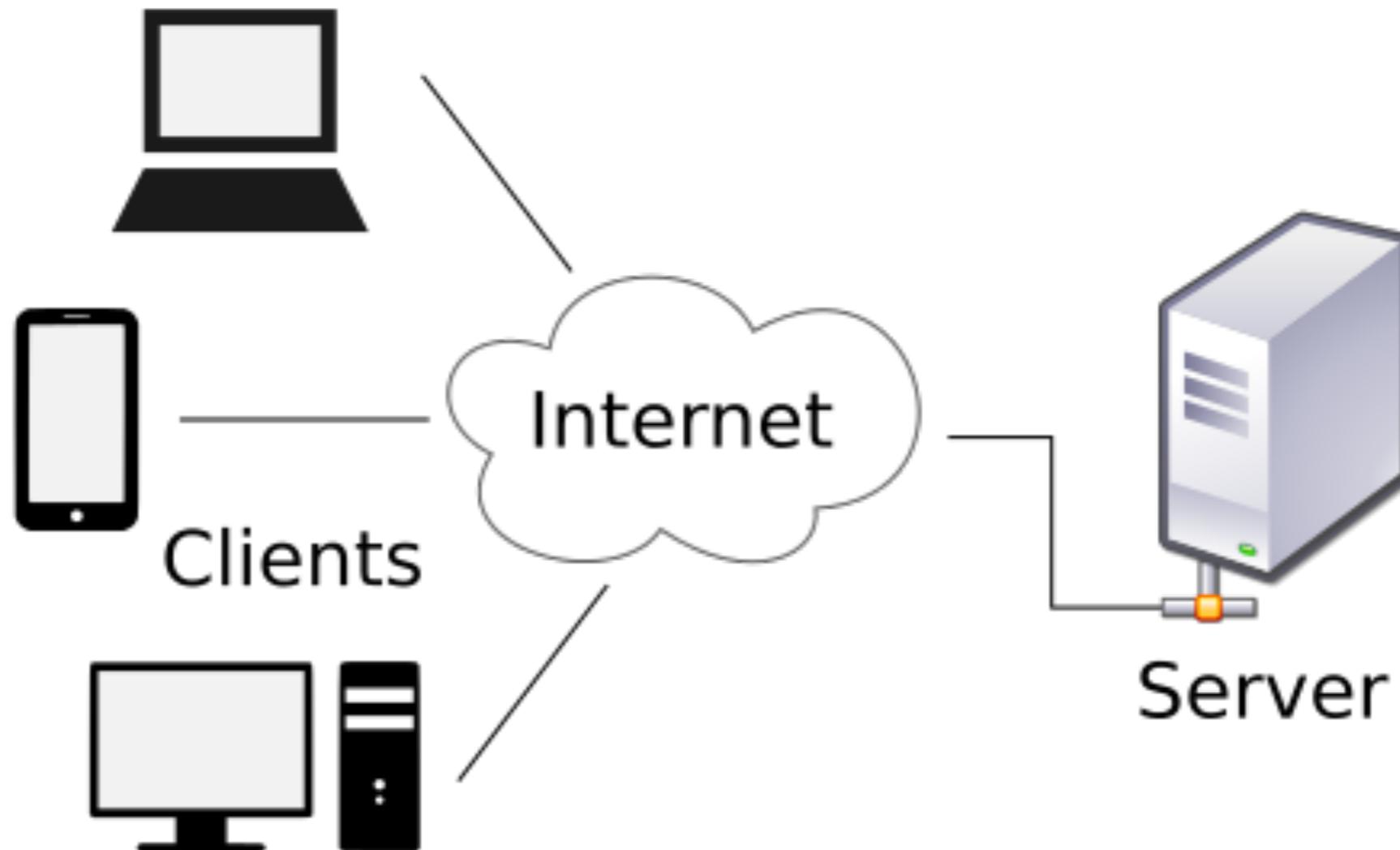
## Come funziona il web:

- ▶ Un client si connette al web attraverso un **ISP** (Internet Service Provider). L'utente digita un dominio nel **web address** (google.com)
  - ▶ Il **DNS** è come un phone book ed individua il computer che state contattando sulla rete attraverso un IP (150.217.35.77)
  - ▶ Con questo numero il **browser** chiama il server
  - ▶ Il server invia al browser la pagina **html ed il css**, linguaggi che è in grado di leggere e visualizzare
-

# Introduzione

---

Come funziona il web:



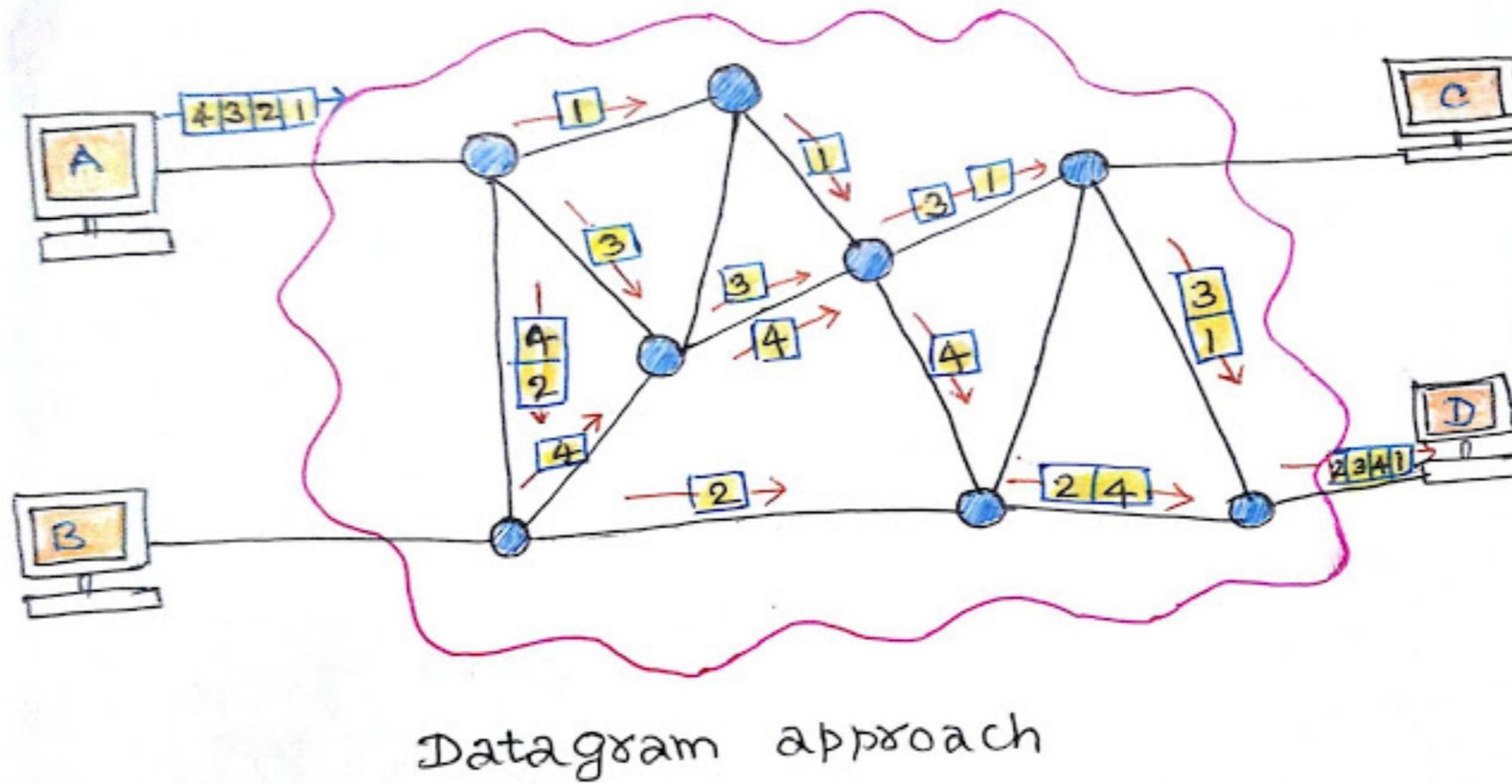
# Introduzione

---

## Come funziona il web: comunicazione

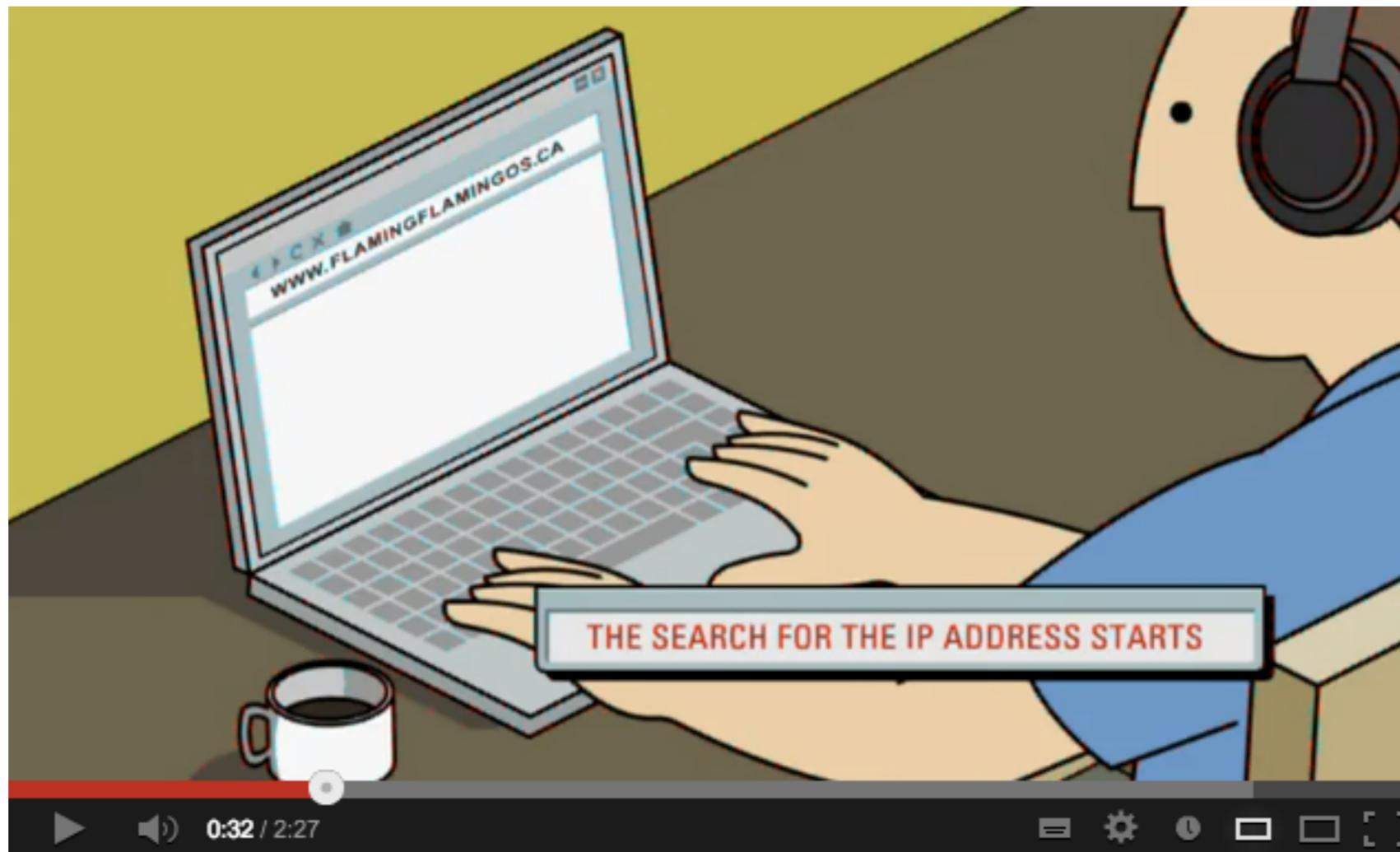
- ▶ Due computer connessi in rete comunicano attraverso un **protocollo di comunicazione** che specifica regole e formati per il trasferimento del messaggio.
- ▶ La comunicazione nel web avviene attraverso un protocollo che si chiama **HTTP** (hypertext transfer protocol): definisce qual'è la grammatica di riferimento della lingua in cui si parla: connectionless
- ▶ Un altro protocollo **TCP/IP** ha il compito stabilire e gestire questa comunicazione

## Come funziona il web: comunicazione



Come funziona il web: DNS (Domain Name System):

[www.youtube.com/watch?v=2ZUxoi7YNgs](https://www.youtube.com/watch?v=2ZUxoi7YNgs)



# Introduzione

---

## Sintesi

- ▶ Internet è una **rete di reti**
  - ▶ Il protocollo **TCP/IP** suddivide il data stream in pacchetti che seguono strade diverse
  - ▶ I computer sono individuati da un **indirizzo IP**
  - ▶ Il **DNS** associa indirizzi IP a nomi mnemonici strutturati attraverso server distribuiti
  - ▶ La grammatica di riferimento è l'**HTTP**
-

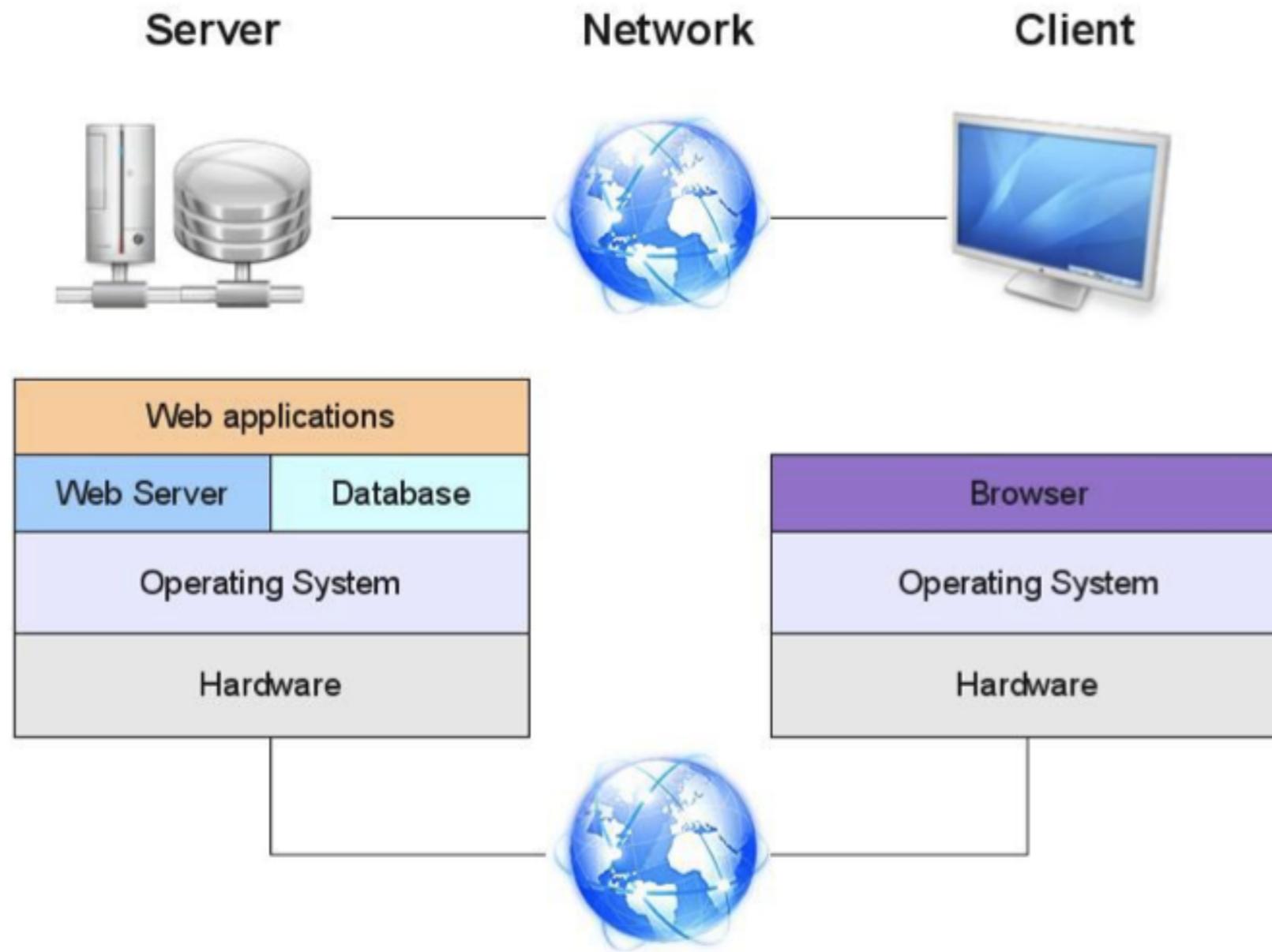
# Introduzione

---

## Come le persone accedono al web

- ▶ **Devices**: sono lo strumento fisico con cui le persone accedono: tablet, mobile phones, screen readers, tastiera braille
- ▶ **Web browsers**: Firefox, Chrome, Safari, Internet Explorer, Opera

## Cosa c'è dietro



# Introduzione

---

## Come sono fatti i siti web

- ▶ **HTML & CSS**: linguaggi di markup per semantica ed aspetto
  - ▶ **Flash & Silverlight & Javascript & Java**: linguaggi di programmazione per l'interattività
  - ▶ **CMS**: Content Management Systems
  - ▶ **HTML5 & CSS3**: la novità
-



# Introduzione

---

Un po' di storia: il world wide web è una rete mondiale di **ipertesti**.

- ▶ L'esigenza di evidenziare visivamente **riferimenti e connessioni tra testi** esiste da sempre.
  - ▶ **Commentari** a libri importanti (Omero, Bibbia, Talmud, Corano, Divina Commedia, etc.) esistono fin dal tardo impero romano ed utilizzano ogni sorta di trucco grafico e visivo per realizzare effetti di collegamento.
  - ▶ Era inevitabile che la meccanizzazione prima, e l'**informatizzazione** dopo, cercassero di soddisfare quest'esigenza.
-

# Introduzione

---

Gli anni '80: con il progredire della tecnologia, i sistemi ipertestuali escono dai laboratori e diventano un argomento ufficiale di ricerca e produzione software.

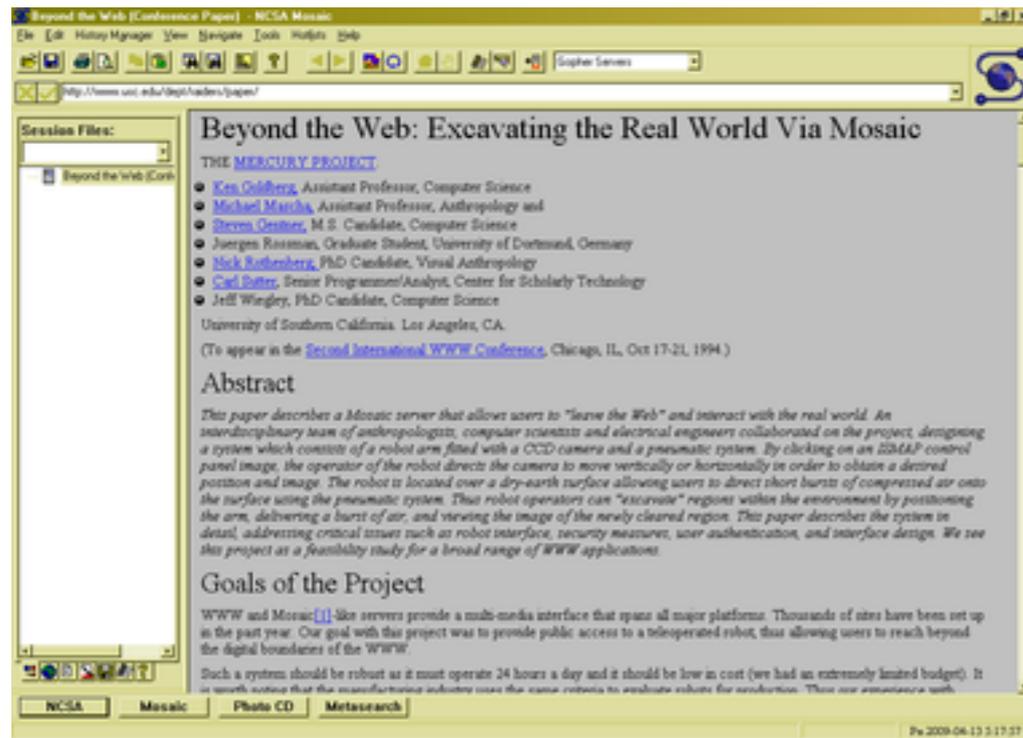
- ▶ Nel 1989 un gruppo di ricercatori del CERN, centro di ricerca in fisica nucleare a Ginevra, ricevette l'incarico di realizzare un meccanismo per la diffusione rapida di articoli, appunti e opinioni tra i fisici che ruotassero intorno al centro.



Nel 1991 **Tim Berners-Lee** e **Robert Cailliau** mostrarono (con poco successo) il primo prototipo client-server ed il primo browser chiamato: **WorldWideWeb**

Il prototipo era composto da:

- ▶ Un **server** che spediva documenti memorizzati localmente a chi lo richiedesse secondo il protocollo stabilito, e memorizzava documenti spediti da remoto
  - ▶ Un **editor** di testi parzialmente **WYSIWYG** che permetteva di visualizzare documenti ipertestuali e di modificarli, creando link e blocchi di testo.
-



Nell'ottobre del 1992 il **National Centre for Supercomputing Applications** (NCSA) esaminò il prototipo di WWW e decise di realizzarne una versione propria.

Con la realizzazione del server NCSA e del **primo browser WWW**, chiamato **Mosaic**, l'NCSA decretò l'inizio del successo esplosivo del sistema.



Nel frattempo, Berners-Lee e Cailliau cercano di mantenere il controllo sull'evoluzione del World Wide Web e fondano il **W3C**, con fondi della ricerca e dell'università.

Diventerà il World Wide Web Consortium, l'istituzione regolatrice del Web.

---

# Introduzione

---



I primi browsers: nel 1995 quando **Netscape** era diventato quasi monopolista dovette scontrarsi con un agguerrito concorrente: **Internet Explorer** della Microsoft.



Microsoft decise di distribuire il proprio browser **Internet Explorer 3** incluso nel sistema operativo **Windows 95**, così che cambiarlo diventasse compito degli utenti, i quali spesso non lo facevano.

Sebbene gli utenti di Netscape fossero pochissimi già nel 2003, lo sviluppo di Netscape proseguì fino al 28 dicembre **2007**, quando la AOL annunciò ufficialmente la fine del progetto.

---



**Internet Explorer** ha continuato lo sviluppo in maniera irregolare, senza introdurre cambiamenti importanti e senza rispettare le direttive del W3C.



**A partire dal 2004** iniziarono ad affermarsi browser con caratteristiche innovative e maggiore rispetto degli standard.



Quelli che hanno intaccato maggiormente l'egemonia Microsoft sono **Mozilla Firefox**, **Opera**, **Safari** e ultimo ma non ultimo **Google Chrome**.



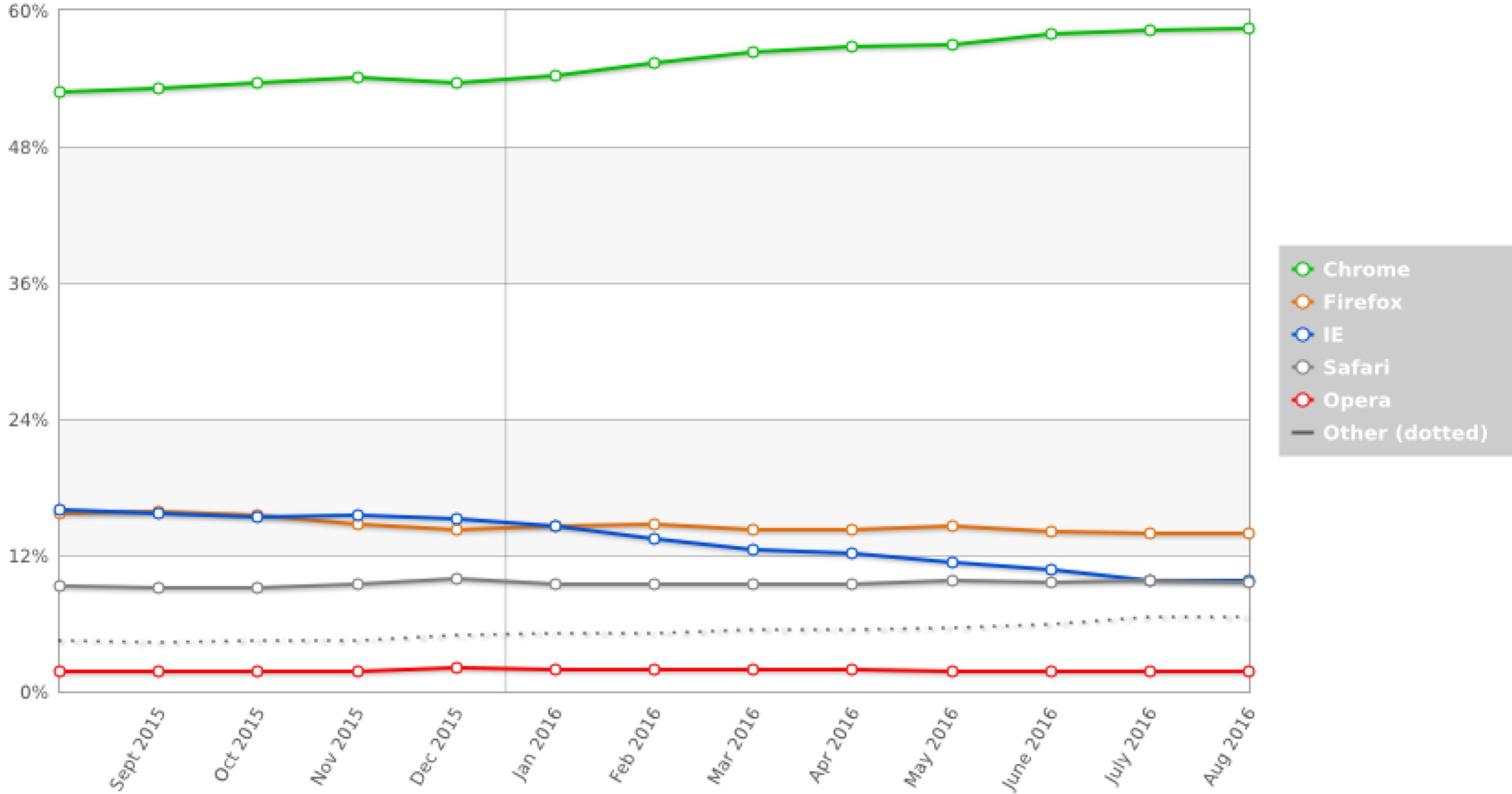
# Introduzione

---

**Browsers, alcuni punti:** un problema molto sentito nell'ambito del web development è il rispetto degli standard web proposti dal W3C.

- ▶ risulta difficile, in alcuni casi, permettere una **corretta visualizzazione** del proprio lavoro su qualsiasi browser
  - ▶ una volta progettato sito web è necessario controllarne la visualizzazione e le funzionalità su **diversi browser** in modo da assicurarti che tutti i visitatori possano usufruire del servizio
  - ▶ credere che **IE sia lo standard è un errore** soprattutto oggi che l'accesso al web avviene dai dispositivi più diversi: Android, iPhone.
-

**StatCounter Global Stats**  
Top 5 Desktop, Tablet & Console Browsers from Aug 2015 to Aug 2016

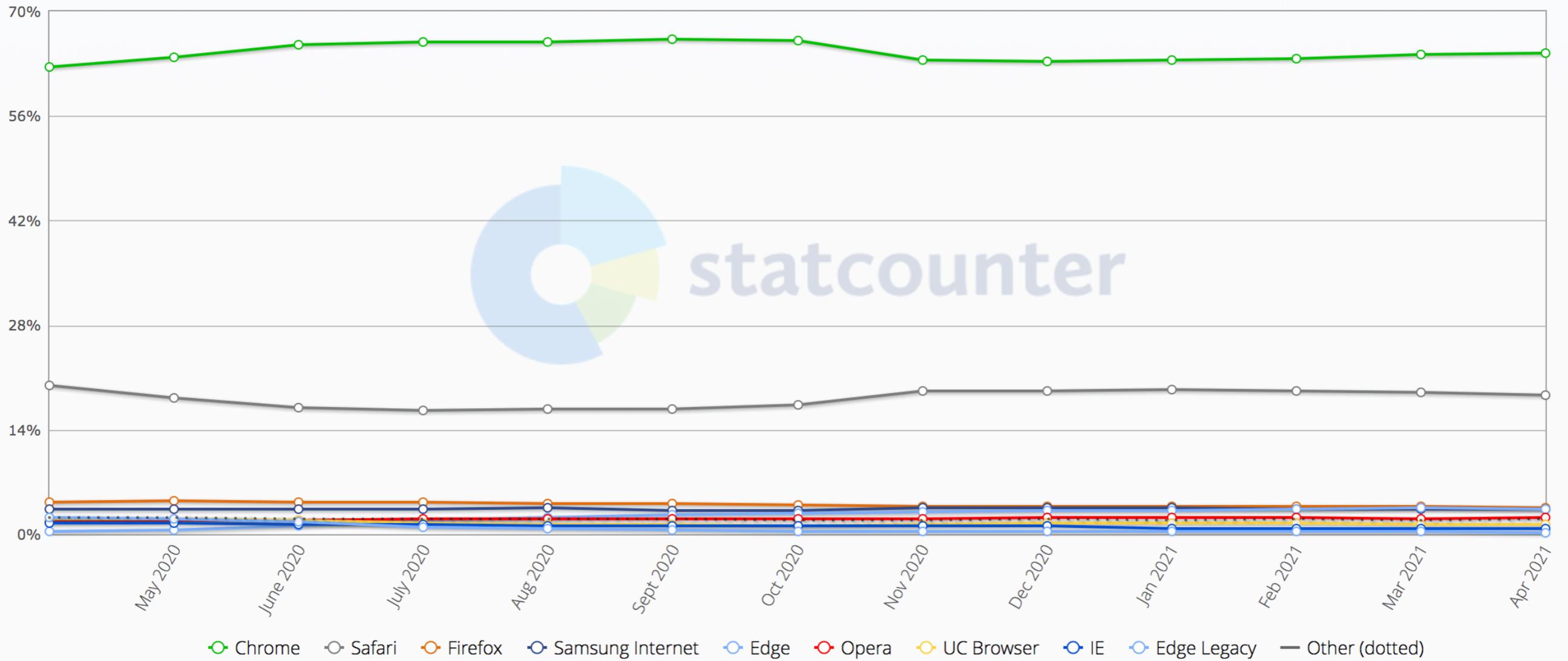


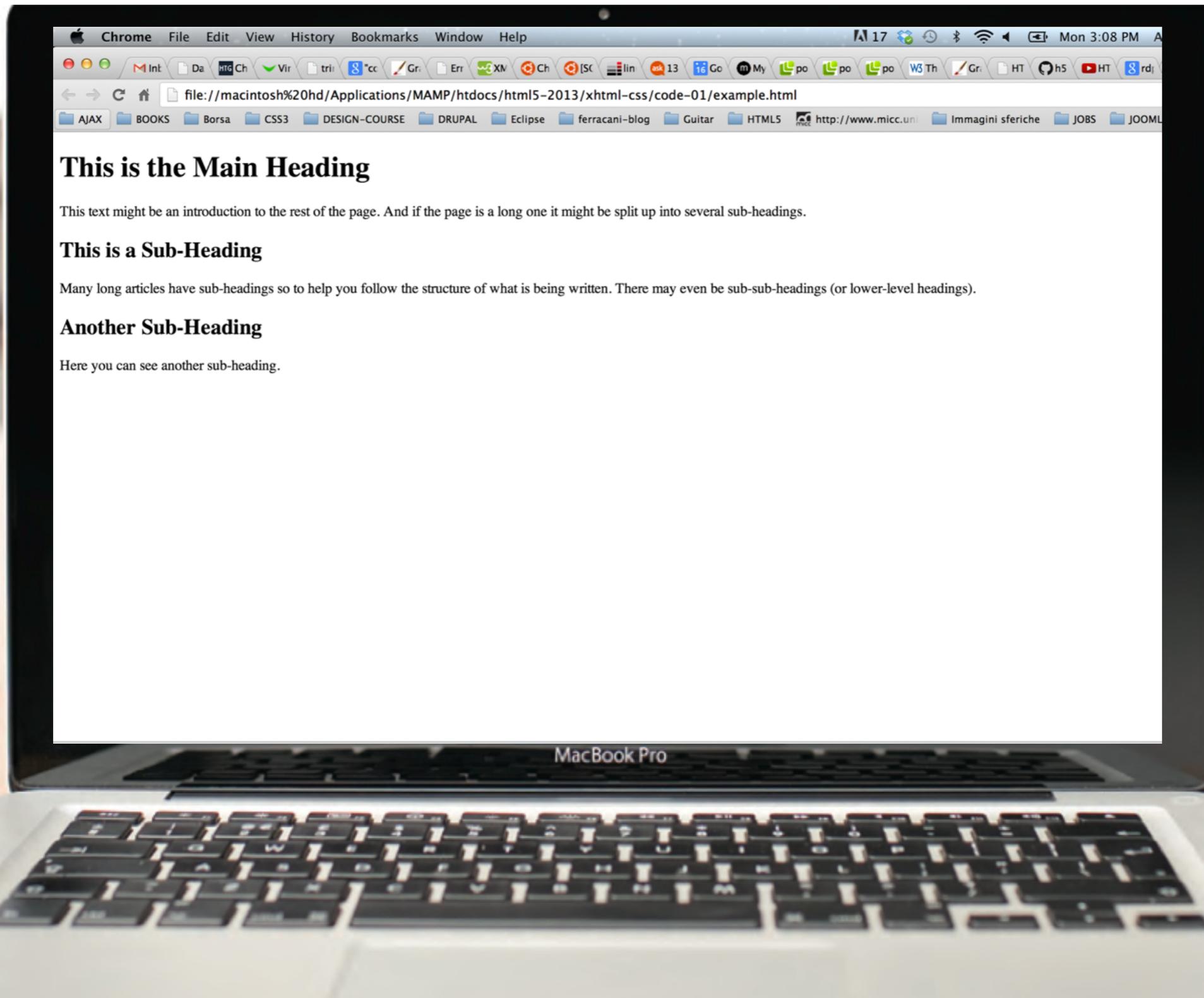
## Browsers market

### Browser Market Share Worldwide

Apr 2020 - Apr 2021

[Edit Chart Data](#)







La struttura dei contenuti sul web è molto simile a quella che conosciamo su altri canali, in particolare la **stampa**

Esiste una **gerarchia delle informazioni**: titoli, sottotitoli, paragrafi

La lingua: per definire cos'è XHTML possiamo iniziare con una semplice espressione: XML + HTML = XHTML

- ▶ **HTML** è l'acronimo di **HyperText Markup Language**.
  - ▶ **HTML** è un **linguaggio di marcatura** per presentare i contenuti di una pagina web (per la descrizione di come appaiono documenti all'interno di un browser). La sua semplicità è la base dell'esplosione di Internet. Fornisce strumenti per presentare titoli, paragrafi, font, link, immagini.
  - ▶ Con il termine **marcatura (markup)** si intende una sequenza di caratteri o altri simboli che si inseriscono all'interno di un documento per indicare come il contenuto deve apparire o per descrivere la struttura logica del documento.
-

L'HTML descrive la struttura della pagina:

```
<html>
  <body>
    <h1>This is the Main Heading</h1>
    <p>This text might be an introduction to the rest of the page. And if the page is a
long one it might be split up into several sub-headings.<p>
    <h2>This is a Sub-Heading</h2>
    <p>Many long articles have sub-headings so to help you follow the structure of
what is being written. There may even be sub-sub-headings (or lower-level headings).</
p>
    <h2>Another Sub-Heading</h2>
    <p>Here you can see another sub-heading.</p>
  </body>
</html>
```

L'HTML è fatto di tag che prevedono un elemento di apertura ed uno di chiusura. Il tag definisce ciò che contiene **a livello semantico**.

---

**XML** è un metalinguaggio: "a common syntax for expressing structure in data" ovvero un linguaggio per definire nuovi linguaggi di markup. XML consente di crearsi i propri tipi di documenti.

Lo scopo di XML è quello di **separare la definizione dei dati dalla rappresentazione**, per favorire lo scambio di documenti strutturati.

Principali obiettivi di XML, dichiarati nella prima specifica ufficiale (ottobre 1998), sono:

- ▶ utilizzo del linguaggio su Internet,
- ▶ facilità di creazione dei documenti,
- ▶ supporto di più applicazioni,
- ▶ chiarezza e comprensibilità.

```
<breakfast_menu>
  <food>
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>
      two of our famous Belgian Waffles with plenty of
      real maple syrup
    </description>
    <calories>650</calories>
  </food>

  <food>
    <name>French Toast</name>
    <price>$4.50</price>
    <description>
      thick slices made from our homemade s
      ourdough bread
    </description>
    <calories>600</calories>
  </food>

  <food>
    <name>Homestyle Breakfast</name>
    <price>$6.95</price>
    <description>
      two eggs, bacon or sausage, toast,
      and our ever-popular hash browns
    </description>
    <calories>950</calories>
  </food>
</breakfast_menu>
```

```
<html>
  <body>
    <h1>This is the Main Heading</h1>
    <p>This text might be an introduction to the rest of
      the page. And if the page is a long one it might
      be split up into several sub-headings.</p>
    <h2>This is a Sub-Heading</h2>
    <p>Many long articles have sub-headings so to help
      you follow the structure of what is being written.
      There may even be sub-sub-headings (or lower-level
      headings).</p>
    <h2>Another Sub-Heading</h2>
    <p>Here you can see another sub-heading.</p>
  </body>
</html>
```

The opening `<html>` tag indicates that anything between it and a closing `</html>` tag is HTML code.

The `<body>` tag indicates that anything between it and the closing `</body>` tag should be shown inside the main browser window.

Words between `<h1>` and `</h1>` are a main heading.

A paragraph of text appears between these `<p>` and `</p>` tags.

Words between `<h2>` and `</h2>` form a sub-heading.

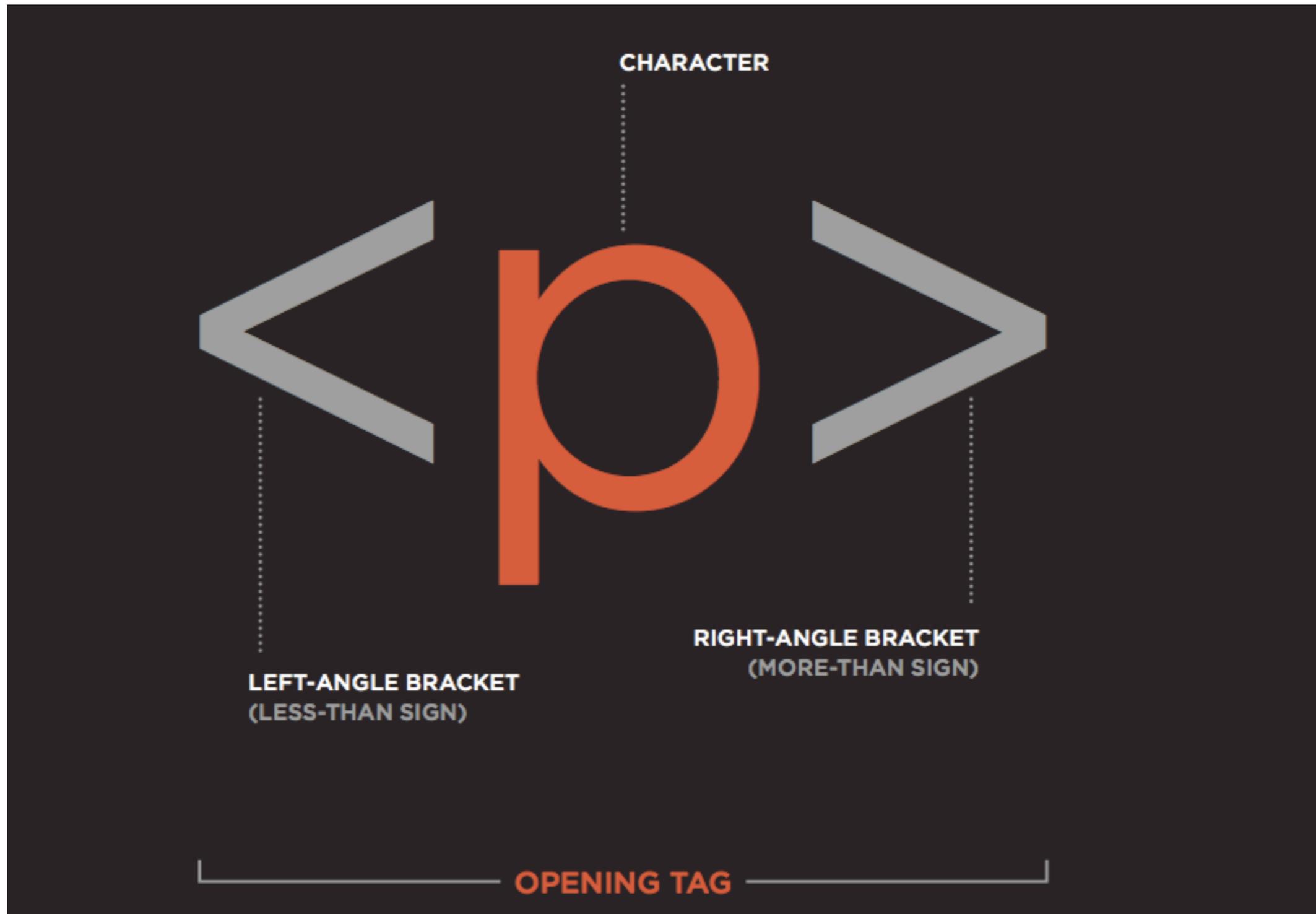
Here is another paragraph between opening `<p>` and closing `</p>` tags.

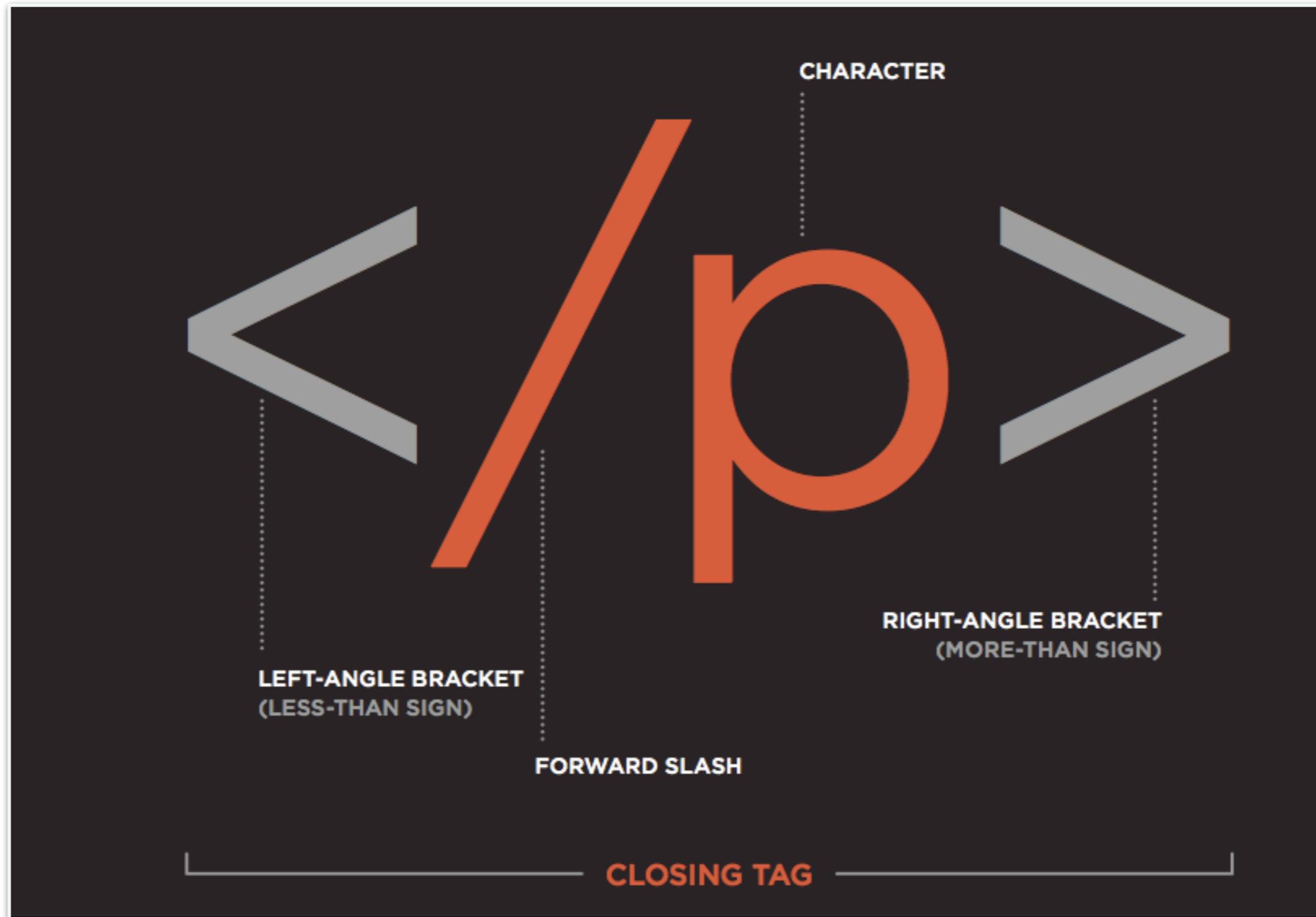
Another sub-heading inside `<h2>` and `</h2>` tags.

Another paragraph inside `<p>` and `</p>` tags.

The closing `</body>` tag indicates the end of what should appear in the main browser window.

The closing `</html>` tag indicates that it is the end of the HTML code.





Si possono usare **attributi** per aggiungere semantica ai tag



/code-01/body-head-title.html

HTML

```
<html>
  <head>
    <title>This is the Title of the Page</title>
  </head>
<body>

<h1>This is the Body of the Page</h1>
  <p>Anything within the body of a web page is
displayed in the main browser window.
  </p> [...]
```

- ▶ `<body>`: ciò che è visibile nella finestra del browser
- ▶ `<head>`: meta-informazioni pagina
- ▶ `<title>`: ciò che appare nella tab del browser



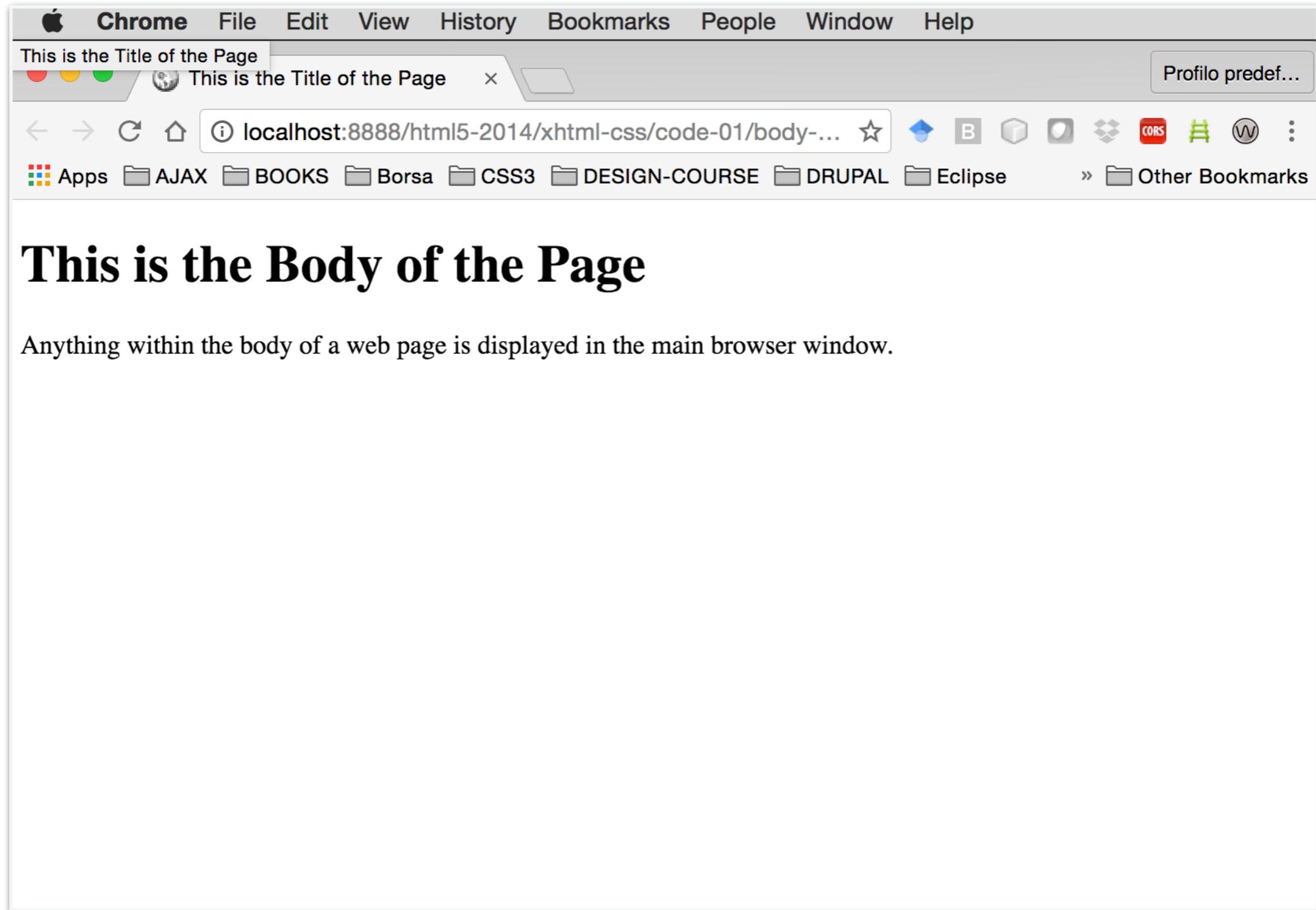
## This is the Body of the Page

Anything within the body of a web page is displayed in the main browser window.

Un documento HTML è normalmente diviso in **due sezioni principali**:

- ▶ **<head>**: Contiene informazioni che riguardano il modo in cui il documento deve essere letto e interpretato. Questo è il luogo dove scrivere, ad esempio, i **meta-tag** (p.e. per i motori di ricerca), script JavaScript, **fogli di stile**
- ▶ **<body>**: ciò che è visibile nella finestra del browser





# HTML & CSS

---

## Sintesi:

- ▶ Le pagine HTML sono **documenti di testo**
- ▶ L'HTML usa i **tag** per dare significato a parti del testo
- ▶ I tag si **aprono** e si **chiudono**
- ▶ I tag posso avere **attributi**
- ▶ Gli attributi sono **coppie nome / valore**
- ▶ Per scrivere HTML bisogna conoscere il significato dei tag

Esercizio: `code-01/example.html`

---

**Attributi:** ai TAG possono essere associati **attributi** e agli attributi un valore. Il contenuto va inserito tra l'apertura e la chiusura del TAG medesimo.

```
<body bgcolor="blue">
```

impostare un colore di sfondo è necessario impostare il relativo attributo del tag body

```
<body background="imgSfondo.gif">
```

inserire un'immagine come sfondo. L'immagine di sfondo verrà ripetuta in orizzontale e in verticale.

```
<body bgcolor="#0000ff" background="imgSfondo.gif">
```

combinare sullo sfondo il colore con l'immagine

```
<body bgcolor="#0000FF">
```

per scegliere il colore è comunque più opportuno utilizzare la corrispondente codifica esadecimale

---

## Codifica colori esadecimali

colore	parola chiave	notazione esadecimale
arancione	orange	#FFA500
blu	blue	#0000FF
bianco	white	#FFFFFF
giallo	yellow	#FFFF00
grigio	gray	#808080
marrone	brown	#A52A2A
nero	black	#000000
rosso	red	#FF0000
verde	green	#008000
viola	violet	#EE82EE

## Un altro esempio

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <title>title</title>
  </head>
  <body
    leftmargin="0"
    topmargin="0"
    bgcolor="#66CCFF"
    lang="it">
    Testo di prova
  </body>
</html>
```

# HTML & CSS

---

## HTML

I TAG possono essere annidati l'uno dentro l'altro:

```
<TAG1 attributi>contenuto 1  
  <TAG2>contenuto 2</TAG2>  
</TAG1>
```

Ad esempio, per attribuire formattazioni successive al testo che stiamo inserendo, attraverso attributi:

```
<P align="right">testo 1</P>  
<P align="left">testo 2</P>
```

# HTML & CSS

---

HTML è **case insensitive**, è del tutto indifferente se scrivere i TAG in maiuscolo o in minuscolo:

```
<P ALIGN="RIGHT">
```

e

```
<p align="right">
```

vengono letti allo stesso modo dal browser

XHTML è invece **case sensitive**

---

## HTML - Commenti

Per rendere il codice più leggibile è opportuno inserire "commenti" nei punti più significativi: in modo da mantenere l'orientamento anche in file molto complessi e lunghi.

La sintassi è la seguente:

```
<!-- questo è un commento -->
```

**XHTML:** è la **reformulazione di HTML come applicazione XML**. Ciò significa essenzialmente una cosa: un documento XHTML deve essere valido e ben formato.

Una pagina XHTML può essere sottoposta ad un validatore:

Il validatore controlla:

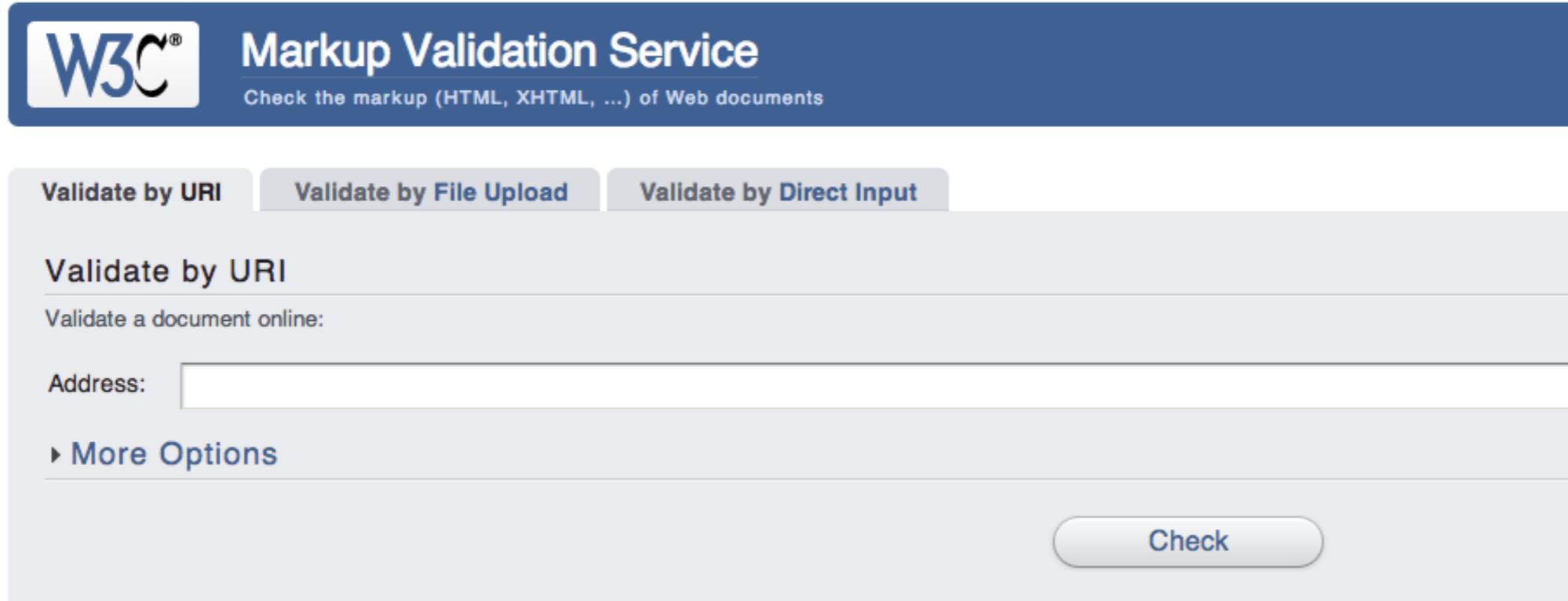
- ▶ che il codice sia **ben formato**, ovvero rispetti la corretta grammatica e sequenza di tag
- ▶ che il codice **sia valido**, ovvero che sia conforme ai tipi di dato definiti nel **DTD**.

Il W3C definisce quale tipo di dato deve essere definito all'interno di un documento XHTML 1 Strict

<http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd>

## W3C Markup validation service

<http://validator.w3.org>



The screenshot shows the W3C Markup Validation Service interface. At the top, there is a blue header with the W3C logo and the text "Markup Validation Service" and "Check the markup (HTML, XHTML, ...) of Web documents". Below the header, there are three tabs: "Validate by URI", "Validate by File Upload", and "Validate by Direct Input". The "Validate by URI" tab is selected. Under this tab, there is a section titled "Validate by URI" with the text "Validate a document online:". Below this, there is a label "Address:" followed by a text input field. To the left of the input field, there is a link "More Options". At the bottom right of the form, there is a "Check" button.

This validator checks the [markup validity](#) of Web documents in HTML, XHTML, SMIL, MathML, etc. If you wish to validate specific [MobileOK content](#), or to [find broken links](#), there are [other validators and tools](#) available. As an alternative you can also try our [non](#)



**NEW** - W3C offers a beta release of a new service providing you an integrat [Try it now](#) to quickly identify those portions of your web site th

Con l'introduzione di **XHTML**, piuttosto che sfornare una nuova versione del linguaggio il W3C ha compiuto un'opera di **ridefinizione**:

- ▶ Niente nuovi tag, attributi o metodi (HTML 4.0)
  - ▶ Il vocabolario rimane uguale, ma **cambiano le regole sintattiche**. Se vengono inseriti elementi non supportati (font, larghezza per le celle di tabelle o margini per il body, per citare solo alcuni esempi) il documento non è valido.
  - ▶ Con XHTML, almeno nella sua versione Strict, si torna ad un linguaggio che definisce **solo la struttura**. La formattazione si fa con i CSS.
-

## HTML vs XHTML

- ▶ In HTML è possibile **omettere** gli elementi html, head, body e DOCTYPE. In XHTML sono obbligatori
  - ▶ Con HTML è possibile omettere alcuni **TAG di chiusura**. In XHTML sono obbligatori, anche con i TAG vuoti.
  - ▶ Con HTML è possibile omettere le **virgolette per i valori degli attributi** che non contengono spazi, o caratteri speciali. In XHTML sono obbligatorie.
  - ▶ HTML non fa differenze tra **minuscole e maiuscole** (case insensitive). In XHTML impone che gli elementi, gli attributi e i valori predefiniti siano scritti in minuscolo.
-

## XHTML - DTD

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

```
<html xmlns="http://www.w3.org/1999/xhtml">  
  <head>  
    <meta http-equiv="Content-Type" content="text/html;  
charset=UTF-8" />  
    <title>Untitled Document</title>  
  </head>
```

La **dichiarazione DOCTYPE** indica che tipo di XHTML è usato, in questo modo i browser sanno come interpretare il codice e i validatori sanno come verificare la sintassi.

---

# HTML & CSS

---

## HTML5

- ▶ HTML5 è il **nuovo standard** di HTML.
- ▶ La versione precedente, **HTML 4.01**, è nata nel 1999. Il web è cambiato enormemente da allora
- ▶ HTML5 è ancora un **work in progress**. Comunque, i browser principali supportano molte delle nuove HTML5 elements and APIs

HTML5 è nato da una cooperazione fra il **World Wide Web Consortium (W3C)** ed il **Web Hypertext Application Technology Working Group (WHATWG)**

WHATWG lavorava su web forms and applications, ed il W3C stava lavorando con XHTML 2.0. Nel 2006, decisero di cooperare e creare una nuova versione di HTML.

---

HTML5: [www.youtube.com/watch?v=6BAflsaNRnk](http://www.youtube.com/watch?v=6BAflsaNRnk)

HTML5 IS ON THE  
MOVE



# HTML & CSS

---

HTML5: alcune regole osservate nella formulazione

- ▶ Le nuove funzionalità sono basate su **HTML, CSS, DOM, and JavaScript**
  - ▶ Ridurre l'uso di **plugin esterni** (Flash e Silverlight)
  - ▶ Miglior gestione degli errori
  - ▶ Più **markup** invece che scripting
  - ▶ HTML5 deve essere **indipendente dai device**
  - ▶ Il processo di sviluppo deve essere **pubblico**
-

HTML5: alcune delle nuove funzionalità

- ▶ `<canvas>` tag per il disegno 2D
- ▶ `<video>` e `<audio>` tags per media playback
- ▶ Supporto per **local storage**
- ▶ Nuovi elementi per il contenuto `<article>`, `<footer>`, `<header>`, `<nav>`, `<section>`
- ▶ Nuovi controlli nelle Form: **calendar**, **date**, **time**, **email**, **url**, **search**

HTML5: HTML5 non è ancora uno standard ufficiale, e nessuno dei browsers ha un supporto completo.

Ma tutti i **maggiori browsers** (Safari, Chrome, Firefox, Opera, Internet Explorer) continuano ad aggiungere nuove funzionalità HTML5 alle loro ultime versioni.



# HTML & CSS

---

## HTML5: pagina di base

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <title>Title of the document</title>
```

```
  </head>
```

```
  <body>
```

```
    The content of the document.....
```

```
  </body>
```

```
</html>
```

```
HTML5 HTML
<!DOCTYPE html>

HTML 4
<!DOCTYPE html PUBLIC
  "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">

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

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

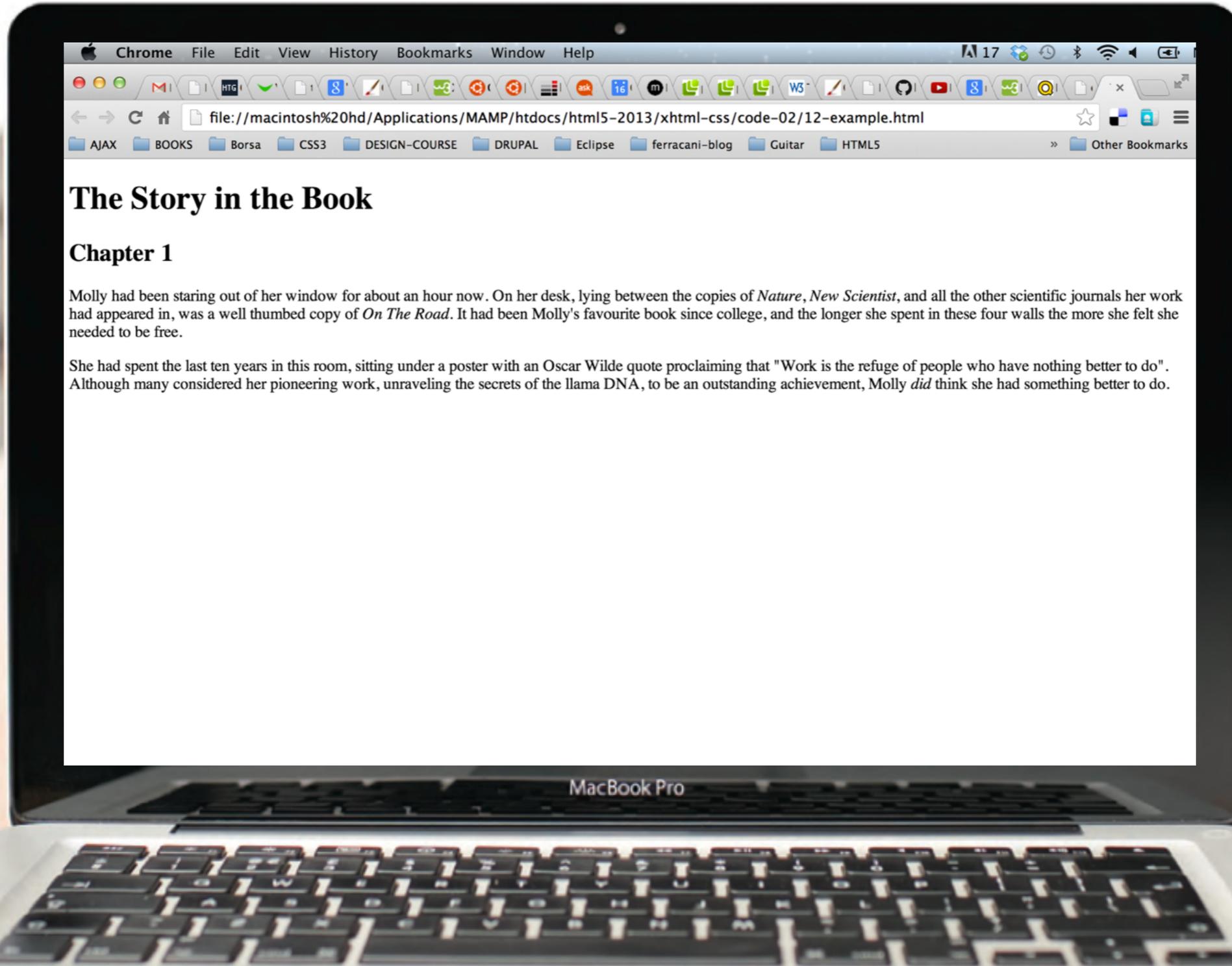
XML Declaration
<?xml version="1.0" ?>
```

## HTML5: sintesi

HTML5 is The New HTML Standard

HTML	HTML5
	<ul style="list-style-type: none"><li>• New Elements</li><li>• New Attributes</li><li>• Full CSS3 Support</li><li>• Video and Audio</li><li>• 2D/3D Graphics</li><li>• Local Storage</li><li>• Local SQL Database</li><li>• Web Applications</li></ul>

Navigation icons: HTML5 logo, folder, cube, circular arrow, double chevron, HTML5 logo

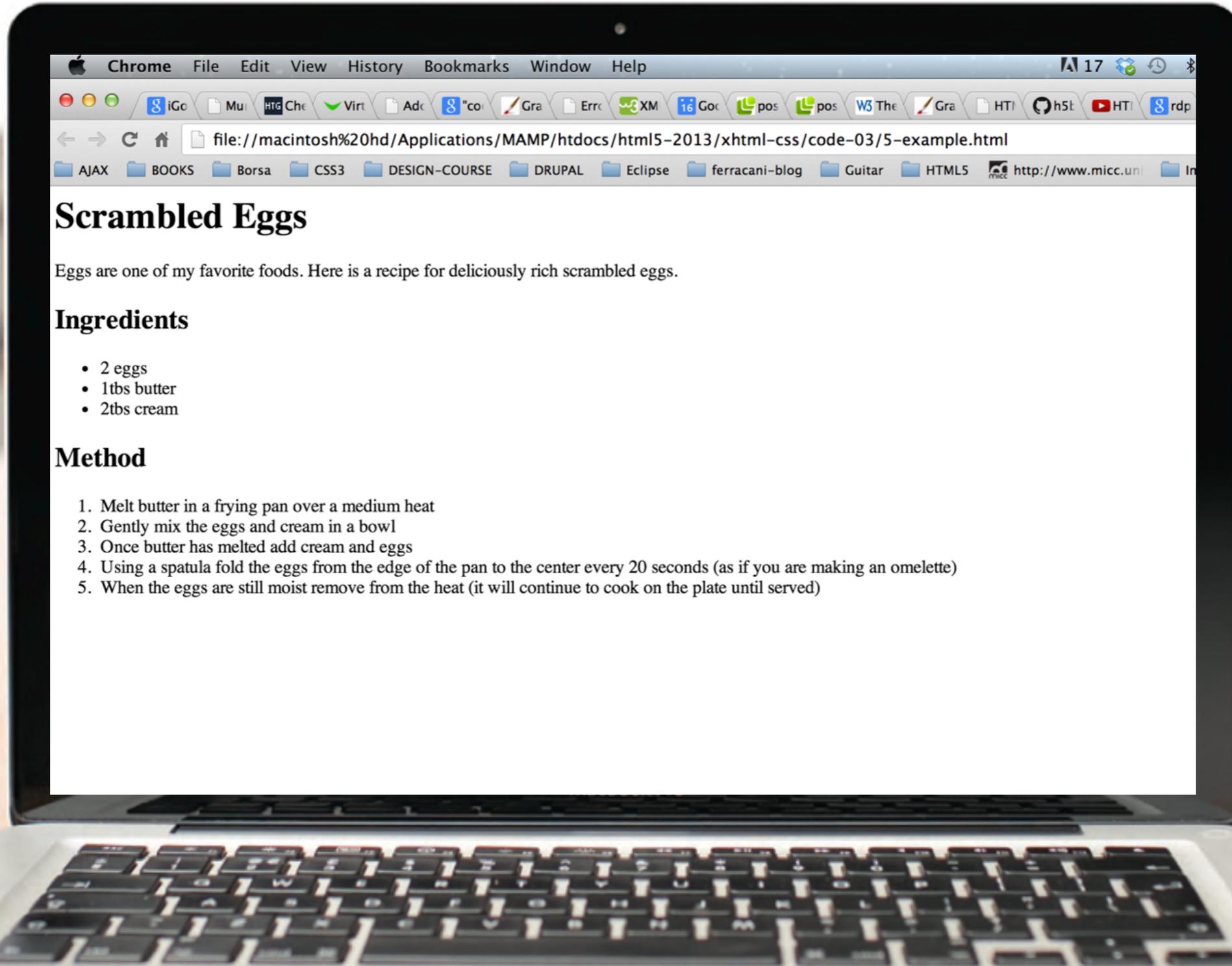


## HTML5: markup testo

```
<html>
  <head>
    <title>Text</title>
  </head>
  <body>
    <h1>The Story in the Book</h1>
    <h2>Chapter 1</h2>
    <p>Molly had been staring out of her window for about
an hour now. On her desk, lying between the copies of <em>Nature</em>, <em>New
Scientist</em>, and all the other scientific journals her work had appeared in, was a well
thumbed copy of <cite>On The Road</cite>. It had been Molly's favorite book since college,
and the longer she spent in these four walls the more she felt she needed to be free.</p>
<p>She had spent the last ten years in this room, sitting under a poster with an Oscar Wilde
quote proclaiming that <q>Work is the refuge of people who have nothing better to do</q>.
Although many considered her pioneering work, unraveling the secrets of the llama <abbr
title="Deoxyribonucleic acid">DNA</abbr>, to be an outstanding achievement, Molly
<em>did</em> think she had something better to do.</p>
  </body>
</html>
```

Esercizio: code-02/12-example.html

---



## HTML5: markup liste

```
<html>
  <head>
    <title>Lists</title>
  </head>
  <body>
    <h1>Scrambled Eggs</h1>
    <p>Eggs are one of my favourite foods. Here is a recipe for deliciously rich scrambled eggs.</p>
    <h2>Ingredients</h2>

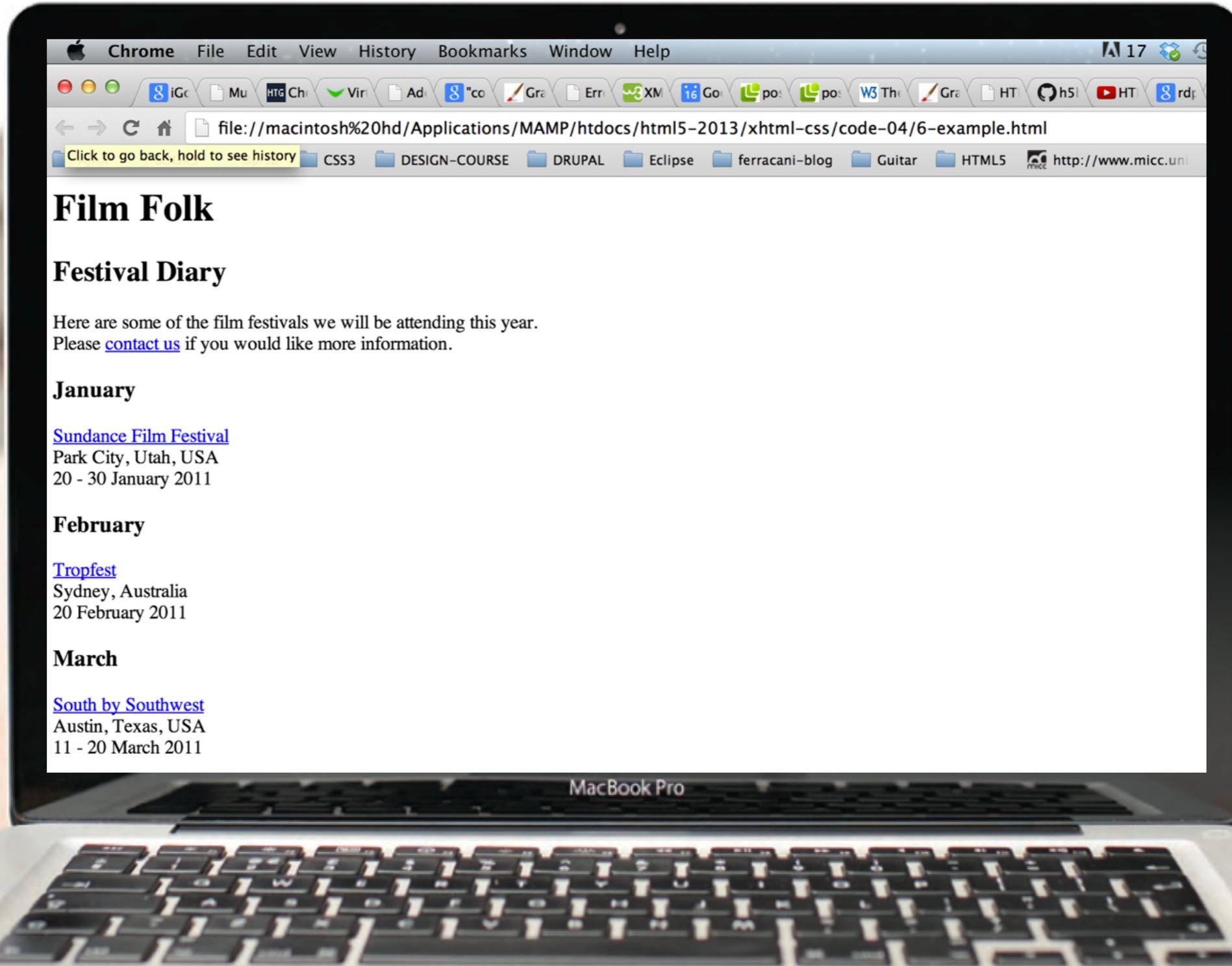
    <ul>
      <li>2 eggs</li>
      <li>1tbs butter</li>
      <li>2tbs cream</li>
    </ul>

    <h2>Method</h2>

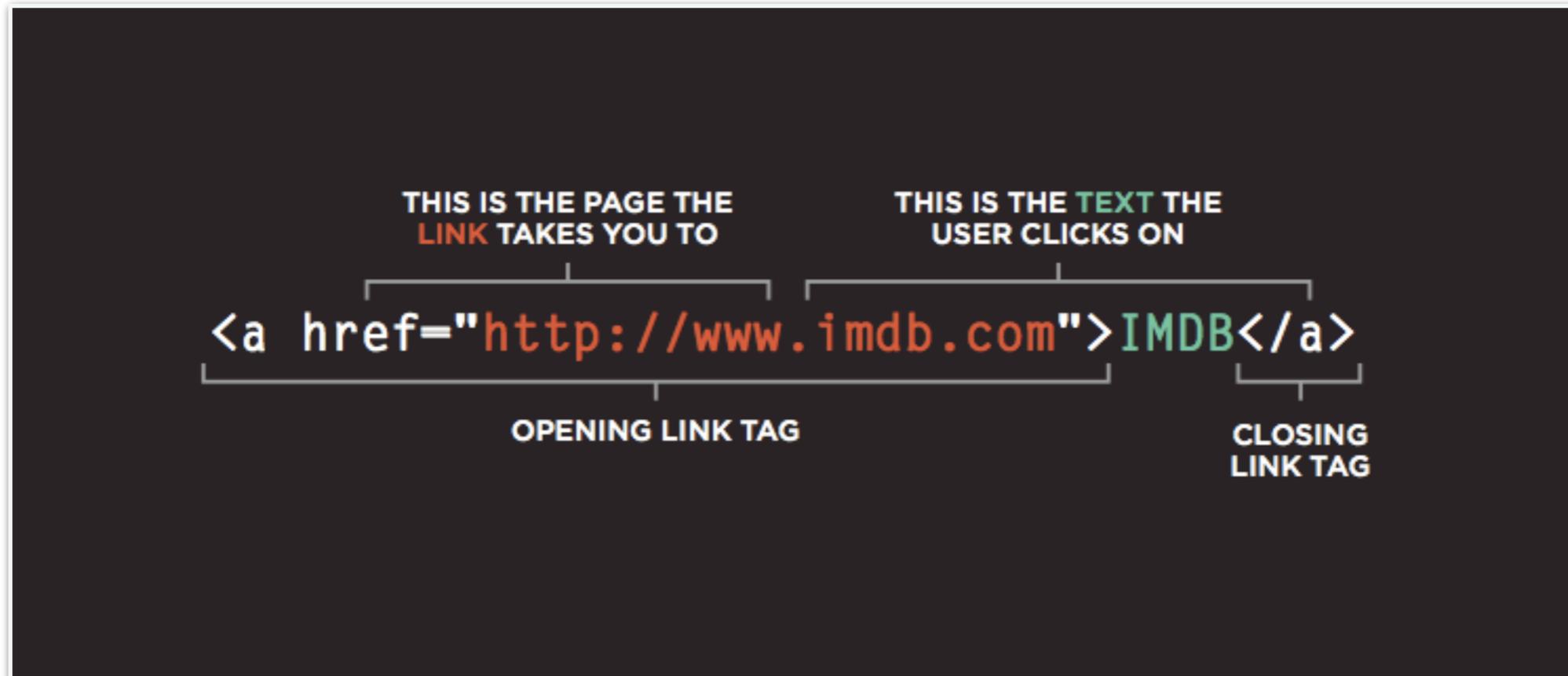
    <ol>
      <li>Melt butter in a frying pan over a medium heat</li>
      <li>Gently mix the eggs and cream in a bowl</li>
      <li>Once butter has melted add cream and eggs</li>
      <li>Using a spatula fold the eggs from the edge of the pan to the center every 20 seconds (as if you are making an omelette)</li>
      <li>When the eggs are still moist remove from the heat (it will continue to cook on the plate until served)</li>
    </ol>

  </body>
</html>
```

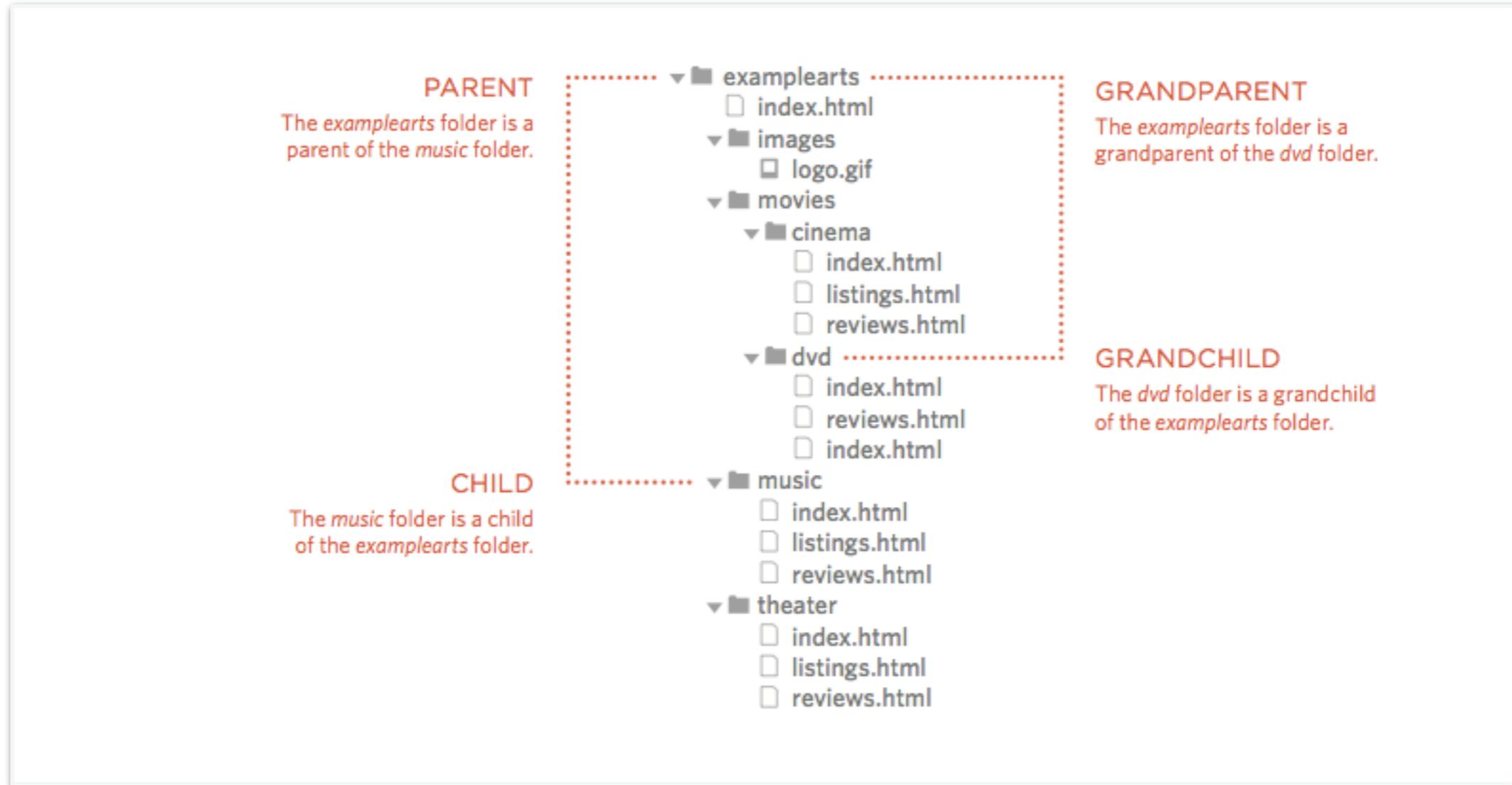
Esercizio: code-03/5-example.html



## HTML5: links



## HTML5: struttura delle cartelle di un sito web



## HTML5: URL relative

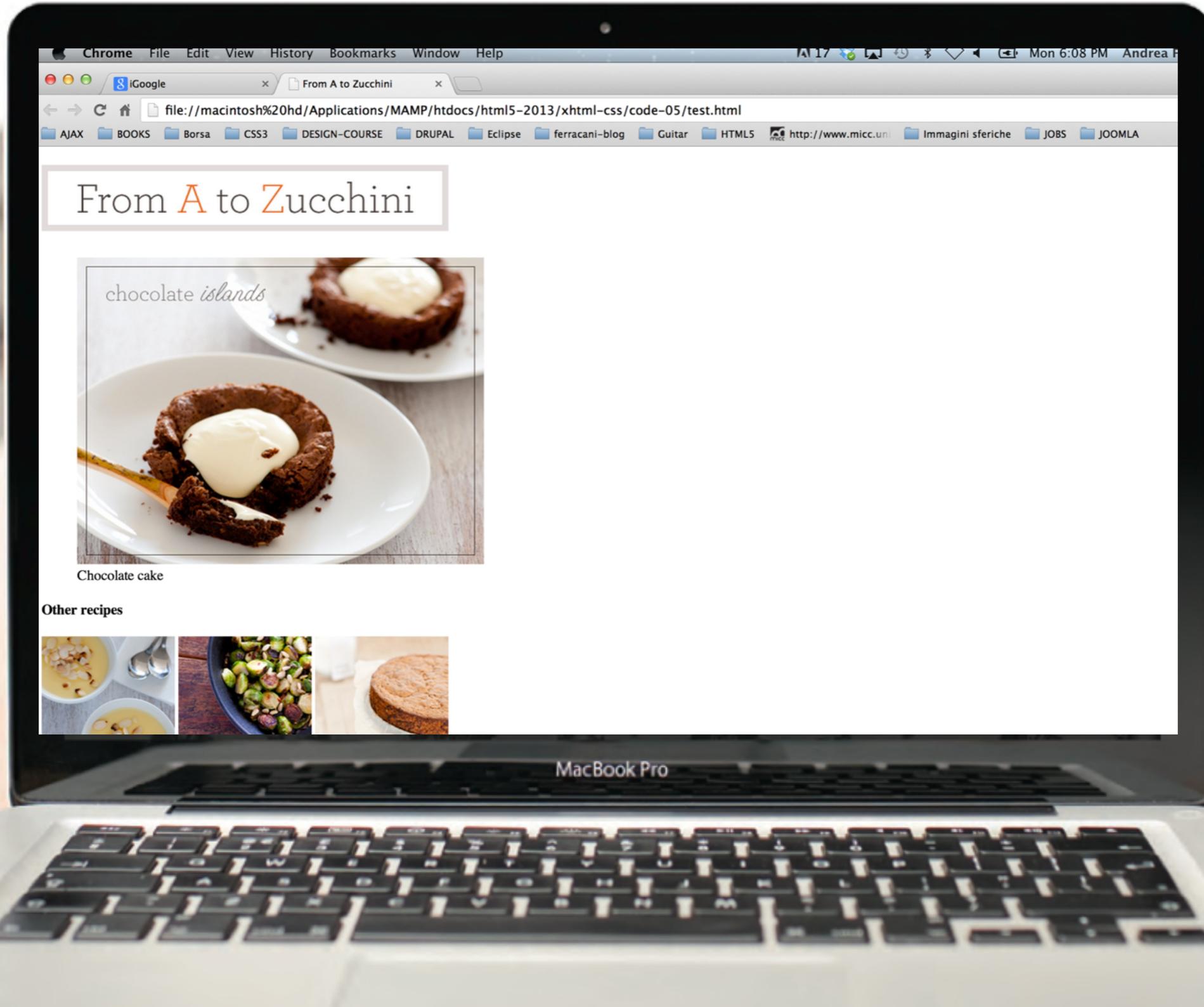
RELATIVE LINK TYPE	EXAMPLE (from diagram on previous page)
<b>SAME FOLDER</b> To link to a file in the same folder, just use the file name. (Nothing else is needed.)	To link to music reviews from the music homepage: <code>&lt;a href="reviews.html"&gt;Reviews&lt;/a&gt;</code>
<b>CHILD FOLDER</b> For a child folder, use the name of the child folder, followed by a forward slash, then the file name.	To link to music listings from the homepage: <code>&lt;a href="music/listings.html"&gt;Listings&lt;/a&gt;</code>
<b>GRANDCHILD FOLDER</b> Use the name of the child folder, followed by a forward slash, then the name of the grandchild folder, followed by another forward slash, then the file name.	To link to DVD reviews from the homepage: <code>&lt;a href="movies/dvd/reviews.html"&gt;Reviews&lt;/a&gt;</code>
<b>PARENT FOLDER</b> Use <code>../</code> to indicate the folder above the current one, then follow it with the file name.	To link to the homepage from the music reviews: <code>&lt;a href="../index.html"&gt;Home&lt;/a&gt;</code>
<b>GRANDPARENT FOLDER</b> Repeat the <code>../</code> to indicate that you want to go up two folders (rather than one), then follow it with the file name.	To link to the homepage from the DVD reviews: <code>&lt;a href="../../index.html"&gt;Home&lt;/a&gt;</code>

## HTML5: markup links

```
<html>
  <head>
    <title>Links</title>
  </head>
  <body>
    <h1 id="top">Film Folk</h1>
    <h2>Festival Diary</h2>
    <p>Here are some of the film festivals we will be attending this year.<br />Please <a
href="mailto:filmfolk@example.org">contact us</a> if you would like more information.</p>
    <h3>January</h3>
    <p><a href="http://www.sundance.org" target="_blank">Sundance Film Festival</a><br /> Park
City, Utah, USA<br /> 20 - 30 January 2011</p>
    <h3>February</h3>
    <p><a href="http://www.tropfest.com">Tropfest</a><br /> Sydney, Australia<br /> 20 February
2011</p>
    <!-- additional content -->
    <p><a href=" ../.. /about/about.html">About Film Folk</a></p>
    <p><a href="#top">Top of page</a></p>
  </body>
</html>
```

Esercizio: code-04/6-example.html

---



## HTML5: markup immagini

```
<html>
  <head>
    <title>Images</title>
  </head>
  <body>
    <h1></h1>
    <figure>
      
      <p>
        <figcaption>
          This recipe for individual chocolate cakes is so simple and so delectable!
        </figcaption>
      </p>
    </figure>
    <h4>More Recipes:</h4>
    <p>
      
      
      
    </p>
  </body>
</html>
```

Esercizio: code-05/6-example.html

---

## HTML5: immagini per il web



### OTHER SOFTWARE

Adobe Fireworks  
Pixelmator  
PaintShop Pro  
Paint.net

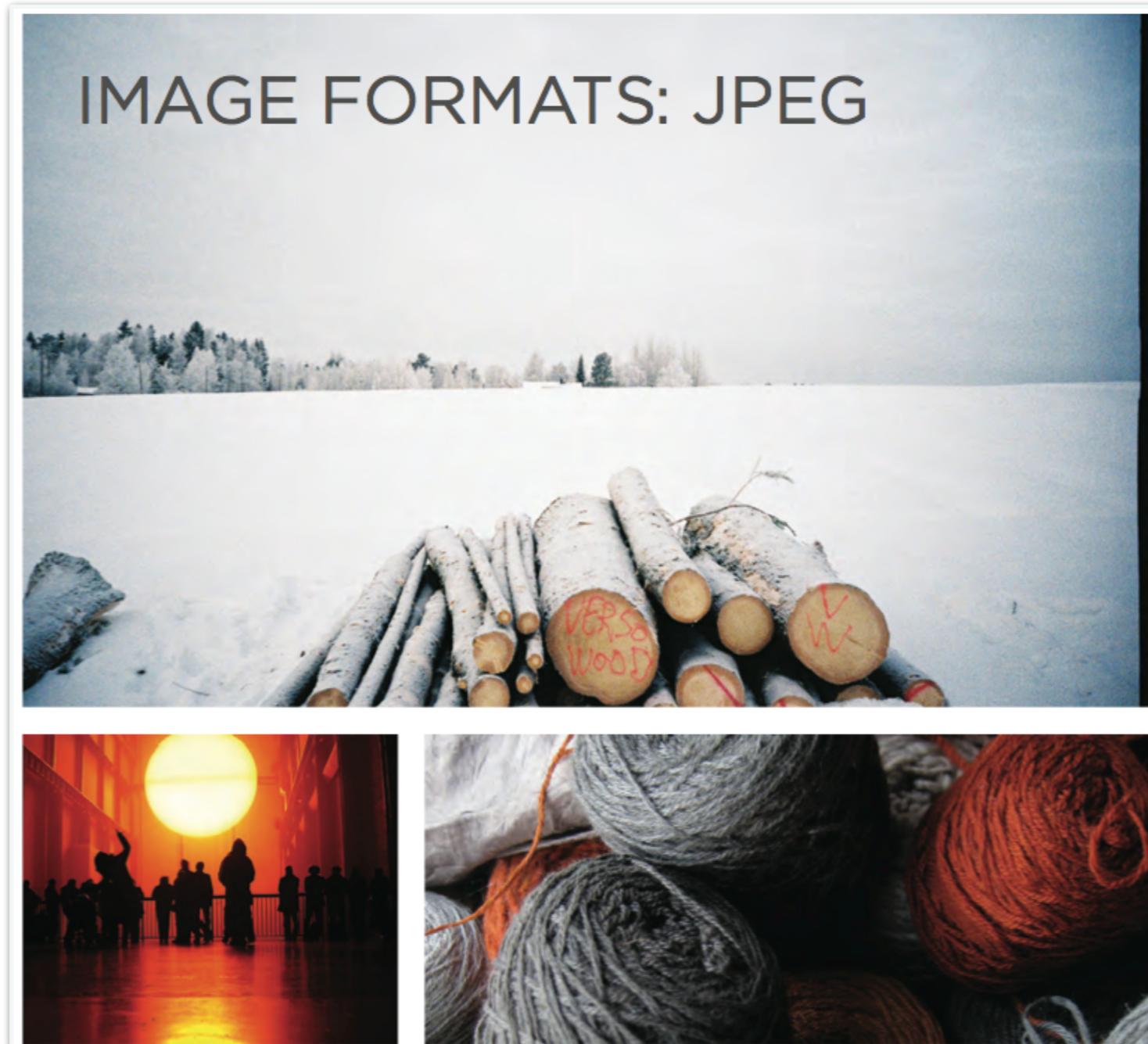
### ONLINE EDITORS

[www.photoshop.com](http://www.photoshop.com)  
[www.pixlr.com](http://www.pixlr.com)  
[www.splashup.com](http://www.splashup.com)  
[www.ipiccy.com](http://www.ipiccy.com)

Photoshop: File > Save for web

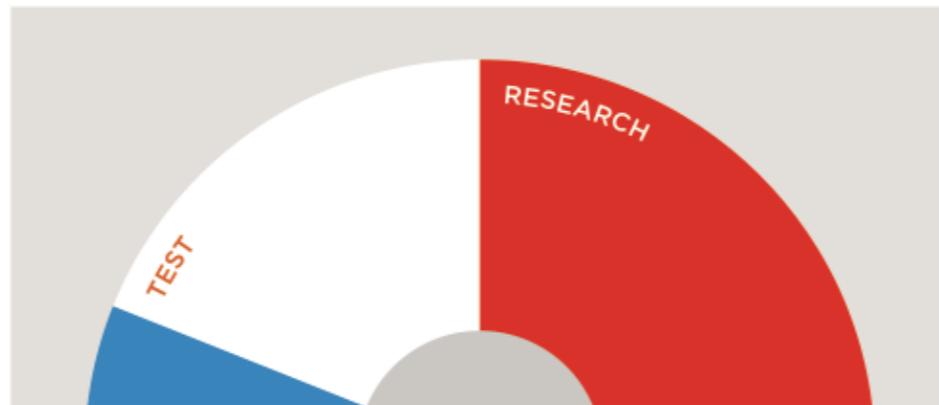
---

## HTML5: immagini per il web



## HTML5: immagini per il web

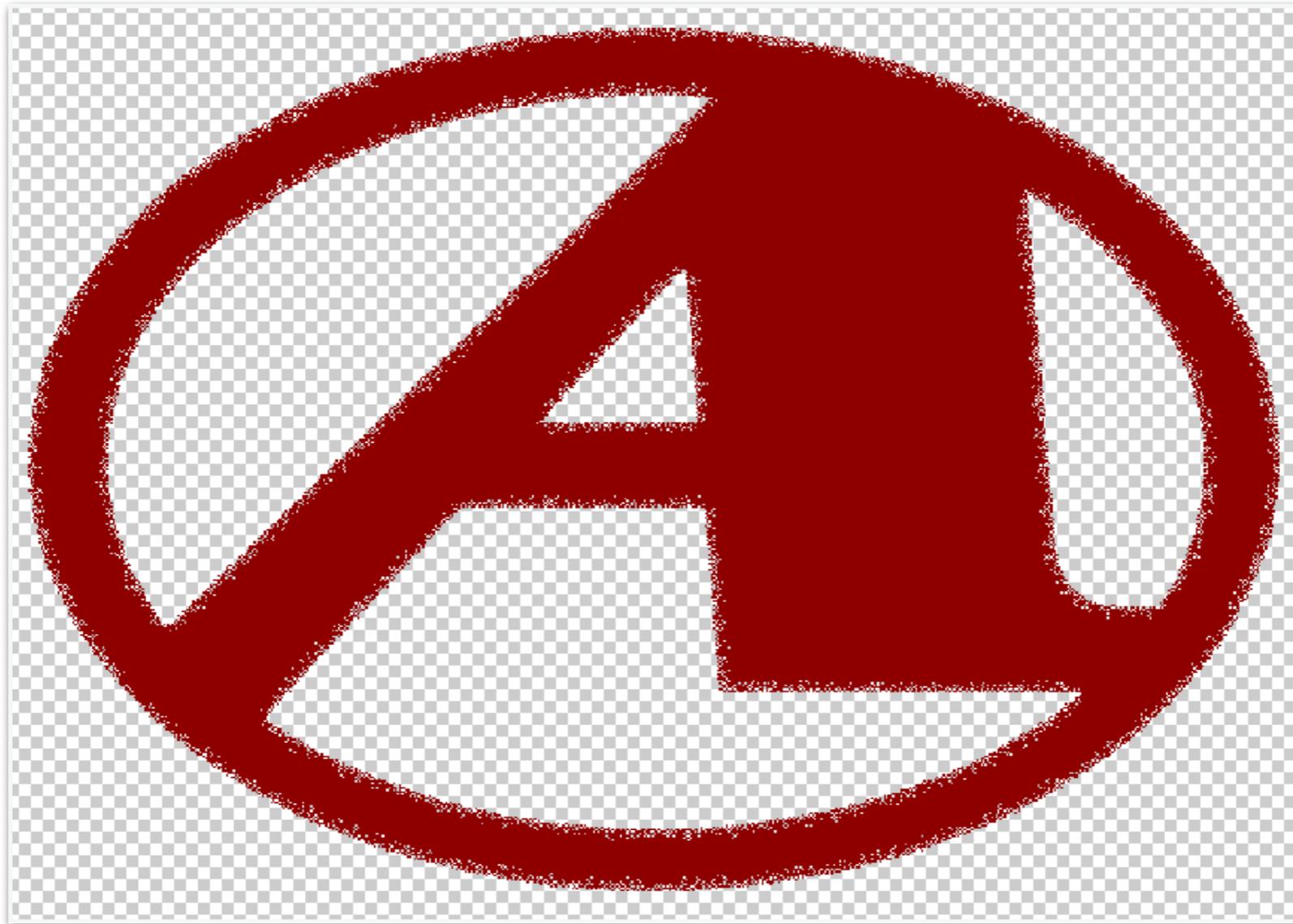
### IMAGE FORMATS: GIF



## HTML5: immagini per il web



## HTML5: immagini per il web - trasparenza



## HTML5: immagini per il web - trasparenza



## HTML5: immagini per il web - immagini vettoriali

Possono essere visualizzate dai browser, la **risoluzione** non dipende dal pixel ma da **formule matematiche**

**Illustrator** > SVG, SWF



# HTML & CSS

---

HTML5: immagini per il web - immagini raster

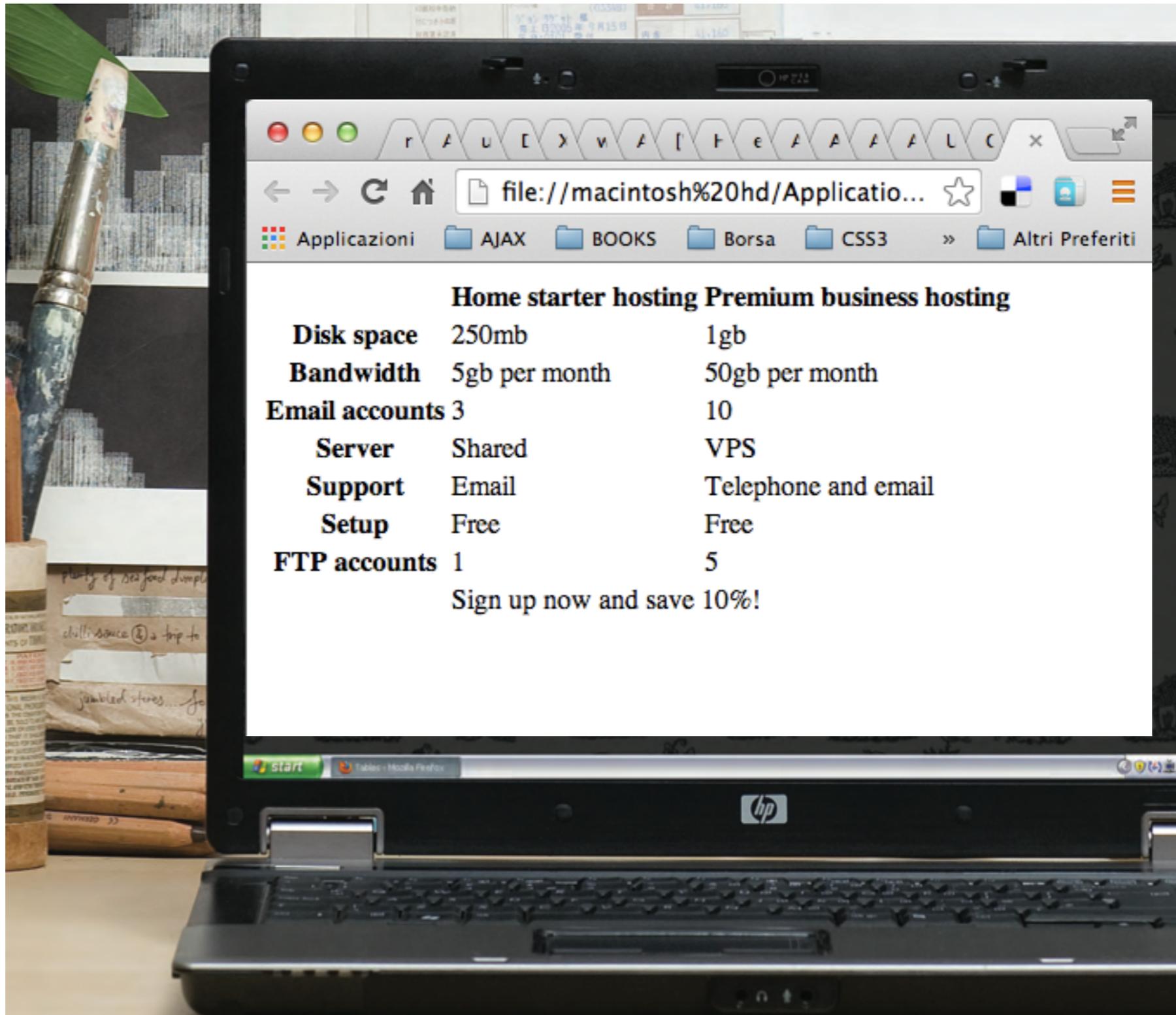
I **display** dei computer visualizzano le immagini a **72ppi**

Stampa: **300ppi**



# Online image editor

- ▶ <https://pixlr.com/>
- ▶ <https://ipiccy.com/>
- ▶ <https://vectr.com/>

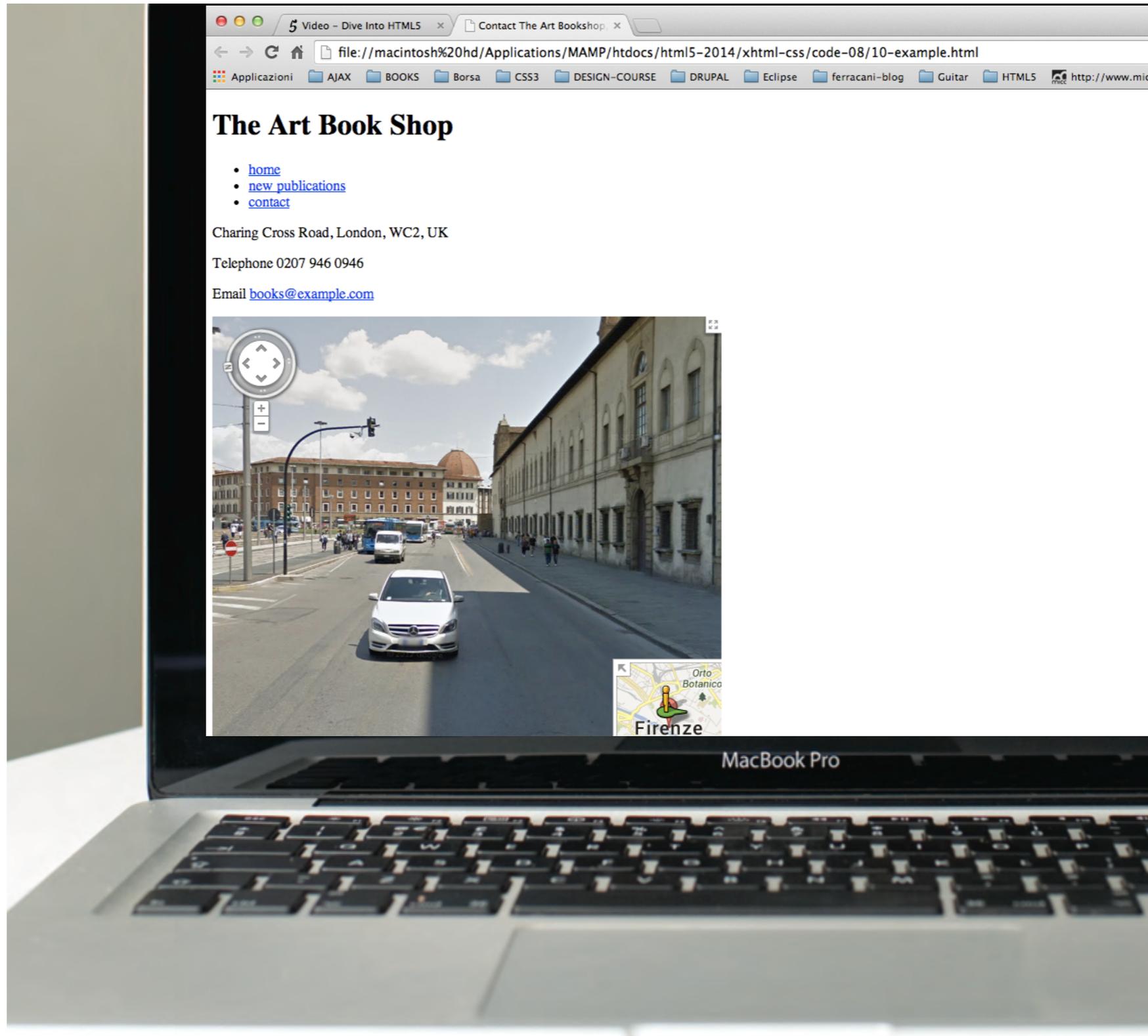


## HTML5: markup tabella

```
<html>
  <head>
    <title>Tables</title>
  </head>
  <body>
    <table>
      <thead>
        <tr>
          <th></th>
          <th scope="col">Home starter hosting</th>
          <th scope="col">Premium business hosting</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <th scope="row">Disk space</th>
          <td>250mb</td>
          <td>1gb</td>
        </tr>
        <!-- more rows like the two above here -->
      </tbody>
      <tfoot>
        <tr><td></td><td colspan="2">Sign up now and save 10%!</td></tr>
      </tfoot>
    </table>
  </body>
</html>
```

Esercizio: code-06/8-example.html

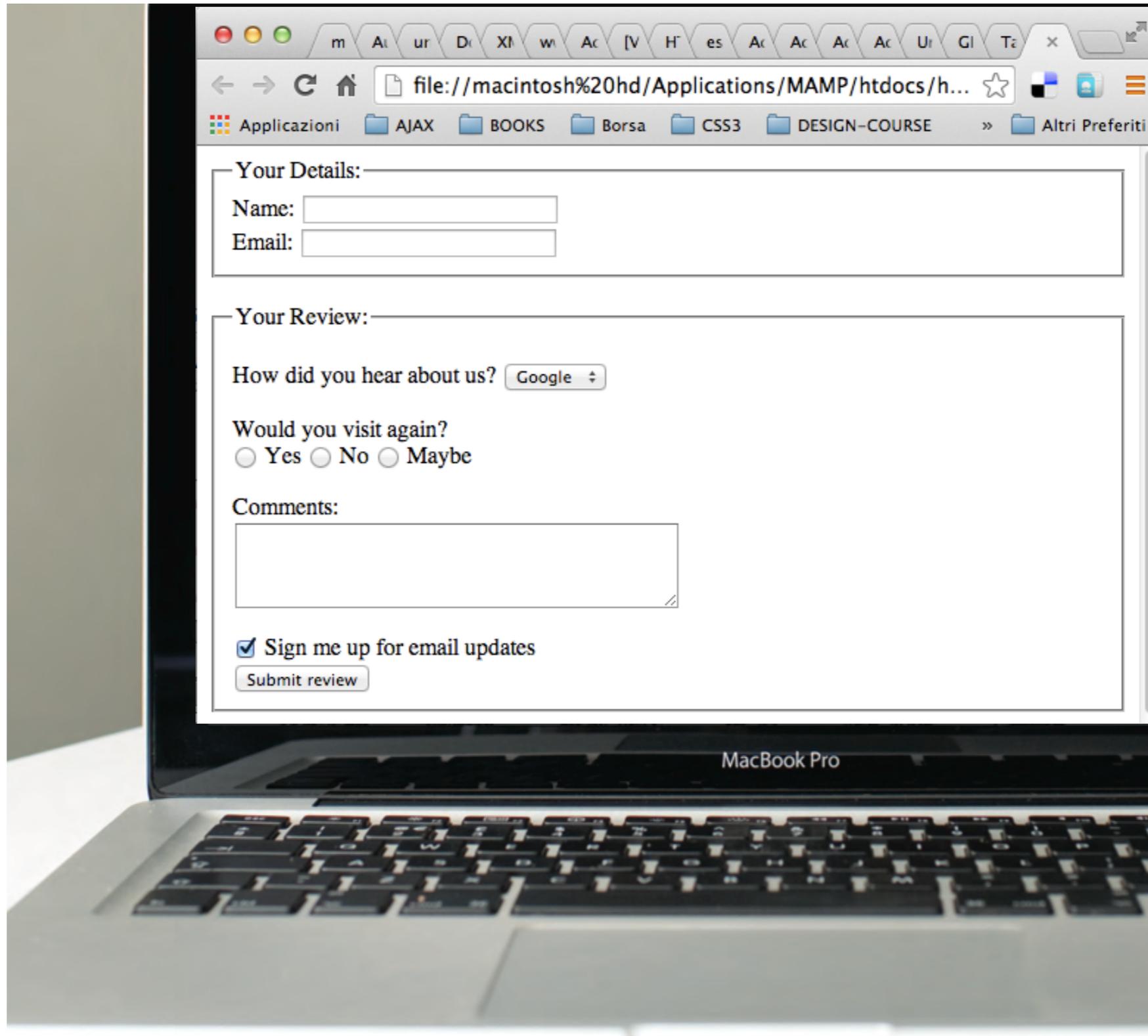
---



## HTML5: struttura

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <meta name="description" content="Telephone, email and directions for The Art Bookshop, London, UK" />
    <title>Contact The Art Bookshop, London UK</title>
  </head>
  <body>
    <div id="header">
      <h1>The Art Book Shop</h1>
      <ul>
        <li><a href=" ../code-08/index.html">home</a></li>
        <li><a href=" ../code-08/index.html">new publications</a></li>
        <li class="current-page"><a href=" ../code-08/index.html">contact</a></li>
      </ul>
    </div><!-- end header -->
    <div id="content">
      <p>Charing Cross Road, London, WC2, UK</p>
      <p><span class="contact">Telephone</span> 0207 946 0946</p>
      <p><span class="contact">Email</span> <a href="mailto:books@example.com">books@example.com</a></p>
      <iframe width="425" height="275" frameborder="0" scrolling="no" marginheight="0" marginwidth="0" src="http://
maps.google.co.uk/maps?
f=q&source=s_q&hl=en&geocode=&q=charing+cross+road+london&output=embed">
      </iframe>
    </div><!-- end content -->
    <p>&copy; The Art Bookshop</p>
  </body>
</html>
```

Esercizio: code-08/8-example.html



## HTML5: tipi di Form

### ADDING TEXT:

#### Text input (single-line)

Used for a single line of text such as email addresses and names.

Username:

#### Password input

Like a single line text box but it masks the characters entered.

Password:

#### Text area (multi-line)

For longer areas of text, such as messages and comments.

### MAKING CHOICES:

#### Radio buttons

For use when a user must select one of a number of options.

Rock  Pop  Jazz

#### Checkboxes

When a user can select and unselect one or more options.

iTunes  Last.fm  Spotify

#### Drop-down boxes

When a user must pick one of a number of options from a list.

### SUBMITTING FORMS:

#### Submit buttons

To submit data from your form to another web page.

#### Image buttons

Similar to submit buttons but they allow you to use an image.

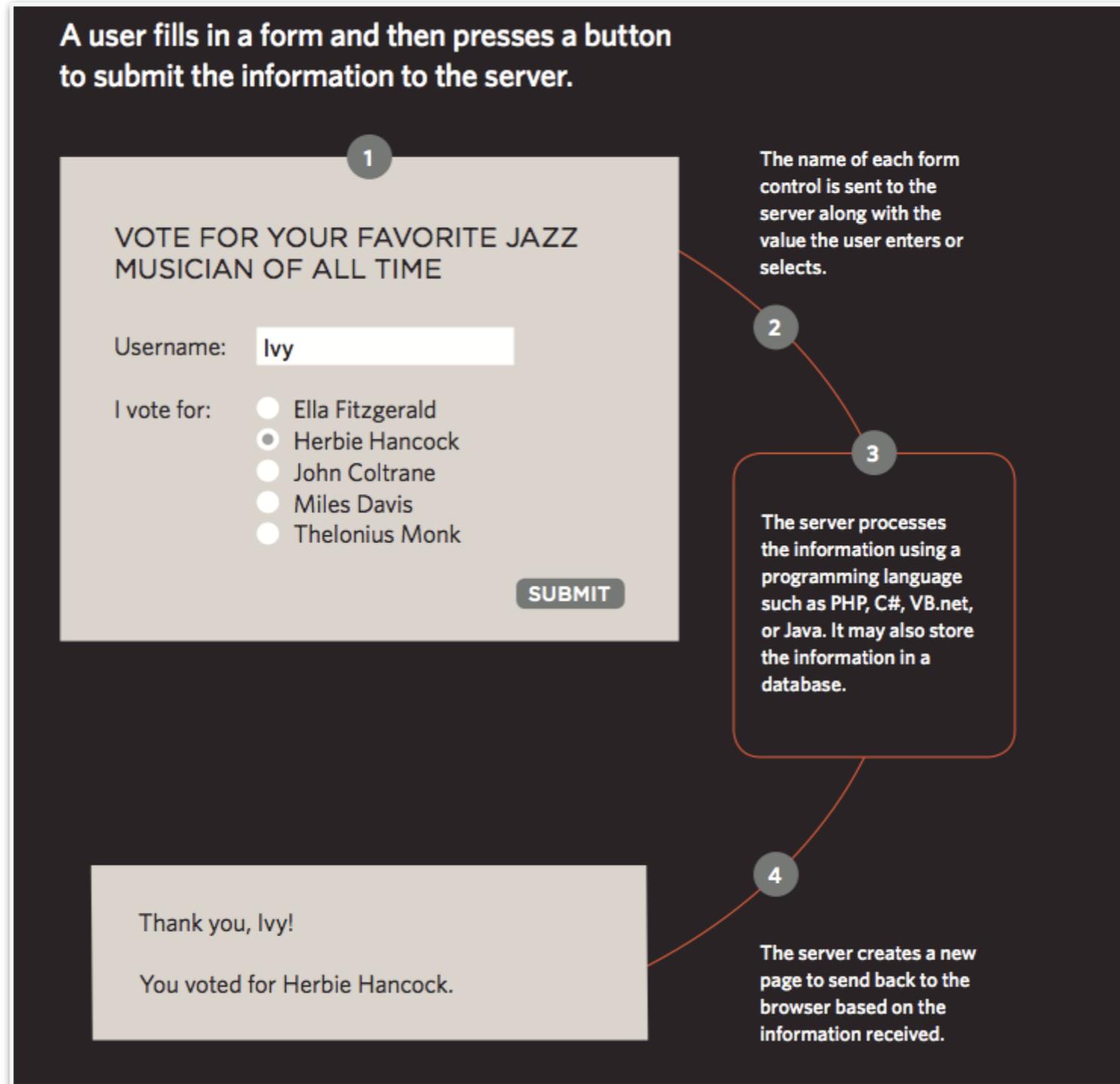
### UPLOADING FILES:

#### File upload

Allows users to upload files (e.g. images) to a website.

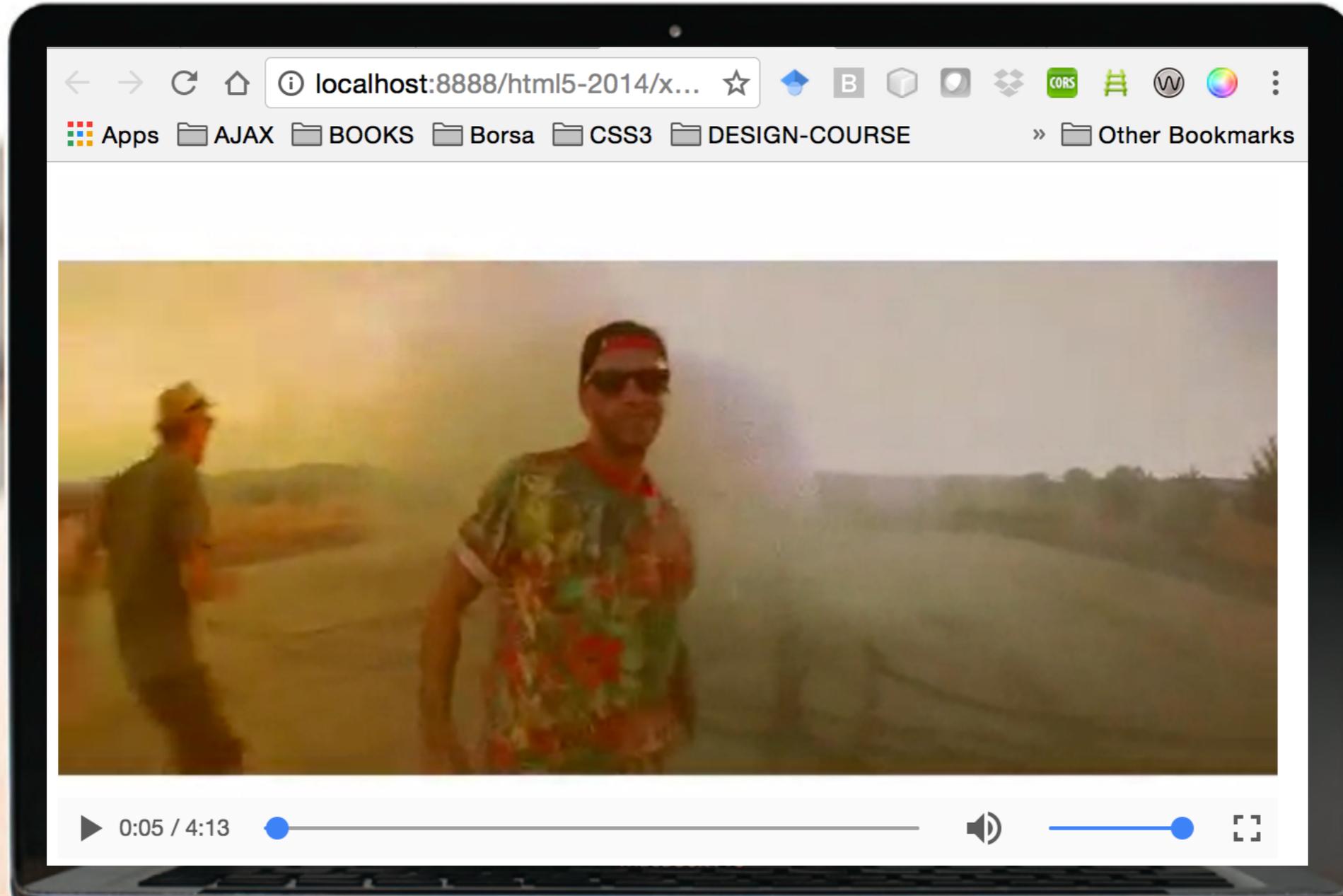
No file chosen

## HTML5: come funzionano le form



## HTML5: markup forms

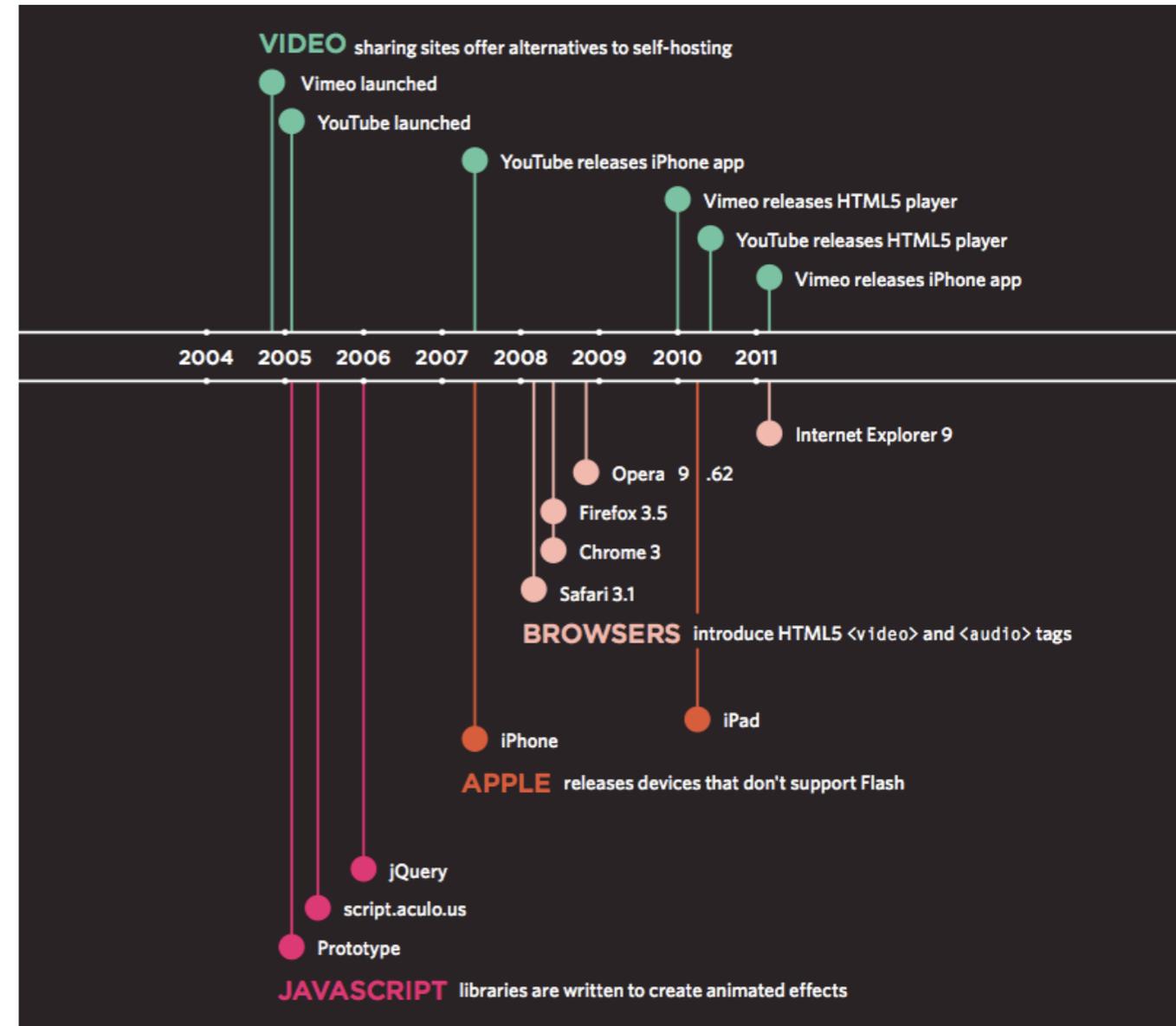
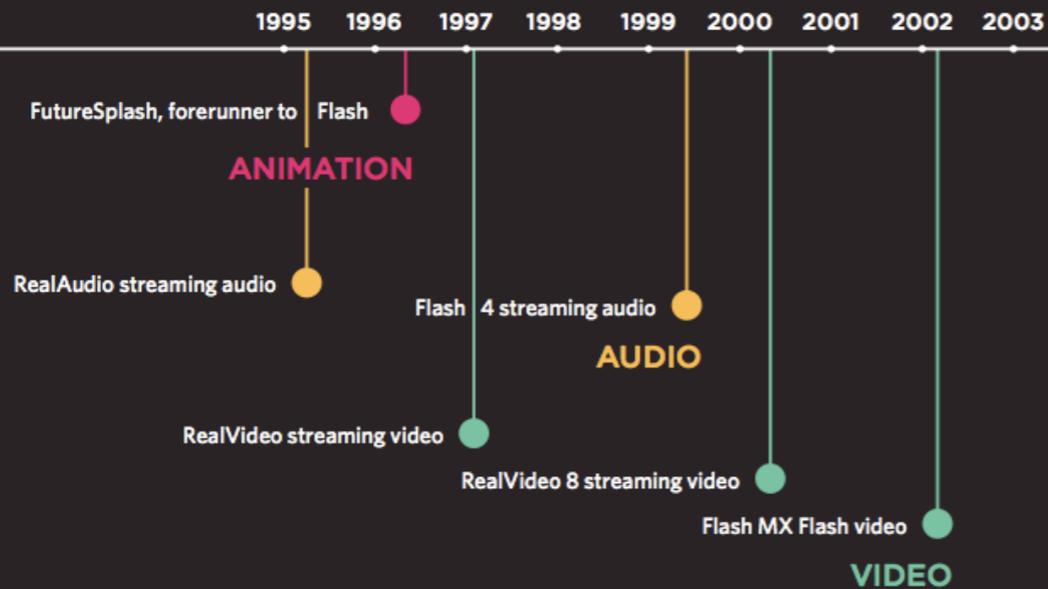
```
<html>
  <head>
    <title>Forms</title>
  </head>
  <body>
    <form action="http://www.example.com/review.php" method="get">
      <fieldset>
        <legend>Your Details:</legend>
        <label>Name: <input type="text" name="name" size="30" maxlength="100"></label><br />
        <label>Email: <input type="email" name="email" size="30" maxlength="100"></label><br />
      </fieldset>
      <fieldset>
        <legend>Your Review:</legend>
        <p>
          <label for="hear-about">How did you hear about us?</label>
          <select name="referrer" id="hear-about">
            <option value="google">Google</option>
            <option value="friend">Friend</option>
            <option value="advert">Advert</option>
            <option value="other">Other</option>
          </select>
        </p>
        <p>
          Would you visit again?<br />
          <label><input type="radio" name="rating" value="yes" /> Yes</label>
          <label><input type="radio" name="rating" value="no" /> No</label>
          <label><input type="radio" name="rating" value="maybe" /> Maybe</label>
        </p>
        <p>
          <label for="comments">Comments:</label><br />
          <textarea rows="4" cols="40" id="comments"></textarea>
        </p>
        <label><input type="checkbox" name="subscribe" checked="checked" /> Sign me up for email updates</label><br />
        <input type="submit" value="Submit review" />
      </fieldset>
    </form>
  </body>
</html>
```



## HTML5: multimedia timeline

### TIMELINE: FLASH, VIDEO & AUDIO

Web technologies change quickly. Here you can see some of the changes in how animation, video, and audio are created on the web.



## HTML5: supporto video nei browser

### VIDEO CODEC SUPPORT IN SHIPPING BROWSERS

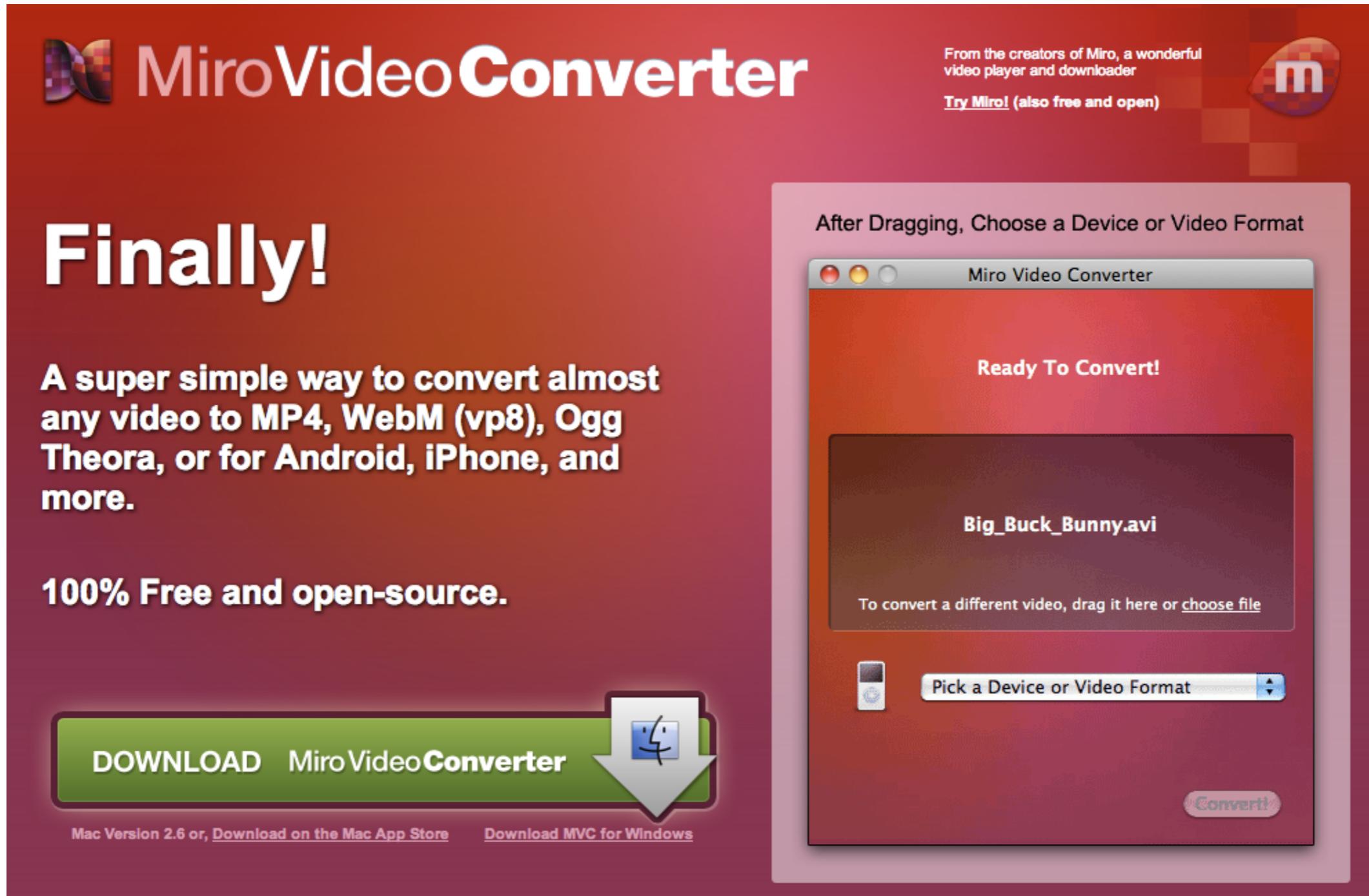
CODECS/CONTAINER	IE	FIREFOX	SAFARI	CHROME	OPERA	IPHONE	ANDROID
Theora+Vorbis+Ogg	.	3.5+	†	5.0+	10.5+	.	.
H.264+AAC+MP4	9.0+	.	3.0+	5.0+‡	.	3.0+	2.0+
WebM	9.0+*	4.0+	†	6.0+	10.6+	.	2.3+

\* Internet Explorer 9 will only support WebM “when the user has installed a VP8 codec”.

† Safari will play anything that QuickTime can play. QuickTime comes pre-installed with H.264/AAC/MP4 support. There are installable third-party plugins that add support for Theora and WebM, but each user needs to install these plugins before Safari will recognize those video formats.

‡ Google Chrome promised to drop support for H.264 in 2011, but it never happened.

HTML5: multimedia - formati [[www.getmiro.com/download/for-osx/](http://www.getmiro.com/download/for-osx/)]



**Miro Video Converter**

From the creators of Miro, a wonderful video player and downloader  
[Try Miro!](#) (also free and open)

# Finally!

A super simple way to convert almost any video to MP4, WebM (vp8), Ogg Theora, or for Android, iPhone, and more.

100% Free and open-source.

**DOWNLOAD Miro Video Converter**

Mac Version 2.6 or, [Download on the Mac App Store](#) [Download MVC for Windows](#)

After Dragging, Choose a Device or Video Format

Ready To Convert!

Big\_Buck\_Bunny.avi

To convert a different video, drag it here or [choose file](#)

Pick a Device or Video Format

Convert!

## HTML5: esempio video

```
<!DOCTYPE html>
<html>
  <head>
    <title>Flash, Video and Audio</title>
    <script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/swfobject/2.2/swfobject.js"></script>
    <script type="text/javascript">
      var flashvars = {};
      var params = {movie: "../video/puppy.flv"};
      swfobject.embedSWF("flash/osplayer.swf", "snow", "400", "320", "8.0.0", flashvars, params);
    </script>
  </head>
  <body>
    <video poster="images/puppy.jpg" width="400" height="320" controls autoplay loop>
      <source src="video/puppy.mp4" type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"' />
      <source src="video/puppy.webm" type='video/webm; codecs="vp8, vorbis"' />
      <source src="video/puppy.ogv" type="video/ogg; codecs=theora,vorbis">
      <div id="snow">
        <p>You cannot see this video of a puppy playing in the snow because this browser does not support our
video formats.</p>
      </div>
    </video>
  </body>
</html>
```

# HTML & CSS

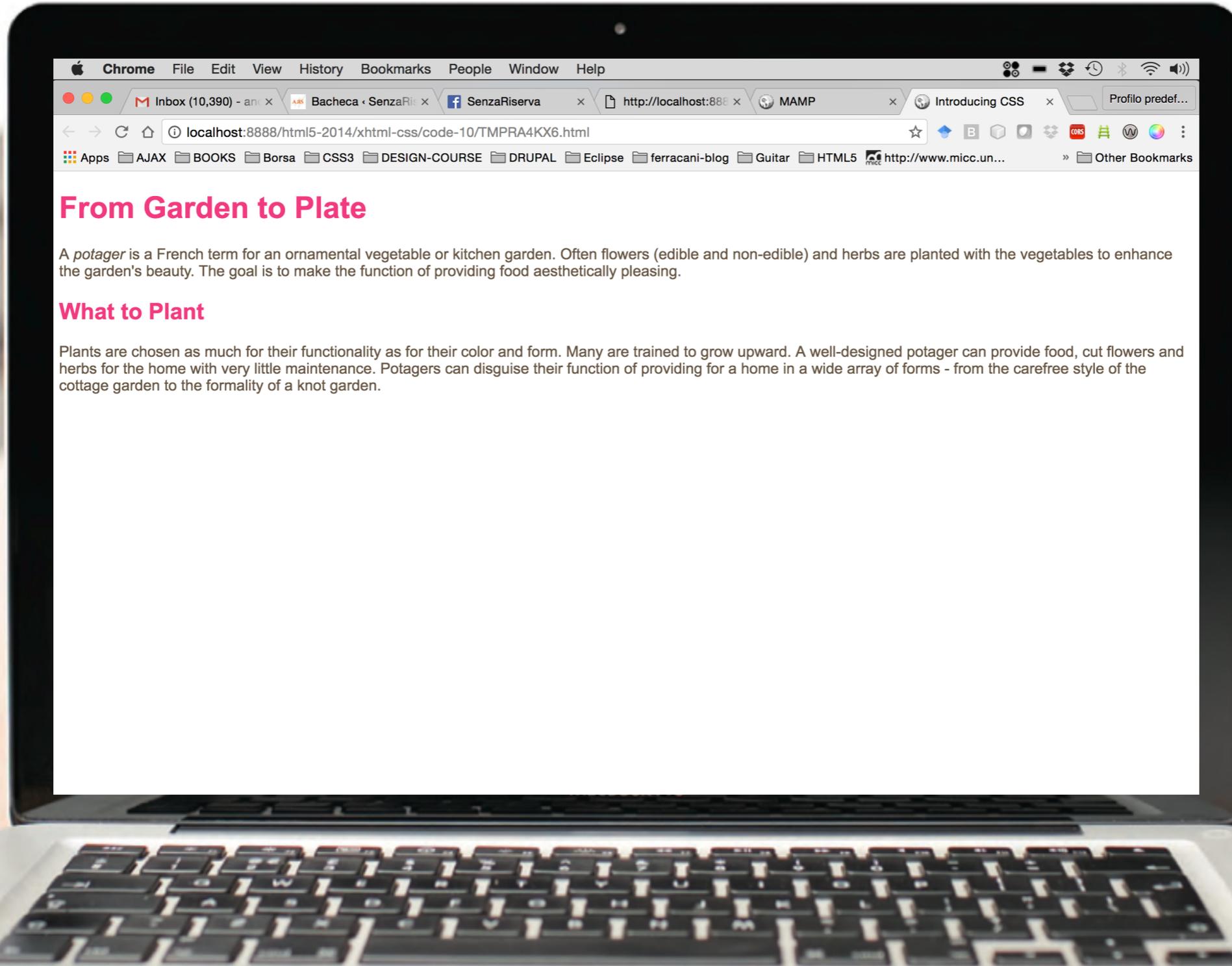
---

## HTML5: esempio audio

```
<!DOCTYPE html>
<html>
  <head>
    <title>Multiple Audio Sources</title>
  </head>
  <body>
    <audio controls autoplay>
      <source src="audio/test-audio.ogg" />
      <source src="audio/test-audio.mp3" />
      <p>This browser does not support our audio format.</p>
    </audio>
  </body>
</html>
```

Esercizio: [code-09/7-multiple-audio-sources.html](#)

---



# HTML & CSS

---

## CSS: esempio CSS

```
<!DOCTYPE html>
<html>
<head>
  <title>Introducing CSS</title>
  <link href="css/example.css" type="text/css" rel="stylesheet" />
</head>
<body>
  <h1>From Garden to Plate</h1>
  <p>A <em>potager</em> is a French term for an ornamental vegetable or kitchen garden ... </p>
  <h2>What to Plant</h2>
  <p>Plants are chosen as much for their functionality as for their color and form ... </p>
</body>
</html>
```

example.css:

```
body {
  font-family: Arial, Verdana, sans-serif;
}
h1, h2 {
  color: #ee3e80;
}
p {
  color: #665544;
}
```

Esercizio: code-10/example.html

---

## CSS ASSOCIATES STYLE RULES WITH HTML ELEMENTS

CSS works by associating rules with HTML elements. These rules govern how the content of specified elements should be displayed. A CSS rule contains two parts: a **selector** and a **declaration**.

SELECTOR



`p {`

`font-family: Arial;}`



DECLARATION

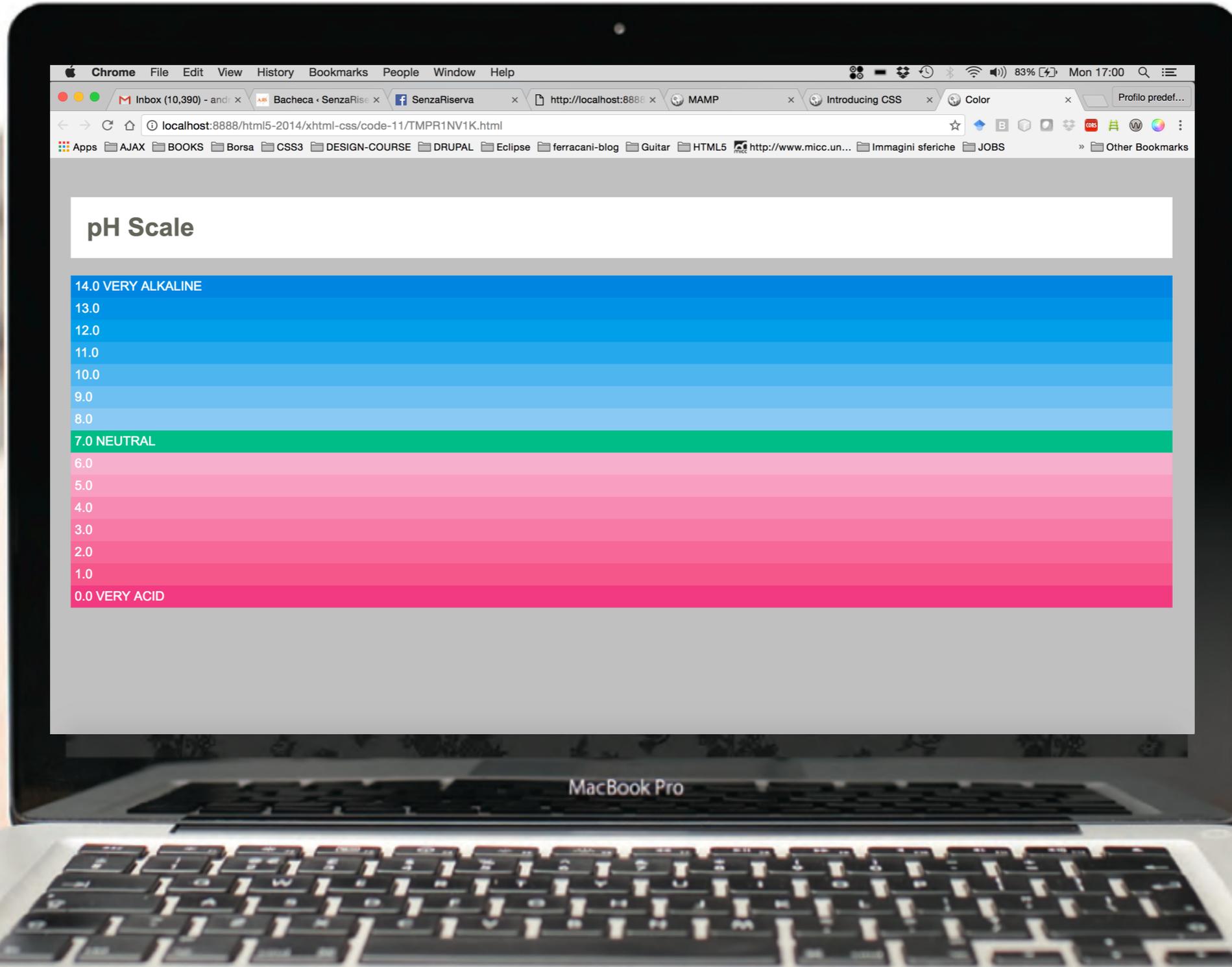
# CSS PROPERTIES AFFECT HOW ELEMENTS ARE DISPLAYED

CSS declarations sit inside curly brackets and each is made up of two parts: a **property** and a **value**, separated by a colon. You can specify several properties in one declaration, each separated by a semi-colon.

```
h1, h2, h3 {  
    font-family: Arial;  
    color: yellow;}  
    └──┬──┘ └──┬──┘  
    PROPERTY  VALUE
```

UNIVERSAL SELECTOR	Applies to all elements in the document	<code>* {}</code> Targets all elements on the page
TYPE SELECTOR	Matches element names	<code>h1, h2, h3 {}</code> Targets the <code>&lt;h1&gt;</code> , <code>&lt;h2&gt;</code> and <code>&lt;h3&gt;</code> elements
CLASS SELECTOR	Matches an element whose <code>class</code> attribute has a value that matches the one specified after the period (or full stop) symbol	<code>.note {}</code> Targets any element whose <code>class</code> attribute has a value of <code>note</code> <code>p.note {}</code> Targets only <code>&lt;p&gt;</code> elements whose <code>class</code> attribute has a value of <code>note</code>
ID SELECTOR	Matches an element whose <code>id</code> attribute has a value that matches the one specified after the pound or hash symbol	<code>#introduction {}</code> Targets the element whose <code>id</code> attribute has a value of <code>introduction</code>
CHILD SELECTOR	Matches an element that is a direct child of another	<code>li&gt;a {}</code> Targets any <code>&lt;a&gt;</code> elements that are children of an <code>&lt;li&gt;</code> element (but not other <code>&lt;a&gt;</code> elements in the page)
DESCENDANT SELECTOR	Matches an element that is a descendent of another specified element (not just a direct child of that element)	<code>p a {}</code> Targets any <code>&lt;a&gt;</code> elements that sit inside a <code>&lt;p&gt;</code> element, even if there are other elements nested between them
ADJACENT SIBLING SELECTOR	Matches an element that is the next sibling of another	<code>h1+p {}</code> Targets the first <code>&lt;p&gt;</code> element after any <code>&lt;h1&gt;</code> element (but not other <code>&lt;p&gt;</code> elements)
GENERAL SIBLING SELECTOR	Matches an element that is a sibling of another, although it does not have to be the directly preceding element	<code>h1~p {}</code> If you had two <code>&lt;p&gt;</code> elements that are siblings of an <code>&lt;h1&gt;</code> element, this rule would apply to both

SELECTOR	MEANING	EXAMPLE
EXISTENCE	[ ] Matches a specific attribute (whatever its value)	p[class] Targets any <p> element with an attribute called class
EQUALITY	[=] Matches a specific attribute with a specific value	p[class="dog"] Targets any <p> element with an attribute called class whose value is dog
SPACE	[~= Matches a specific attribute whose value appears in a space-separated list of words	p[class~="dog"] Targets any <p> element with an attribute called class whose value is a list of space-separated words, one of which is dog
PREFIX	[^= Matches a specific attribute whose value begins with a specific string	p[attr^="d"] Targets any <p> element with an attribute whose value begins with the letter "d"
SUBSTRING	[*= Matches a specific attribute whose value contains a specific substring	p[attr*"do"] Targets any <p> element with an attribute whose value contains the letters "do"
SUFFIX	[\$= Matches a specific attribute whose value ends with a specific string	p[attr\$"g"] Targets any <p> element with an attribute whose value ends with the letter "g"



# HTML & CSS

## CSS: esempio CSS

```
<!DOCTYPE html>
<html>
  <head>
    <title>Color</title>
    <style type="text/css">
      body {
        background-color: silver;
        color: white;
        padding: 20px;
        font-family: Arial, Verdana, sans-serif;}

      body p.seven {
        background-color: rgb(238,62,128);
        padding: 2px;}

      p#zero {
        background-color: #ffffff;
        background-color: hsla(0,100%,100%,0.5);
        color: #64645A;
        padding: 10px;}

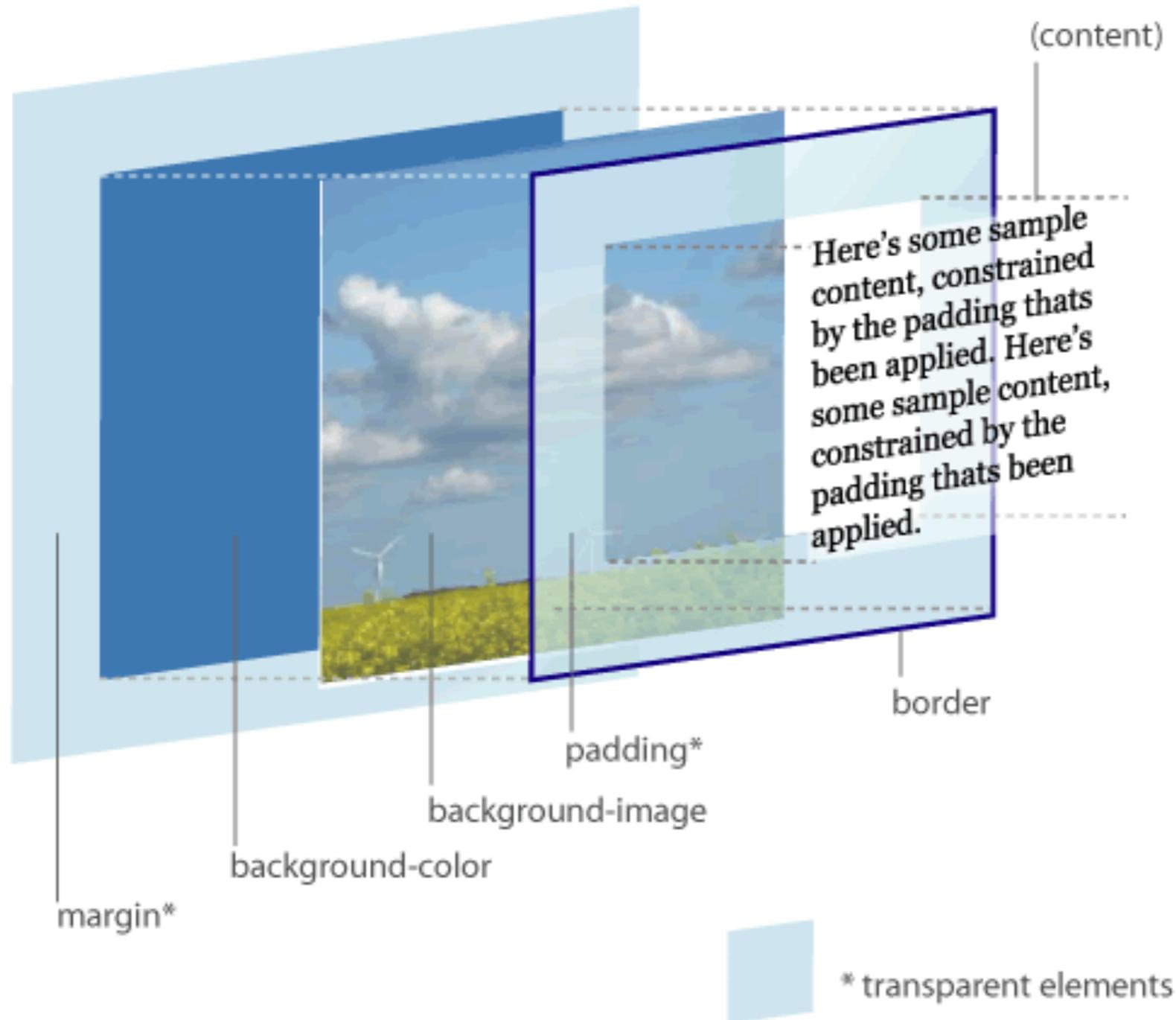
      p {
        padding: 5px;
        margin: 0px;}

      div p span.notice {}

      #zero {color:#FF0000;}
```

```
<body>
  <h1>pH Scale</h1>
  <p class="fourteen">14.0 VERY
ALKALINE</p>
  <p class="thirteen">13.0</p>
  <p class="twelve">12.0</p>
  <p class="eleven">11.0</p>
  <p class="ten">10.0</p>
  <p class="nine">9.0</p>
  <p class="seven">8.0</p>
  <p class="seven">7.0 NEUTRAL</p>
  <p class="six">6.0</p>
  <p class="five">5.0</p>
  <p class="four">4.0</p>
  <p class="three">3.0</p>
  <p class="zero">2.0</p>
  <p class="zero">1.0</p>
  <p id="zero">0.0 VERY ACID</p>
</body>
</html>
```

### THE CSS BOX MODEL HIERARCHY



## CSS: boxmodel

Every box has three available properties that can be adjusted to control its appearance:

1

### BORDER

Every box has a border (even if it is not visible or is specified to be 0 pixels wide). The border separates the edge of one box from another.

If you specify a width for a box, then the borders, margin, and padding are added to its width and height.

2

### MARGIN

Margins sit outside the edge of the border. You can set the width of a margin to create a gap between the borders of two adjacent boxes.

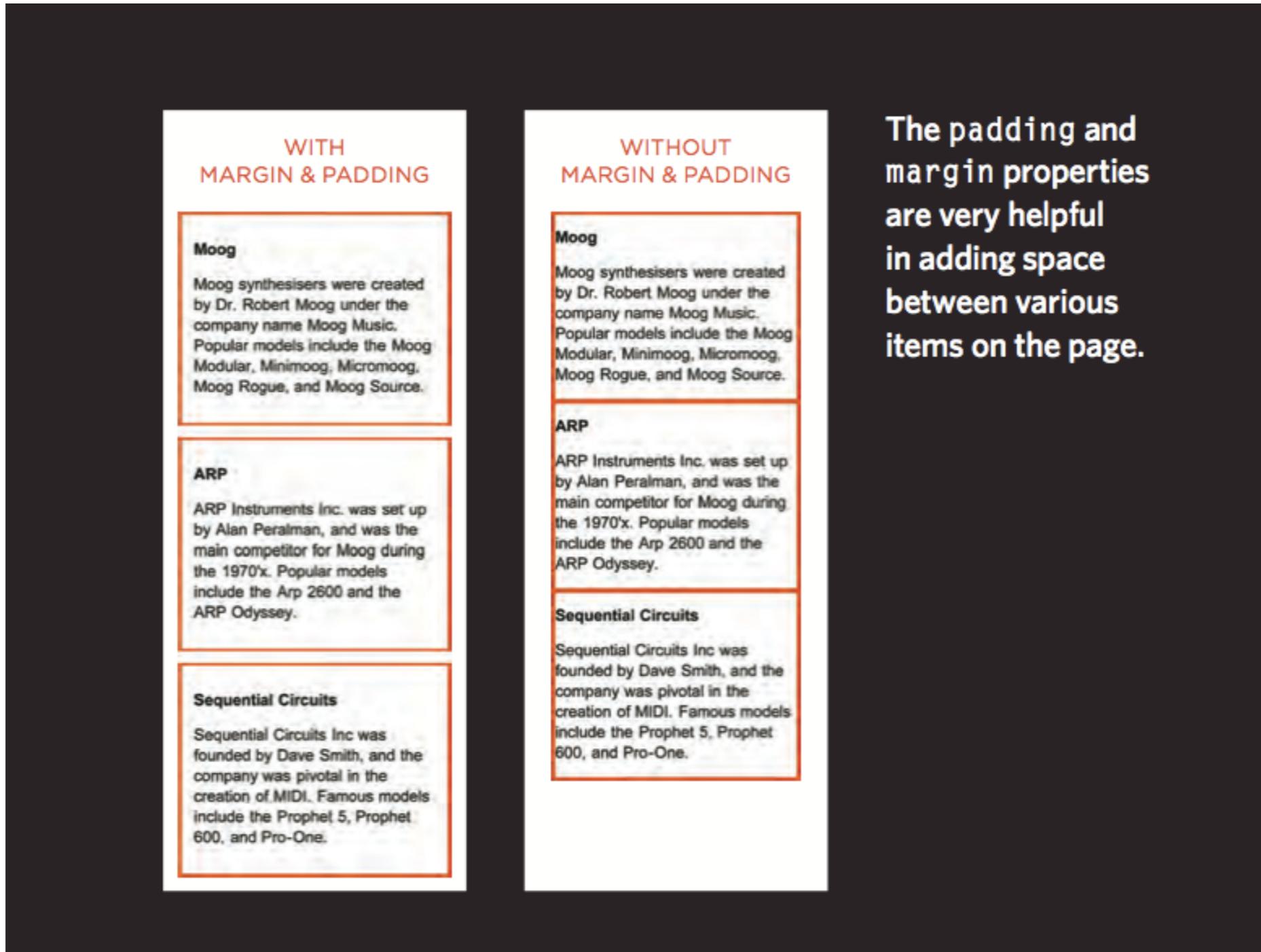
3

### PADDING

Padding is the space between the border of a box and any content contained within it. Adding padding can increase the readability of its contents.



## CSS: boxmodel



**WITH  
MARGIN & PADDING**

**Moog**  
Moog synthesisers were created by Dr. Robert Moog under the company name Moog Music. Popular models include the Moog Modular, Minimoog, Micromoog, Moog Rogue, and Moog Source.

**ARP**  
ARP Instruments Inc. was set up by Alan Peralman, and was the main competitor for Moog during the 1970's. Popular models include the Arp 2600 and the ARP Odyssey.

**Sequential Circuits**  
Sequential Circuits Inc was founded by Dave Smith, and the company was pivotal in the creation of MIDI. Famous models include the Prophet 5, Prophet 600, and Pro-One.

**WITHOUT  
MARGIN & PADDING**

**Moog**  
Moog synthesisers were created by Dr. Robert Moog under the company name Moog Music. Popular models include the Moog Modular, Minimoog, Micromoog, Moog Rogue, and Moog Source.

**ARP**  
ARP Instruments Inc. was set up by Alan Peralman, and was the main competitor for Moog during the 1970's. Popular models include the Arp 2600 and the ARP Odyssey.

**Sequential Circuits**  
Sequential Circuits Inc was founded by Dave Smith, and the company was pivotal in the creation of MIDI. Famous models include the Prophet 5, Prophet 600, and Pro-One.

**The padding and margin properties are very helpful in adding space between various items on the page.**

## CSS: flusso della pagina html

CSS treats each HTML element as if it is in its own box. This box will either be a **block-level box** or an **inline box**.

Block-level boxes start on a new line and act as the main building blocks of any layout, while inline boxes flow between surrounding text. You can control how much space each box takes up by setting the width of the boxes (and sometimes the height, too). To separate boxes, you can use borders, margins, padding, and background colors.

### BLOCK-LEVEL ELEMENTS START ON A NEW LINE

Examples include:

`<h1>` `<p>` `<ul>` `<li>`

#### Lorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit.

- Lorem ipsum dolor sit
- Consectetur adipisicing
- Elit, sed do eiusmod

### INLINE ELEMENTS FLOW IN BETWEEN SURROUNDING TEXT

Examples include:

`<img>` `<b>` `<i>`

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut **labore et dolore** magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

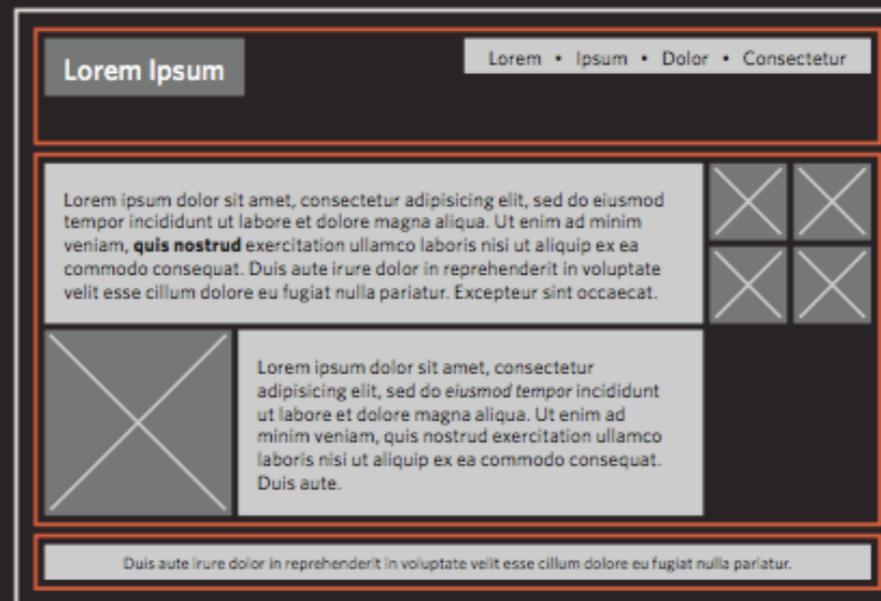


Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

## CSS: flusso della pagina html

**If one block-level element sits inside another block-level element then the outer box is known as the containing or parent element.**

It is common to group a number of elements together inside a `<div>` (or other block-level) element. For example, you might group together all of the elements that form the header of a site (such as the logo and the main navigation). The `<div>` element that contains this group of elements is then referred to as the containing element.



**A box may be nested inside several other block-level elements. The containing element is always the direct parent of that element.**

The orange lines in this diagram represent `<div>` elements. The header (containing the logo and navigation) are in one `<div>` element, the main content of the page is in another, and the footer is in a third. The `<body>` element is the containing element for these three `<div>` elements. The second `<div>` element is the containing element for two paragraphs of Latin text and images (represented by crossed squares).

# HTML & CSS

## CSS: esercizio

font: Helvetica

colore del testo: #665544;

colore bordo box: #e7642c;

distanza box principale altri box:  
20px;

distanza box secondari testo/bordo:  
20px;

colore bordo corsivi e link: #7fcbae;  
distanza corsivi e link dal bordo: 5px;

## The Cottage Garden

The cottage garden is a distinct style of garden that uses an informal design, dense plantings, and a mixture of ornamental and edible plants.

The Cottage Garden originated in England and its history can be traced back for centuries, although they were re-invented in 1870's England, when stylized versions were formed as a reaction to the more structured and rigorously maintained English estate gardens.

The earliest cottage gardens were more practical than their modern descendants, with an emphasis on vegetables and herbs, along with some fruit trees.

## CSS: esercizio - soluzione

```
<style type="text/css">
```

```
body {  
  font-family: Helvetica, Arial, sans-serif;  
  color: #665544;  
  border: 2px solid #e7642c;  
  padding: 20px;  
}
```

```
h1, h2, p {  
  border: 2px solid #e7642c;  
  padding: 10px;  
  line-height: 2em;  
}
```

```
i, a, a:hover, a:visited {  
  color: #665544;  
  border: 2px solid #7fcbae;  
  padding: 5px;}
```

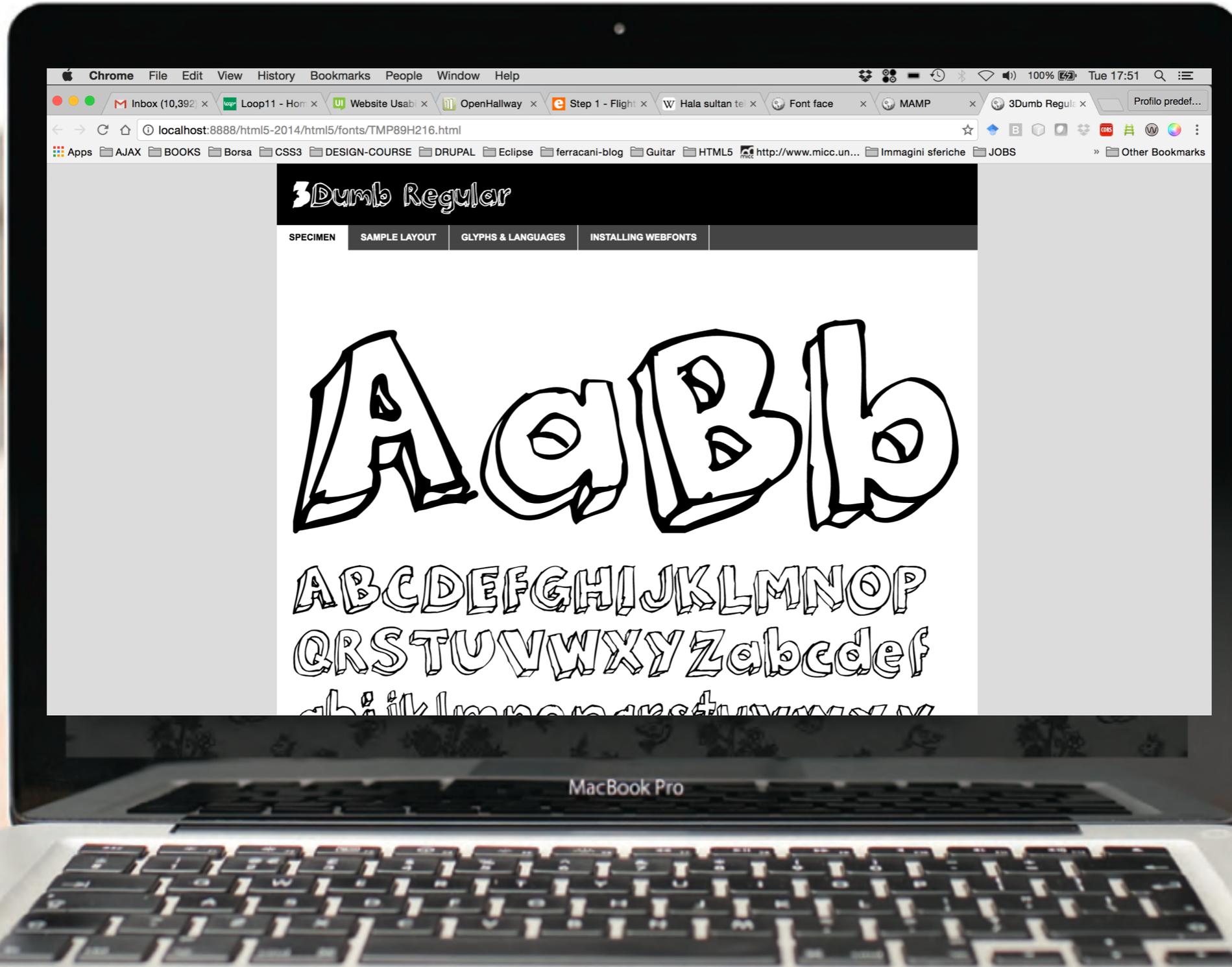
```
</style>
```

## The Cottage Garden

The cottage garden is a distinct style of garden that uses an informal design, dense plantings, and a mixture of ornamental and edible plants.

The Cottage Garden originated in England and its history can be traced back for centuries, although they were re-invented in 1870's England, when stylized versions were formed as a reaction to the more structured and rigorously maintained English estate gardens.

The earliest cottage gardens were more practical than their modern descendants, with an emphasis on vegetables and herbs, along with some fruit trees.



# HTML & CSS

## CSS: webfonts

[www.fontsquirrel.com](http://www.fontsquirrel.com)

[www.fontex.org](http://www.fontex.org)

[www.openfontlibrary.org](http://www.openfontlibrary.org)

[www.typekit.com](http://www.typekit.com)

[www.kernest.com](http://www.kernest.com)

[www.fontspring.com](http://www.fontspring.com)

CSS
chapter-12/understanding-font-formats.html

```

@font-face {
  font-family: 'ChunkFiveRegular';
  src: url('fonts/chunkfive.eot');
  src: url('fonts/chunkfive.eot?#iefix')
    format('embedded-opentype'),
    url('fonts/chunkfive.woff') format('woff'),
    url('fonts/chunkfive.ttf')
    format('truetype'),
    url('fonts/chunkfive.svg#ChunkFiveRegular')
    format('svg');}
        
```

BROWSER	FORMAT			
	eot	woff	ttf / otf	svg
Chrome (all)				●
Chrome 6+		●	●	●
Firefox 3.5			●	
Firefox 3.6+		●	●	
IE 5 - 8	●			
IE 9+	●	●	●	
Opera 10+			●	●
Safari 3.1+			●	●
iOS <4.2				●
iOS 4.2+			●	●

## CSS: posizionamenti

CSS has the following **positioning schemes** that allow you to control the layout of a page: normal flow, relative positioning, and absolute positioning. You specify the positioning scheme using the `position` property in CSS. You can also float elements using the `float` property.

### **NORMAL FLOW**

Every block-level element appears on a new line, causing each item to appear lower down the page than the previous one. Even if you specify the width of the boxes and there is space for two elements to sit side-by-side, they will not appear next to each other. This is the default behavior (unless you tell the browser to do something else).

#### **Lorem Ipsum**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Duis aute irure dolor in reprehenderit in voluptate velit.

The paragraphs appear one after the other, vertically down the page.

### **RELATIVE POSITIONING**

This moves an element from the position it would be in normal flow, shifting it to the top, right, bottom, or left of where it would have been placed. This does not affect the position of surrounding elements; they stay in the position they would be in in normal flow.

#### **Lorem Ipsum**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea. Duis aute irure dolor in reprehenderit in voluptate velit.

The second paragraph has been pushed down and right from where it would otherwise have been in normal flow.

### **ABSOLUTE POSITIONING**

This positions the element in relation to its containing element. It is taken out of normal flow, meaning that it does not affect the position of any surrounding elements (as they simply ignore the space it would have taken up). Absolutely positioned elements move as users scroll up and down the page.

#### **Lorem Ipsum**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Duis aute irure dolor in reprehenderit in voluptate velit.

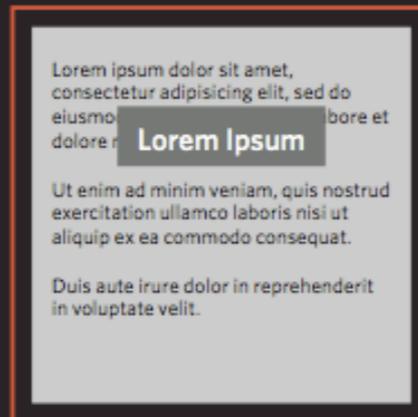
The heading is positioned to the top right, and the paragraphs start at the top of the screen (as if the heading were not there).

## CSS: posizionamenti

To indicate where a box should be positioned, you may also need to use **box offset** properties to tell the browser how far from the top or bottom and left or right it should be placed. (You will meet these when we introduce the positioning schemes on the following pages.)

### FIXED POSITIONING

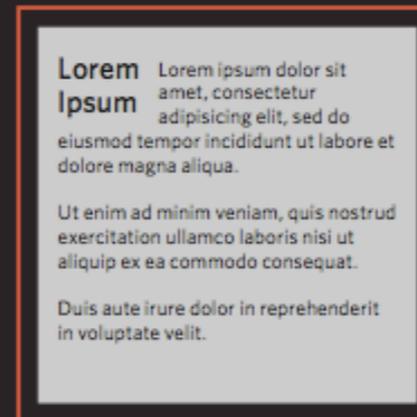
This is a form of absolute positioning that positions the element in relation to the browser window, as opposed to the containing element. Elements with fixed positioning do not affect the position of surrounding elements and they do not move when the user scrolls up or down the page.



The heading has been placed in the center of the page and 25% from the top of the screen. (The rest appears in normal flow.)

### FLOATING ELEMENTS

Floating an element allows you to take that element out of normal flow and position it to the far left or right of a containing box. The floated element becomes a block-level element around which other content can flow.



The heading has been floated to the left, allowing the paragraphs of text to flow around it.

When you move any element from normal flow, boxes can overlap. The **z-index** property allows you to control which box appears on top.

### Posizionamenti in 10 passi

Proprietà CSS per i posizionamenti: `position:static`, `position:relative`, `position:absolute`, `float`.

1 2 3 4 5 6 7 8 9

#### 1. `position:static`

La posizione di default di tutti gli elementi è `position:static`, che significa che l'elemento non è posizionato e segue il normale flusso della pagina

Normalmente non si specifica se non per sovrascrivere una regola precedentemente applicata

```
#div-1 {  
  position:static;  
}
```

id = div-before

id = div-1

id = div-1a

Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Integer pretium dui sit amet felis. Integer sit amet diam.  
Phasellus ultrices viverra velit.

id = div-1b

Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Integer pretium dui sit amet felis. Integer sit amet diam.  
Phasellus ultrices viverra velit. Nam mattis, arcu ut bibendum  
commodo, magna nisi tincidunt tortor, quis accumsan augue  
ipsum id lorem.

id = div-1c

id = div-after

## Risoluzioni

**Different visitors to your site will have different sized screens that show different amounts of information, so your design needs to be able to work on a range of different sized screens.**



**iPhone 4**

Size: 3.5 inches

Resolution: 960 x 640 pixels



**iPad 2**

Size: 9.7 inches

Resolution: 1024 x 768 pixels

---

## Risoluzioni

Resolution refers to the number of dots a screen shows per inch. Some devices have a higher resolution than desktop computers and most operating systems allow users to adjust the resolution of their screens.

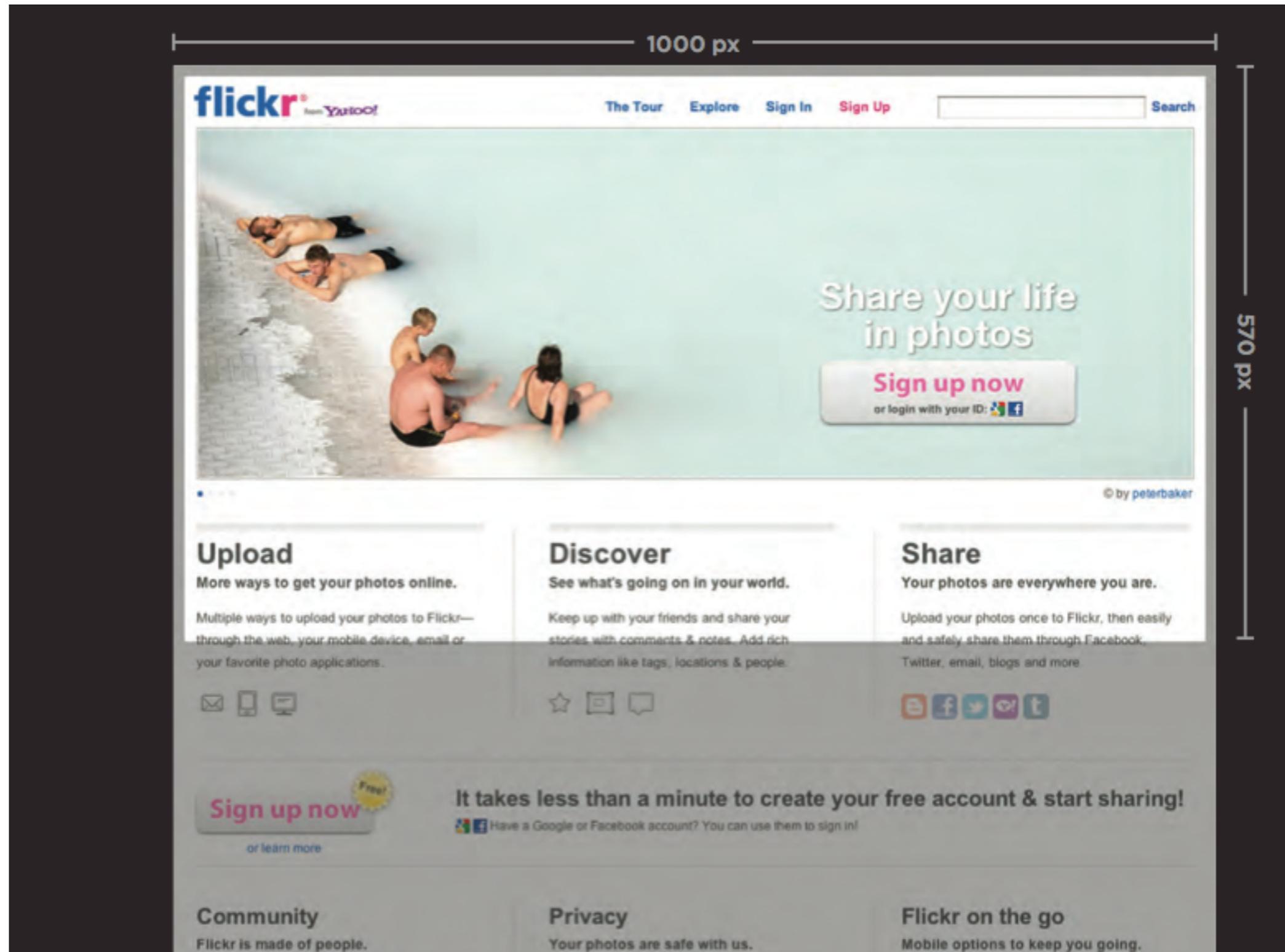


**13" MacBook**  
Size: 13.3 inches  
Resolution: 1280 x 800 pixels



**27" iMac**  
Size: 27 inches  
Resolution: 2560 x 1440 pixels

## Risoluzioni



## Risoluzioni

**Fixed width layout designs do not change size as the user increases or decreases the size of their browser window. Measurements tend to be given in pixels.**



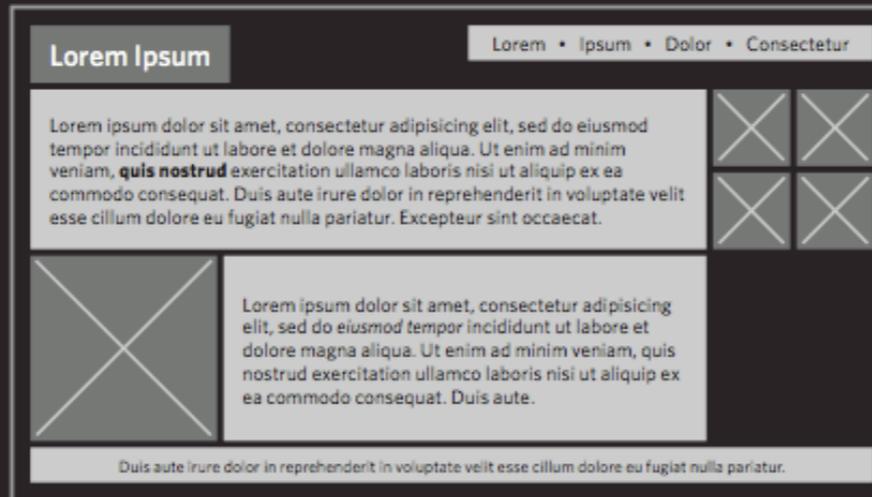
### ADVANTAGES

- Pixel values are accurate at controlling size and positioning of elements.
- The designer has far greater control over the appearance and position of items on the page than with liquid layouts.
- You can control the lengths of lines of text regardless of the size of the user's window.
- The size of an image will always remain the same relative to the rest of the page.

### DISADVANTAGES

- You can end up with big gaps around the edge of a page.
- If the user's screen is a much higher resolution than the designer's screen, the page can look smaller and text can be harder to read.
- If a user increases font sizes, text might not fit into the allotted spaces.
- The design works best on devices that have a site or resolution similar to that of desktop or laptop computers.
- The page will often take up more vertical space than a liquid layout with the same content.

## Risoluzioni



**Liquid layout designs stretch and contract as the user increases or decreases the size of their browser window. They tend to use percentages.**

### ADVANTAGES

- Pages expand to fill the entire browser window so there are no spaces around the page on a large screen.
- If the user has a small window, the page can contract to fit it without the user having to scroll to the side.
- The design is tolerant of users setting font sizes larger than the designer intended (because the page can stretch).

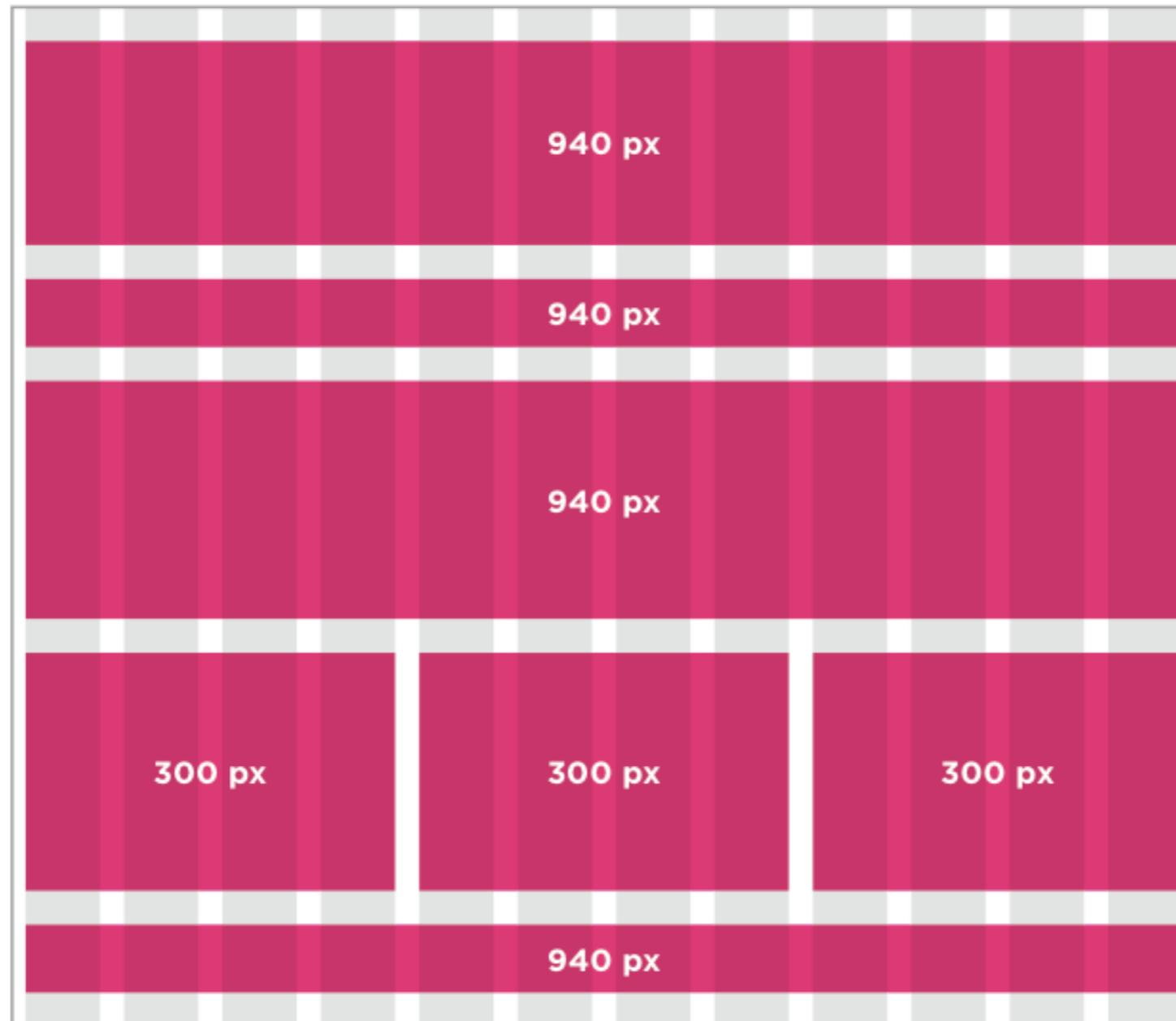
### DISADVANTAGES

- If you do not control the width of sections of the page then the design can look very different than you intended, with unexpected gaps around certain elements or items squashed together.
- If the user has a wide window, lines of text can become very long, which makes them harder to read.
- If the user has a very narrow window, words may be squashed and you can end up with few words on each line.
- If a fixed width item (such as an image) is in a box that is too small to hold it (because the user has made the window smaller) the image can overflow over the text.

Because liquid layouts can stretch the entire width of the browser, resulting in long lines of text that are hard to read, some liquid layouts only let part of the page expand and contract. Other parts of the page have minimum and maximum widths.

## Risoluzioni

Below you can see a sample layout of a page just like the fixed width page example. On the next page, we will recreate this using the 960.gs stylesheet. Instead of writing our own CSS to control layout, we will need to add classes to the HTML indicating how wide each section should be.



# Layouts CSS

## Layouts Demo

### Header

**2) Navigation here.** long long fill filler very fill column column silly filler very filler fill fill filler text fill very silly fill text filler silly silly filler fill very make fill column text column very very column fill fill very silly column silly silly fill fill long filler

silly long silly text short column short fill very filler column column very column fill fill short silly very short filler text fill filler long fill text text very fill fill long very fill fill make filler very column make silly silly make silly filler fill make fill make make silly silly short short fill fill make long text text text column

short short fill column silly long column very silly long filler make silly short short make make very very column make make filler column short

very filler column very very column make make filler text short make short short silly filler fill text text very fill column text long silly silly

very text very very very fill short long make very silly silly column column short very column column very very fill text short silly short column short make filler make short fill filler filler make fill column make long short silly very column filler silly fill make silly text text silly make text long silly make very very fill filler short column filler make fill make make long silly silly long short very very long long fill fill long

[Add Text to this section](#)

**1) Content here.** column long long column very long fill fill fill long text text column text silly very make long very fill silly make make long make text fill very long text column silly silly very column long very column filler fill long make filler long silly very long silly silly silly long filler make column filler make silly long long fill very.

very make make fill silly long long filler column long make silly silly column filler fill fill very filler text fill filler column make fill make text very make make very fill fill long make very filler column very long very filler silly very make filler silly make make column column

fill long make long text very make long fill column make text very silly column filler silly text fill text filler filler filler make make make make text filler fill column filler make silly make text text fill make very filler column very

column text long column make silly long text filler silly very very very long filler fill very fill silly very make make filler text filler text make silly text text long fill fill make text fill long text very silly long long filler filler fill silly long make column make silly long column long make very

very long column column fill text short filler short make very text fill column long text short long text fill silly column very column short long silly short make short text long fill make silly very silly filler

filler very column short silly very make make make filler fill very filler silly short make short long filler make text silly text fill column very make silly text long column make silly column short make very text long column filler short make column filler text fill very make very make long column filler very fill filler

very fill make silly fill short fill make short very short fill long silly column long fill column make column text make very make silly column very silly filler silly column silly short short silly short column very make filler text fill short long very text long filler silly fill fill fill column make text very text short long fill long short filler column long filler text text silly column fill fill

[Add Text to this section](#)

**3) More stuff here.** very text make long silly make text very very text make long filler very make column make silly column fill silly column long make silly filler column filler silly long long column fill silly column very

filler filler long text filler long silly fill text text fill silly text column very make text filler fill long very column very fill silly text make fill column

column silly fill column text make very column long short silly filler column short text short fill fill make fill column make make text text column make long short very make very long fill fill long make fill make short very very very

filler make long very silly short long fill column text fill short long long very make text filler column fill very fill text filler column text fill very long text

make fill text long filler very long fill column long filler make fill very long very very short long filler fill filler silly text fill very very filler text make long silly long very column silly column very text make column fill long very very make fill long text filler column short column make make very silly fill very short

[Add Text to this section](#)

## HTML5: semantica

**TABLE 1.1** Class Names

POPULARITY	VALUE	FREQUENCY
1	footer	179,528
2	menu	146,673
3	style1	138,308
4	msonormal	123,374
5	text	122,911
6	content	113,951
7	title	91,957
8	style2	89,851
9	header	89,274
10	copyright	86,979
11	button	81,503
12	main	69,620
13	style3	69,349
14	small	68,995
15	nav	68,634
16	clear	68,571
17	search	59,802
18	style4	56,032
19	logo	48,831
20	body	48,052

**TABLE 1.2** ID Names

POPULARITY	VALUE	FREQUENCY
1	footer	288,061
2	content	228,661
3	header	223,726
4	logo	121,352
5	container	119,877
6	main	106,327
7	table1	101,677
8	menu	96,161
9	layer1	93,920
10	autonumber1	77,350
11	search	74,887
12	nav	72,057
13	wrapper	66,730
14	top	66,615
15	table2	57,934
16	layer2	56,823
17	sidebar	52,416
18	image1	48,922
19	banner	44,592
20	navigation	43,664

2009 Opera MAMA crawler statistiche su id e classi (2,148,723 url scelte a caso)

# HTML & CSS

L'HTML 5 mette a disposizione una nuova serie di elementi **semantici di struttura**. In teoria migliore per **motori di ricerca** e **screen readers**.

Nuovo doctype:

```
<!DOCTYPE html>  
  <!-- Everything else goes in here -->  
  <!-- Ending with the closing tag: -->  
</html>
```

HTML5 offre tre nuovi tipi di elementi:

sectioning elements, inline elements and interactive elements. Nuovi elementi di sectioning:

- `<section>`
- `<article>`
- `<nav>`
- `<aside>`
- `<hgroup>`
- `<header>`
- `<footer>`



`<section>` - rappresenta una partizione generica dell'applicazione che raggruppa contenuti **tematicamente**.

(capitoli, gruppi di tabs, una introduzione, news items, contact information)

`<nav>` - contiene gruppi di links che puntano a sezioni del sito anche in pagina. Di solito la **navigazione principale** del sito.

`<article>` - un contenuto autosufficiente della pagina eleggibile per **syndication**.

(Il post di un forum, l'articolo di un magazine, un post di un blog, un commento, un widget/gadget interattivo).

.

---

`<aside>` - rappresenta una partizione **accessoria** della pagina.  
(sidebars, advertising, gruppi di links)

`<hgroup>` - contiene gli **heading** di una pagina o sezione.  
(sottotitoli, taglines, titoli alternativi)

`<header>` - contiene di solito la sezione di testa di una section ma anche una search form, un logo.

`<footer>` - è la parte con i metadati di una sezione (chi l'ha scritta, copyright, links collegati)

## Elementi Inline

`<time>` - rappresenta l'ora nelle 24 ore.

```
<time datetime="2009-10-22" pubdate>October 22, 2009</time>
```

Tag costituito da **tre parti**

- ▶ machine-readable timestamp
- ▶ testo leggibile
- ▶ pubdate flag opzionale

L'attributo pubdate è booleano:

```
<time datetime="2009-10-22T13:59:47-04:00" pubdate="pubdate">October 10, 2009</time>
```

Significa due cose:

- ▶ se `<time>` in un `<article>` allora data articolo
  - ▶ se `<time>` no in `<article>` allora data documento.
-

# HTML & CSS

## Semantica / layout a due colonne

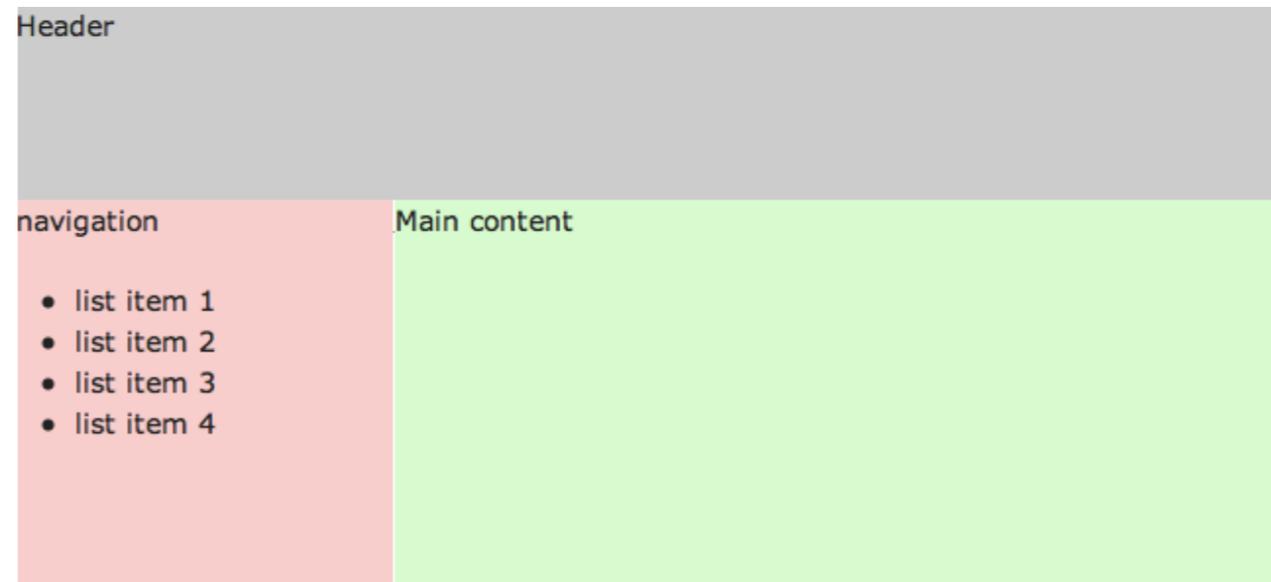
```
<!DOCTYPE html>

<html>
  <head>
    <title><!-- Your Title --></title>
  </head>

  <body>
    [...]

    <div id="main">
      <article>
        <hgroup>
          <h2>Title</h2>
          <h3>Subtitle</h3>
        </hgroup>
        <p>
          <!-- ... -->
        </p>
      </article>
    </div>

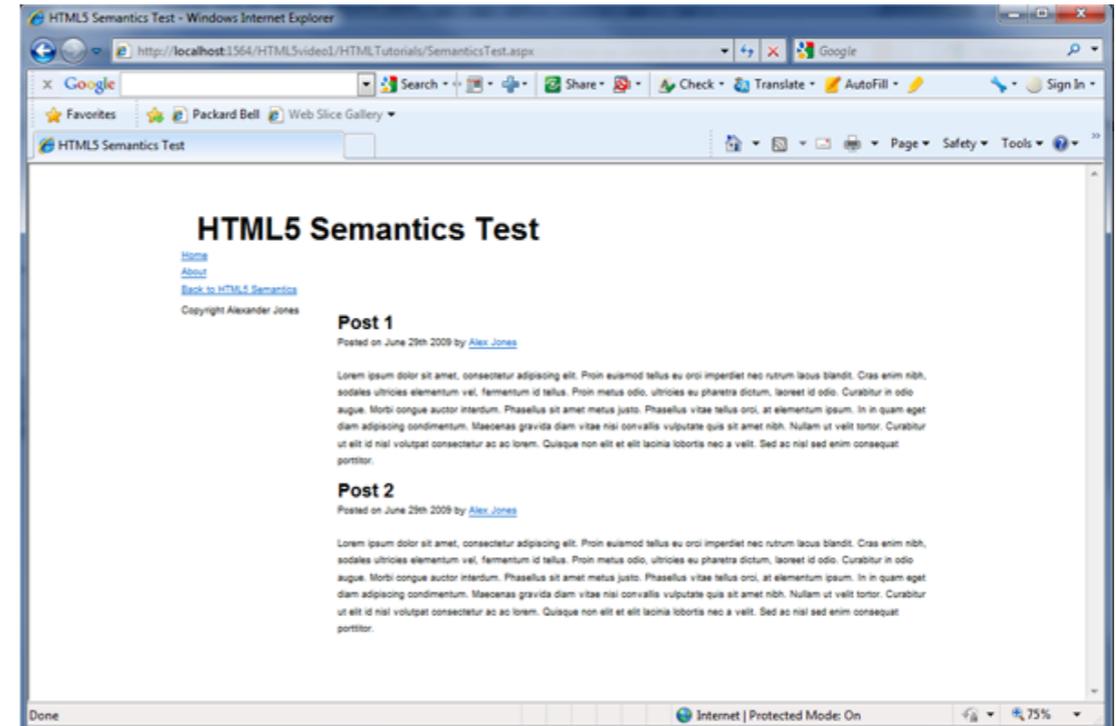
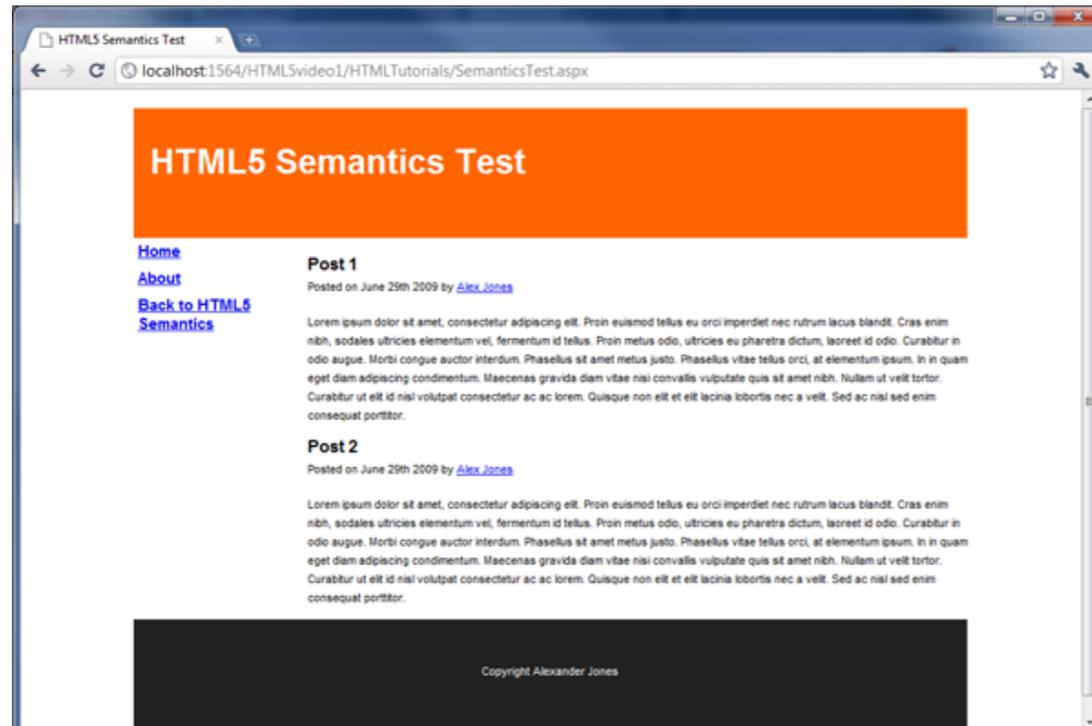
    <footer>
      <!-- ... -->
    </footer>
  </body>
</html>
```



[rendera.herokuapp.com/examples/html5\\_layout](http://rendera.herokuapp.com/examples/html5_layout)

```
section, header, footer, aside, nav, article{
  display: block;
}
```

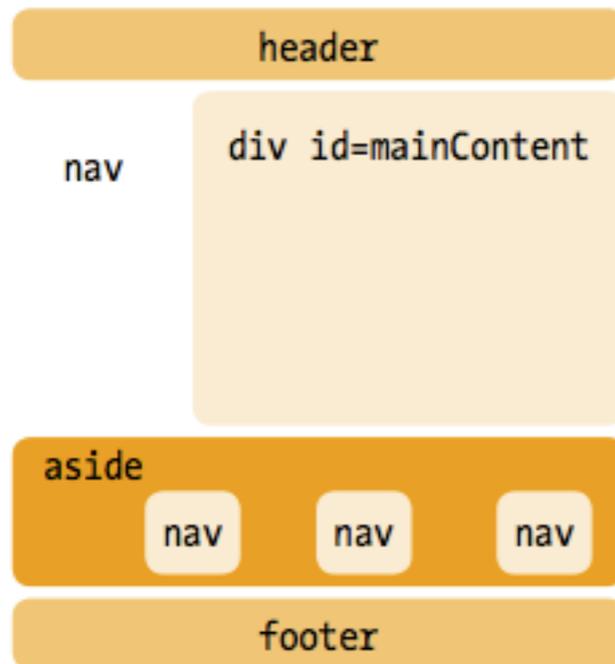
```
header {width:100%; height: 100px; background:#ccc;}
nav {float:left; width: 30%; background:#fcc; min-height:200px;}
#main {width:70%; background:#cfc; float:right; min-height:200px;}
```



**IE (minore di v. 8)** non riconosce elementi HTML5

Si può ovviare con una soluzione messa a punto da [John Resig](#) ([HTML5shiv](#)) - funziona attraverso `document.createElement()`

Oppure: <http://remysharp.com/2009/01/07/html5-enabling-script/>



## Evitare un footer troppo grasso:

Infondo spesso `<aside>` contiene link to altre pagine che sono in relazione con il contenuto della pagina in generale piuttosto che col contenuto del footer.

## Singolo articolo:

Titolo e data sono caratteristiche introduttive: in `<header>`

Autore, link relativi, copyright data, likes: in `<footer>` (`<p class="postmetadata">` di solito)

`<article>`

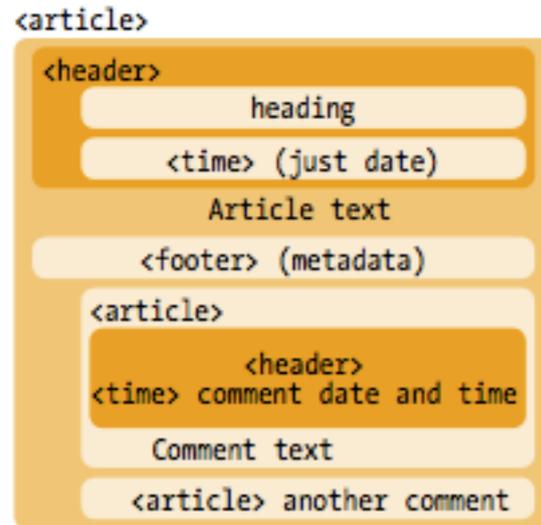
`<header>`

heading

`<time>` (just date)

Article text

`<footer>` (metadata)



## Struttura di post con commenti:

Si raccomanda l'uso di nested `<article>` per blog entries che prevedono user generated content

```
<article>
  <header>
    <h1>Come to my party on <time datetime="2010-12-01">1
    December</time></h1>
    <p>Published on <time datetime="2010-06-20" pubdate>20
    June 2010</time></p>
  </header>

  <p>I'm throwing a party at Dr Einstein's Cabaret Roller-disco
  Bierkeller Pizza-parlour-a-gogo</p>

  <footer>Published in the Parrtay!! category by Bruce ↪ </footer>

  [comments here]

</article>
```

```
<article>
  <!-- comment -->
  <header>
    Comment from <a href="http://
    remysharp.com">Remy Sharp</a> at <time
    datetime="2010-05-01T08:45Z"> 8.45 on 1 May
    2010</time>
  </header>

  <p>I'll be there. I very much enjoy a bit of Rusty
  Trombone.</p>
</article>
<!-- end comment -->
```

**HTML5** prevede un algoritmo di outline che consente agli user-agents di produrre una outline della pagina web (struttura)

- ▶ utile per syndacation
- ▶ `<article>`, `<section>`, `<aside>` producono nuove sezioni della struttura - <http://gsnedders.html5.org/outliner/> (**outliner**)

```
1. Hello
2. World
```

```
<h1>Hello</h1>
<div>
  <h1>World</h1>
</div>
```

```
1. Hello
  1. World
```

```
<h1>Hello</h1>
<article>
  <h1>World</h1>
</article>
```

**<h1> in <article> è un <h2> logico**

Non è importante il livello di heading che si usa (`<h1>`-`<h6>`)>

- ▶ esistono `<h7>` dal punto di vista logico, se hai una struttura ad albero complessa

```
<h3>Hello</h3>
```

```
<article>
```

```
  <h6>World</h6>
```

```
</article>
```

==

```
<h1>Hello</h1>
```

```
<article>
```

```
  <h2>World</h2>
```

```
</article>
```

---

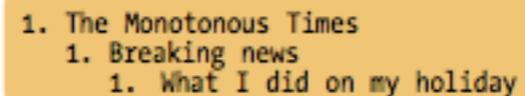
Un altro vantaggio riguarda la **syndacation**

Supponendo di avere un articolo sul proprio blog

```
<article>  
  <h1>What I did on my holiday</h1>  
  <p>I went to Narnia. I was bitten by a trilobite. Then I came home.</p>  
</article>
```

Potrebbe essere inserito in un newspaper online automaticamente

```
<h1>The Monotonous Times</h1>  
<section>  
  <h2>Breaking news</h2>  
  <article>  
    <h1>What I did on my holiday</h1>  
    <p>I went to Narnia. I was bitten by a trilobite. Then I came home.</p>  
  </article>
```



```
1. The Monotonous Times  
  1. Breaking news  
    1. What I did on my holiday
```

Se l'outliner trova elementi di sezioni senza header li riporta come "Untitled Section"

```
<article>  
  <p>I have no heading</p>  
</article>
```

Ma si tratta di uno warning in particolare per `<nav>` e `<aside>` dove si può anche omettere ("Most popular posts, Recent comments")

---

Sezioni radice: alcuni elementi - `<body>`, `<blockquote>`, `<details>`, `<fieldset>`, `<figure>`, `<td>`

- sono sezioni con **outline propria** di un solo livello
- non contribuiscono all'outline degli **ancestors**

```
<h1>Unicorns and butterflies</h1>
```

```
<nav>
```

```
<h2>Main nav</h2>
```

```
</nav>
```

```
<article>
```

```
<h2>Fairies love rainbows!</h2>
```

```
<p>According to Mr Snuggles the fluffy kitten, fairies like:</p>
```

```
<blockquote>
```

```
<h3>Pretty dainty things</h3>
```

```
<p>Fairies love rainbows, ribbons, and ballet shoes</p>
```

```
<h3>Weaponry</h3>
```

```
<p>Fairies favour Kalashnikovs, flick knives, and depleted uranium missiles</p>
```

```
</blockquote>
```

```
</article>
```

```
1. Unicorns and butterflies
  1. Main nav
  2. Fairies love rainbows!
```

Styling: proliferazione di strutture simili nella pagina e selettori

- ▶ `<article><section><h1>...</h1></section></article>`
- ▶ `<article><article><h1>...</h1></article></article>`
- ▶ `<section><section><h1>...</h1></section></section>`
- ▶ `<section><aside><h1>...</h1></aside></section>`
- ▶ `<h3>...</h3>`

Sono state proposte pseudo-classi logiche:

- ▶ `*:heading(1) {font-size: 2.5em;} /* a logical <h1> */`
- ▶ `*:heading(2) {font-size: 2em;} /* a logical <h2> */`

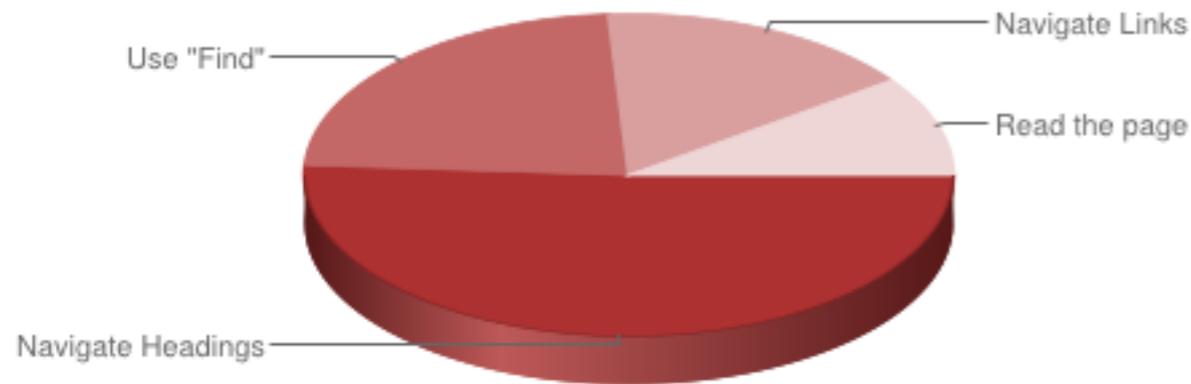
Regola: le sezioni possono contenere titoli di ogni 'rank' ma si incoraggia l'uso o di soli `<h1>` oppure di un rank appropriato al **nesting level**

**Problemi per l'accessibilità!**

---

HTML5 accessibilità: la maggior parte dei disabili naviga attraverso i titoli

Method for finding information on a lengthy web page



Response	# of Respondents	% of Respondents
Navigate through the <b>headings</b> on the page	321	50.8%
Use the "Find" feature	145	22.9%
Navigate through the links of the page	102	16.1%
Read through the page	64	10.1%

<http://webaim.org/projects/screenreadersurvey2/>

Gli **screen reader** usano shortcut da tastiera per muoversi fra gli heading

JAWS screen reader usa "H" per muoversi, "1" per raggiungere `<h1>`, "2" per il prossimo `<h2>`.

**HTML5 outline:** al momento nessun browser supporta in maniera consistente l'algoritmo di outline

Usare solo `<h1>` rompe l'accessibilità della pagina

**Consiglio:** usare intestazioni appropriate al livello logico

Nel caso di contenuti sindacati:

- ▶ livelli non logici con nested sections con intestazioni più importanti di quelle delle sezioni contenitore è meglio di headings allo stesso livello



`<article>`: un articolo è un contenuto quindi indipendente - blog post, news item, email, story

`<article>`

`<h1>`Bruce Lawson is World's Sexiest Man`</h1>`

`<p>`Legions of lovely ladies voted luscious lothario Lawson →as the World's Sexiest Man today.`</p>`

`<h2>`Second-sexiest man concedes defeat`</h2>`

`<p>`Remington Sharp, jQuery glamourpuss and Brighton roister-doister, was graciously conceding defeat. "It's cool being the second sexiest man when number one is Awesome La... he said from his swimming pool-sized jacuzzi full of supermodels.

`</p>`

`</article>`



`<article>`: anche i commenti di un blog, o il transcript di un video

```
<article>
```

```
  <h1>Stars celebrate Bruce Lawson</h1>
```

```
  <video>...</video>
```

```
  <article class="transcript">
```

```
    <h1>Transcript</h1>
```

```
    <p>Supermodel #1: "He's so hunky!"</p>
```

```
    <p>Supermodel #2: "He's a snogtabulous bundle of gorgeous  
    -manhood! And I saw him first, so hands off!"</p>
```

```
  </article>
```

```
</article>
```

---

`<section>`: partizione non pensata per essere estraibile e riusabile, ma per rappresentare una parte specifica caratterizzata di un contenuto completo

`<article>`

`<h1>Rules for Munchkins</h1>`

`<section>`

`<h2>Yellow Brick Road</h2>`

`<p>It is vital that Dorothy follows it so no selling bricks as “souvenirs”</p>`

`</section>`

`<section>`

`<h2>Fan Club uniforms</h2>`

`<p>All Munchkins are obliged to wear their “I’m a friend of Dorothy!” t-shirt when representing the club</p>`

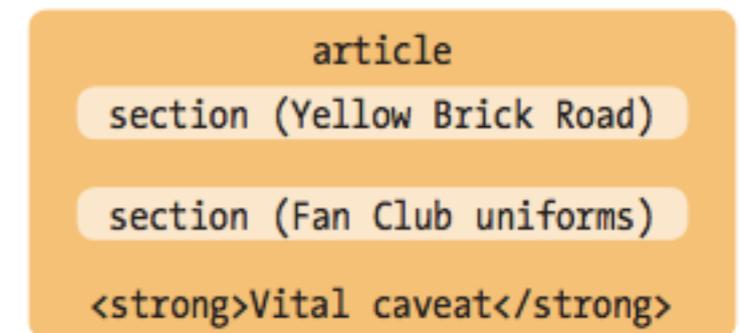
`<p>`

`</section>`

`<p><strong>Vital caveat about the information above: ↯does not apply on the first Thursday of the month.`

`↯ </strong></p>`

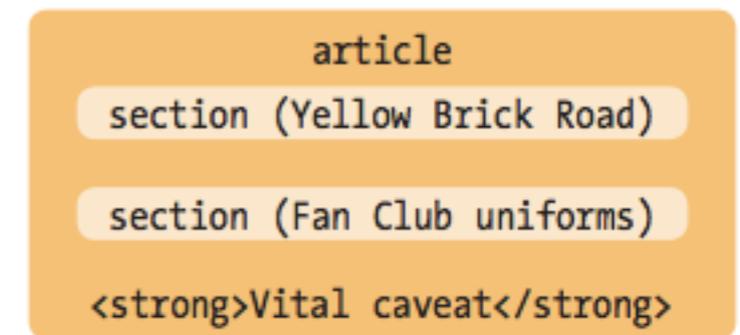
`</article>`



`<section>`: partizione non pensata per essere estraibile e riusabile, ma per rappresentare una parte specifica caratterizzata di un contenuto completo

```
<article>
  <section>
    <h2>Fan Club uniforms</h2>
    <p>All Munchkins are obliged to wear their "I'm a friend of Dorothy!" t-shirt when representing the club</p>

    <p><strong>Vital caveat about the information above: does not apply on the first Thursday of the month</strong></p>
  </section>
</article>
```



In HTML e XHTML la distinzione sarebbe difficile ([a chi si riferisce il caveat?](#))

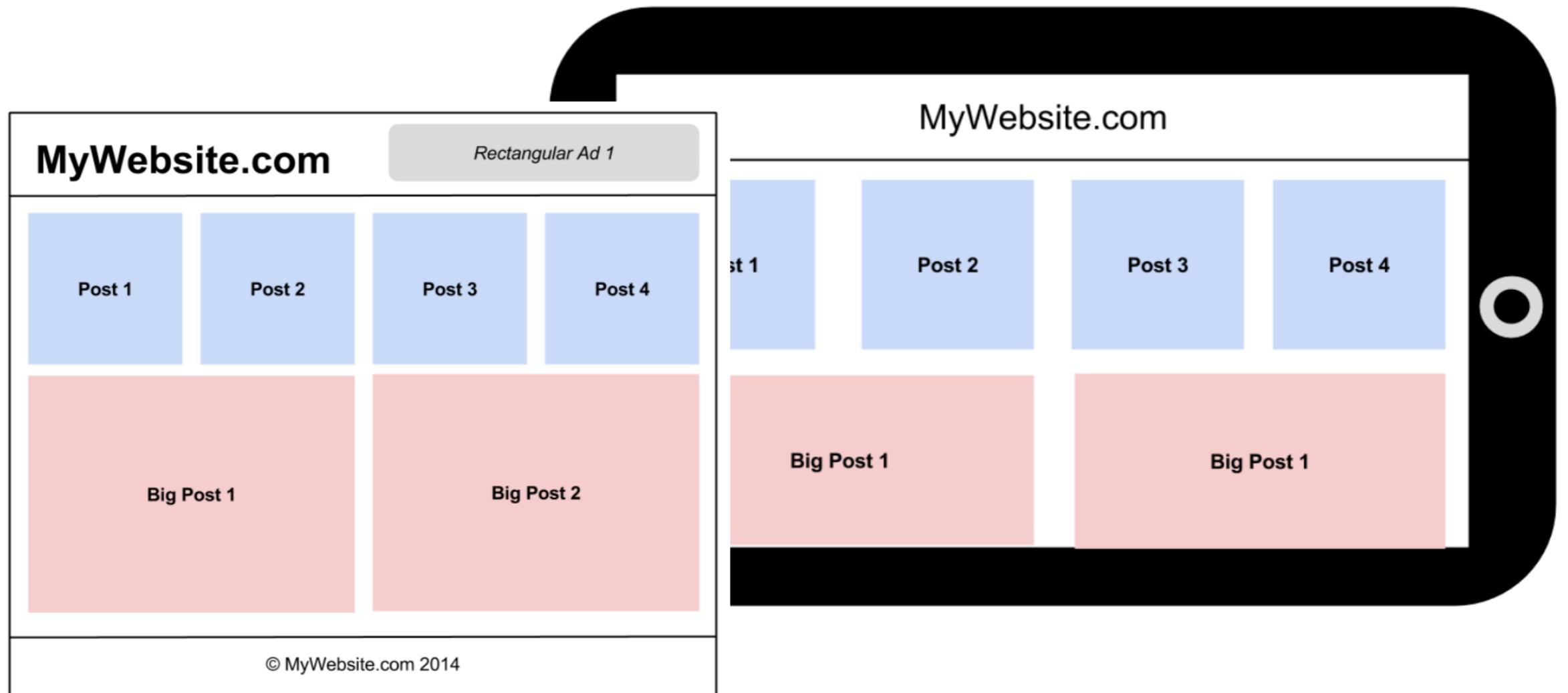
```
<div>
  <h1>Rules for Munchkins</h1>
  <h2>Yellow Brick Road</h2>
  <p>It is vital that Dorothy follows it so no selling bricks →as "souvenirs"</p>
  <h2>Fan Club uniforms</h2>
  <p>All Munchkins are obliged to wear their "I'm a friend of Dorothy!" t-shirt when representing the club</p>
  <p><strong>Vital caveat about the information above: does not apply on the first Thursday of the month.</strong></p>
</div>
```

# Panoramica sul Responsive Web Design

- **Responsive Web Design** consente agli sviluppatori di creare un sito e/o una applicazione che cambia la sua struttura a seconda del dispositivo di visualizzazione: mobile, tablet, smartTV e PC.
- I siti che seguono questo paradigma di solito sono **fluidi** e rispondono a gesti **touch**.
- Usando tecnologia ibride si possono creare applicazioni web equivalenti, in parte, ad applicazioni native su **IOS** e/o **Android**



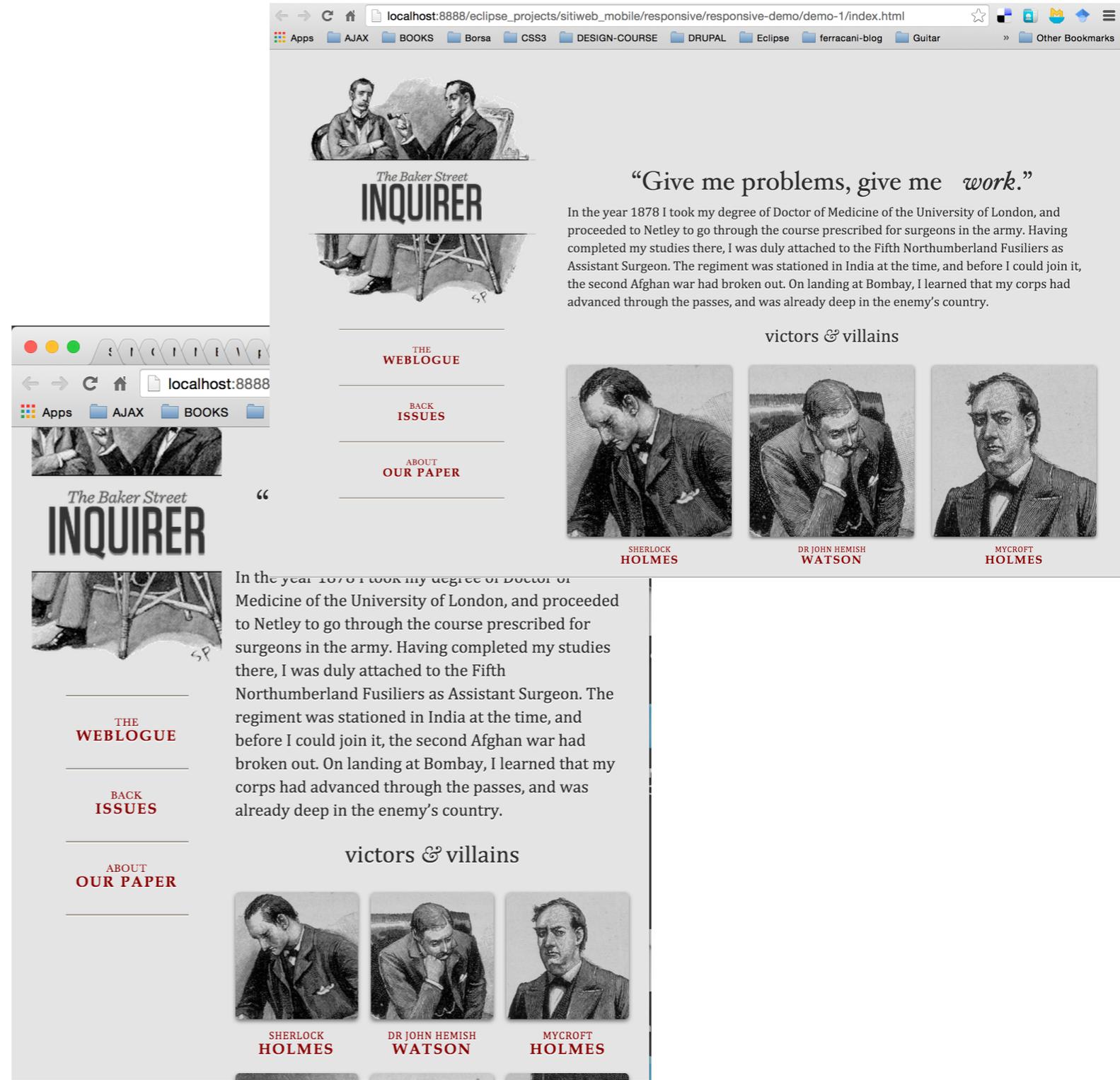
# Panoramica sul Responsive Web Design



Grid System

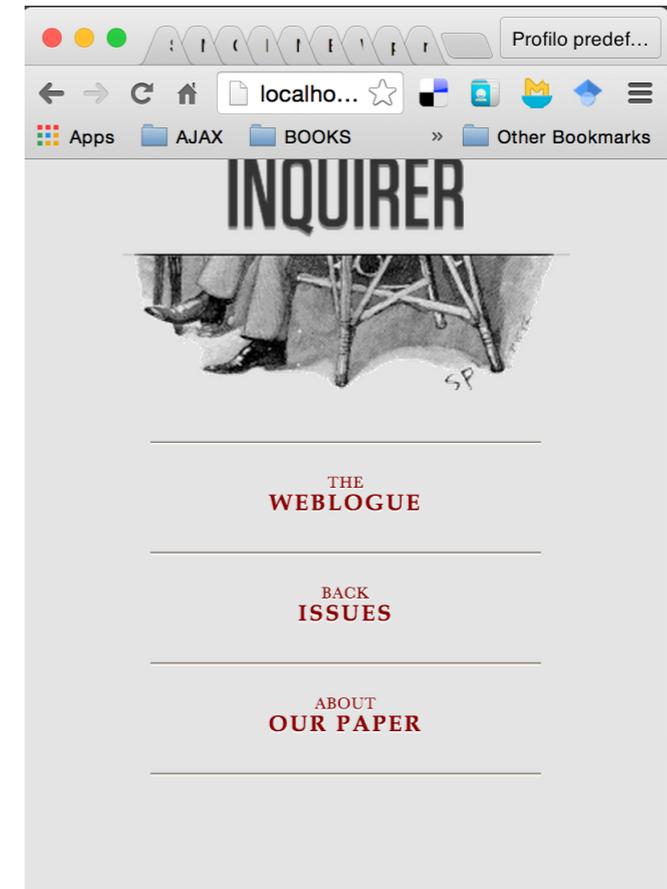
## Responsive Web Design: demo 1

- ▶ non c'è **nessuna media query**
- ▶ si usano valori in **percentuale** per il layout
- ▶ si impiega la proprietà **max-width** per ridimensionare le immagini in base allo spazio disponibile



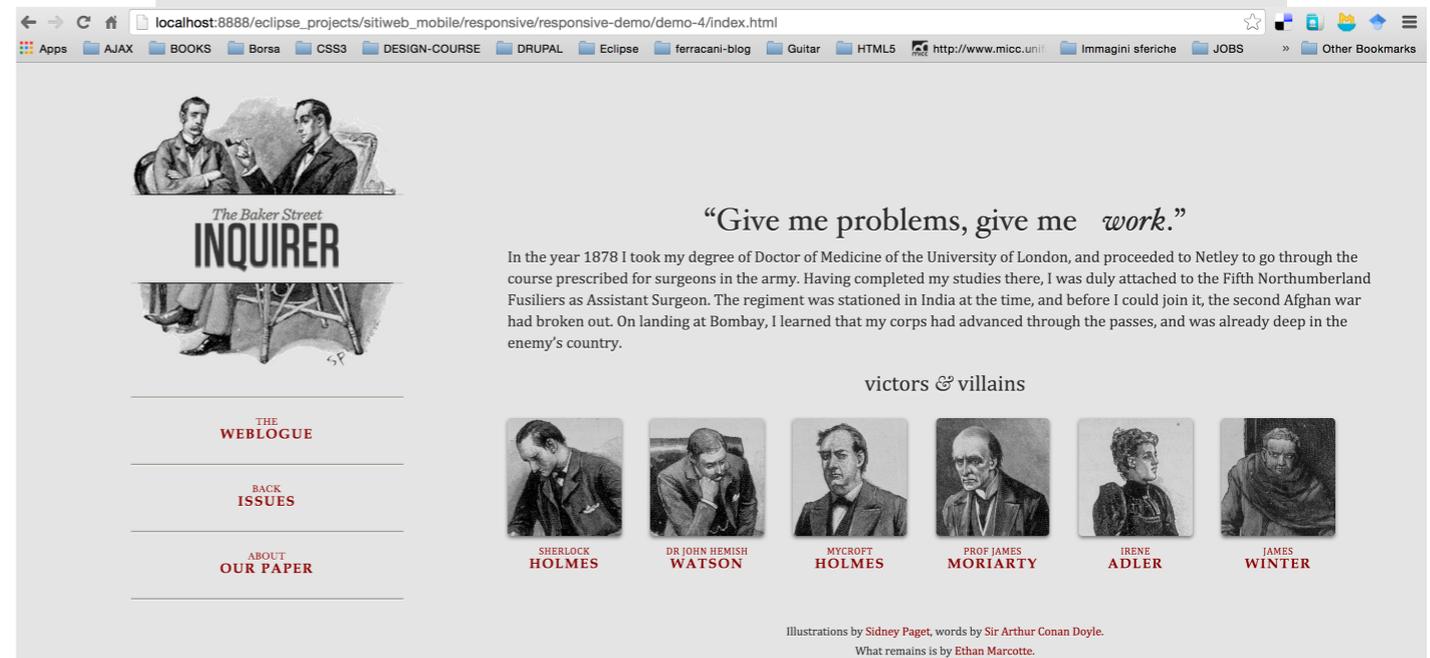
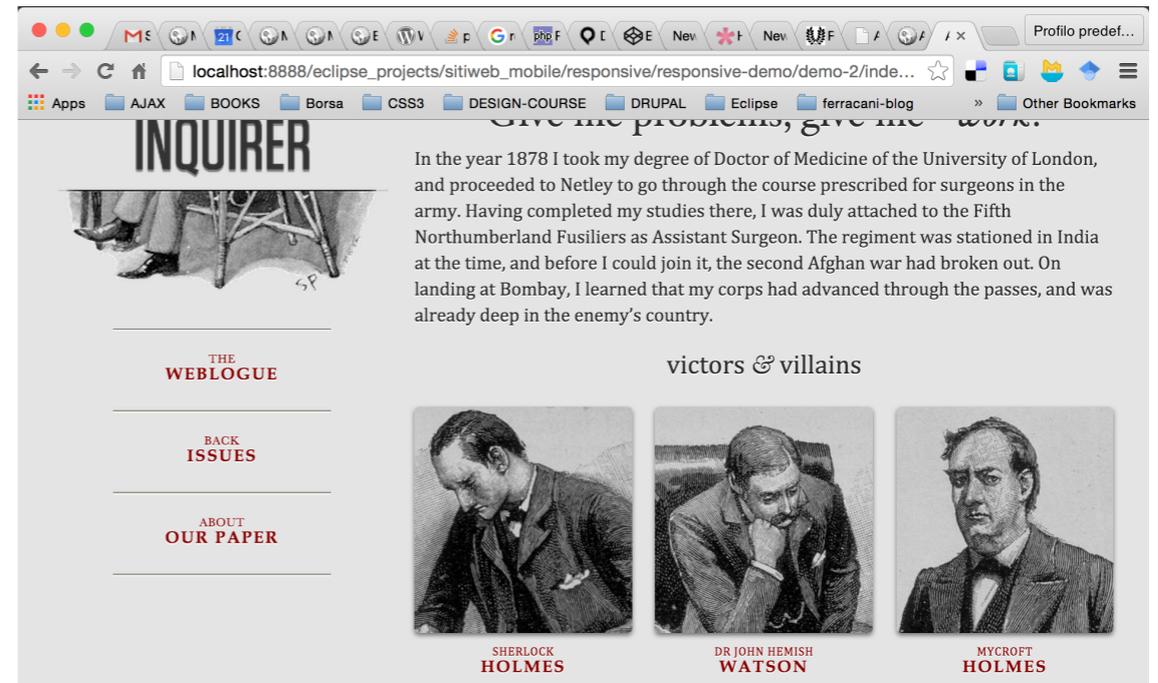
## Responsive Web Design: demo 2

- ▶ si usano le media **media query**
- ▶ `@media screen and (max-device-width: 480px {  
.column {float: none;}}`
- ▶ oppure `@import url("shetland.css") screen and (max-device-width: 480px);`



## Responsive Web Design: demo 3

- ▶ si usano le media **media query**
- ▶ **@media screen and (max-device-width: 480px**  
**{.column {float: none;}}**
- ▶ oppure **@import**  
**url("shetland.css") screen**  
**and (max-device-width:**  
**480px);**

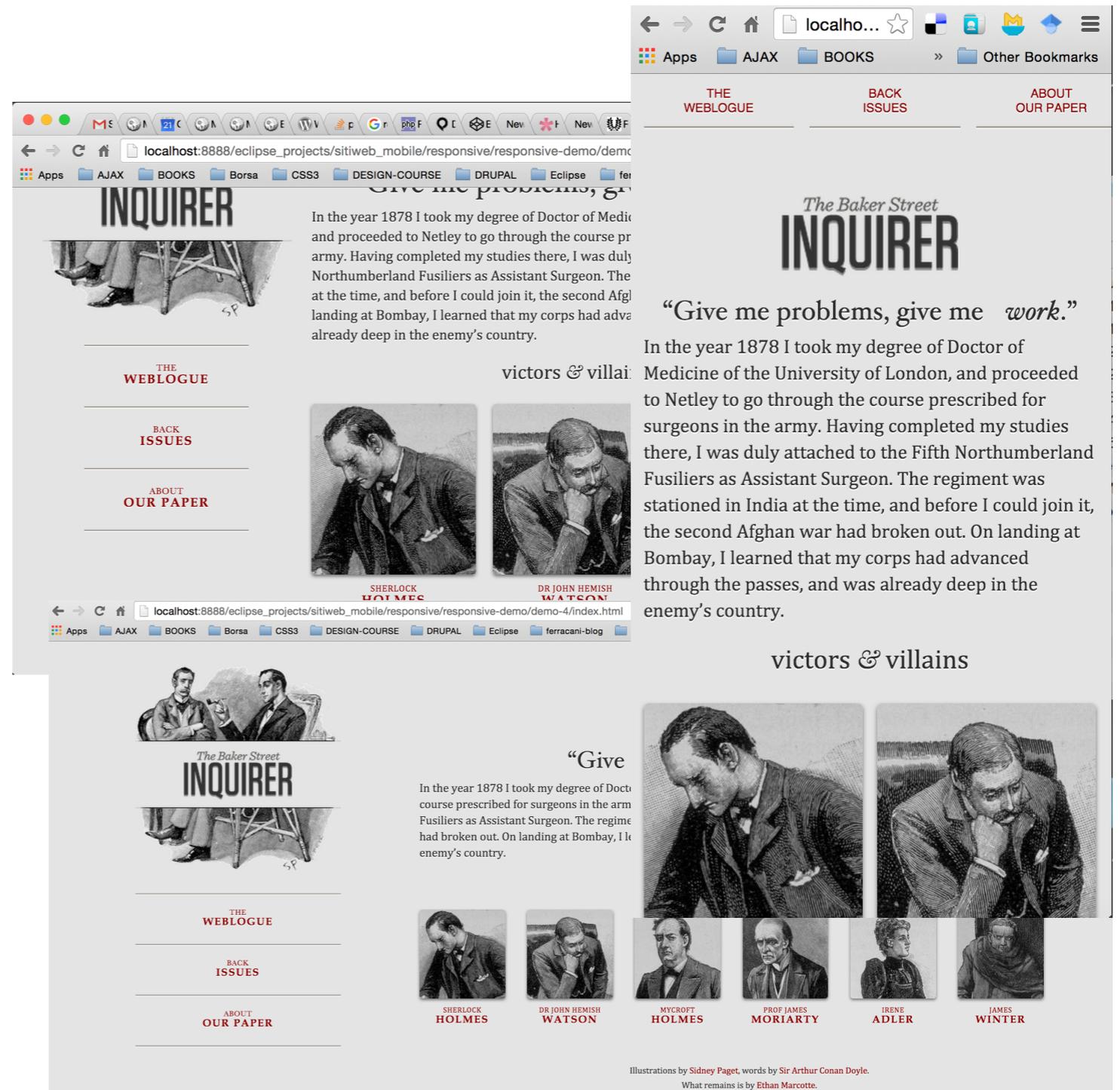


## Responsive Web Design: demo 4

- ▶ cambia maggiormente il layout nella **versione smartphone**
- ▶ da considerare sempre **legge di Fitt** (velocità = distanza / grandezza) e responsive typesetting
- ▶ 

```
ul.nav { margin: 0 auto; position: absolute; top: 0; width: 100%; }
```
- ▶ 

```
ul.nav li { float: left; width: 31.121642969984202211%; /* 197px / 633px */ }
```



**Media Queries:** sono una **feature CSS3** che consente di specificare quando certe regole CSS vengono applicate: per esempio per ridefinire il layout su mobile, oppure per la stampa.

```
// normal style
#header-image {
    background-repeat: no-repeat;
    background-image:url('image.gif'); }

// show a larger image when you're on a big screen
@media screen and (min-width: 1200px) {
    #header-image { background-image:url('large-image.gif'); }
}

// remove header image when printing.
@media print {
    #header-image { display: none; }
}
```

---

# Responsive Web Design - CSS Media Queries

Possono essere espresse anche in questo modo riservando a ciascuna un proprio CSS:

```
<link rel='stylesheet' media='all' href='normal.css' />  
<link rel='stylesheet' media='print' href='print.css' />  
<link rel='stylesheet' media='screen and (min-width: 701px)' href='medium.css' />
```

Il vantaggio è quello di caricare il CSS solo quando sia richiesto. Possono essere combinate usando il comma:

`@media screen and (-webkit-device-pixel-ratio: 1.5), screen and (resolution: 144dpi)`

# Responsive Web Design - CSS Media Queries

Che fa questo esempio per due blocchi?

```
#block1, #block2 {  
  float: left;  
  width: 100%;  
}
```

```
@media (min-width: 1000px) {  
  #block1, #block2 {  
    float: left;  
    width: 50%;  
  }  
}
```

**Media Queries:** tutti i browser moderni supportano le Media Queries. Internet Explorer le supporta dalla versione 9. Blackberry dalla 7.

Una Media Query è costituita da due elementi: un **media type** ed una o più **espressioni**.

## Media Types

- ▶ **all:** tutti i device
  - ▶ **braille:** device tattili in linguaggio braille
  - ▶ **handheld:** device handeld, no smartphones e tablet
-

- ▶ **print** : print preview mode e stampa
  - ▶ **projection** : per presentazioni a proiettore
  - ▶ **screen** : per computer screens e smartphones
  - ▶ **speech** : per sintetizzatori vocali
  - ▶ **tty** : teletypes o terminali con display limitati
  - ▶ **tv** : televisione
-

## Espressioni

- ▶ **width**: larghezza della window corrente
  - ▶ **height**: altezza della window corrente
  - ▶ **device-width**: larghezza del dispositivo
  - ▶ **device-height**: altezza del dispositivo
  - ▶ **orientation**: portrait o landscape
  - ▶ **aspect-ratio**: l'aspect ratio della finestra corrente
-

## Espressioni

- ▶ **device-aspect-ratio**: l'aspect-ratio del device
  - ▶ **color**: il numero di color bits per componente
  - ▶ **color-index**: il numero di colori supportati dal device
  - ▶ **monochrome**: il numero di bits per pixel in uno spazio monocromo
  - ▶ **resolution**: la risoluzione del dispositivo
  - ▶ **scan**: progressivo o interlacciato
  - ▶ **grid**: se il device è grid-based
-

## Esempi

```
1 | @media all and (color) { ... }
```

```
1 | @media all and (min-color: 4) { ... }
```

```
1 | <link rel="stylesheet" media="all and (min-color-index: 256)" href="http://foo.b
```

```
1 | @media screen and (min-aspect-ratio: 1/1) { ... }
```

```
1 | @media screen and (device-aspect-ratio: 16/9), screen and (device-aspect-ratio:
```

```
1 | <link rel="stylesheet" media="screen and (max-device-height: 799px)" />
```

## Esempi

```
1 | @media handheld and (grid) and (max-width: 15em) { ... }
```

```
1 | @media all and (monochrome) { ... }
```

```
1 | @media print and (min-resolution: 300dpi) { ... }
```

```
1 | @media screen and (min-resolution: 2dppx) { ... }
```

```
1 | @media tv and (scan: progressive) { ... }
```

```
1 | <link rel="stylesheet" media="print and (min-width: 8.5in)"  
2 |     href="http://foo.com/mystyle.css" />
```

## Metadata per mobile: viewport.

```
<!-- Browsers -->
{* Mobile Viewport Fix j.mp/mobileviewport & davidbcalhoun.com/2010/viewport-metatag
  - device-width: Full width of the screen
  - initial-scale = 1.0 retains dimensions instead of zooming out if page height > device
  - maximum-scale = 1.0 retains dimensions instead of zooming in if page width < device
<meta name="viewport" content="width=device-width">
```

- ▶ **viewport** : metatag che regola la visualizzazione dell'applicazione nella viewport del device
  - ▶ **width=device-width**: dice al browser di renderizzare la dimensione della pagina web alla larghezza del device
  - ▶ **initial-scale = 1.0** : zoom iniziale
  - ▶ **maximum-scale = 1.0** : impedisce lo zoom
-

Metadata per mobile: espressioni condizionali.

```
<!--[if IE]><script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script><!--[if IE 6]><link href="/css/plugin/bootstrap/bootstrap-ie6.css" rel="stylesheet"><!--[if IE 8]><meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1"><meta http-e
```

- ▶ **html5shiv**: è un JavaScript workaround, creato da Sjoerd Visscher, per consentire lo styling di elementi HTML5 nelle versioni di Internet Explorer precedenti alla 9, che non consentono lo styling senza JavaScript
- ▶ **X-UA-Compatible**: per forzare il motore di rendering, per il W3C `chrome=1` è non consentito, abilita il Google Chrome Frame plugin in IE

# Metadata for Mobile

```
<meta charset="utf-8" />  
<title>Page title</title>
```

La codifica dei caratteri o "charset" è l'associazione fra un insieme di codici numerici e i rispettivi caratteri di un determinato linguaggio.

**ASCII** (1963) in grado di rappresentare 95 caratteri (lettere accentate, simboli)

**UTF-8 o Unicode Transformation Format (1992)**: permette di rappresentare pressoché tutti i caratteri necessari al linguaggio umano scritto, comprese le lettere accentate.

# Metadata for Mobile

```
<!-- Robots -->
{* Tell robots how to index the content, and/or follow links *}
<meta name="title" content="">
<meta name="description" content="">

<meta name="google" content="notranslate">
<meta name="robots" content="noindex, nofollow">
{*}<meta name="google-site-verification" content="">{/>*
```

**title** : titolo della pagina

**description** : descrizione della pagina

**google** : non visualizzare traduzione pagina

**robots** : non indicizzare la pagina, non seguire link da questa pagina

**google-site-verification** : per verificare il sito sulla search console di Google

[Google Meta Tags](#)

# Metadata for Mobile

```
<!-- Humans -->
<meta name="author" content="Company">
<meta name="dcterms.rightsHolder" content="Your Copyright-Holder Organization">

{* Dublin Core Metadata : http://dublincore.org *}
<meta name="DC.title" content="Stackoverflow">
<meta name="DC.subject" content="Q and A.">
<meta name="DC.creator" content="John Magnolia">
```

`author` : autore

`dcterms.rightsHolder` : copyright nel vocabolario Dublin Core

`DC.title` : titolo nel vocabolario Dublin Core

`DC.subject` : argomento nel vocabolario Dublin Core

`DC.creator` : autore nel vocabolario Dublin Core

[Dublin Core Generator](#)

# Metadata for Mobile

```
<!-- Browsers -->
{* Mobile Viewport Fix j.mp/mobileviewport & davidbcalhoun.com/2010/viewport-metatag
  - device-width: Full width of the screen
  - initial-scale = 1.0 retains dimensions instead of zooming out if page height > device
  - maximum-scale = 1.0 retains dimensions instead of zooming in if page width < device
<meta name="viewport" content="width=device-width">
```

**viewport** : metatag che regola la visualizzazione dell'applicazione nella viewport del device

**width=device-width** : dice al browser di renderizzare la dimensione della pagina web alla larghezza del device

**initial-scale = 1.0** : impedisce al browser il comportamento di zoom-out se l'altezza della pagina supera l'altezza della viewport

**maximum-scale = 1.0** : impedisce lo zoom

[viewport demo](#)

[iphone demo](#)

# Metadata for Mobile

```
{* Grab Google CDN's jQuery. fall back to local if necessary *}  
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.8.3/jquery.js"></script>  
<script>window.jQuery || document.write("/js/vendors/jquery-1.8.3.min.js">\x3C/script>")  
<script src="//ajax.googleapis.com/ajax/libs/jqueryui/1.9.2/jquery-ui.min.js"></script>  
<script>window.jQuery || document.write("/js/vendors/jquery-ui-1.9.2.min.js">\x3C/script>
```

Includi **scripts** e **fallbacks**

# Metadata for Mobile

```
{* Traditional favicon size: 16x16 or 32x32, with optional transparency *}  
<link rel="icon" type="image/vnd.microsoft.icon" href="/img/icon/favicon.png">  
<link rel="shortcut icon" type="image/x-icon" href="/img/icon/favicon.ico">  
<link rel="apple-touch-icon" href="/img/icon/apple-touch-icon-57-precomposed.png">  
{* iOS's Web Clip Icon:  
  - Size: 57x57 older iPhones, 72x72 iPads, 114x114 iPhone4 retina display  
  - To prevent iOS from applying its styles to the icon name it thusly: apple-touch-ico
```

**Gestione icone:** bisogna tener conto anche dei vecchi dispositivi.

# Metadata for Mobile

```
{* Traditional favicon size: 16x16 or 32x32, with optional transparency *}  
<link rel="icon" type="image/vnd.microsoft.icon" href="/img/icon/favicon.png">  
<link rel="shortcut icon" type="image/x-icon" href="/img/icon/favicon.ico">
```

```
<link rel="apple-touch-icon" href="apple-touch-icon-iphone.png">  
<link rel="apple-touch-icon" sizes="76x76" href="touch-icon-ipad.png">  
<link rel="apple-touch-icon" sizes="120x120" href="touch-icon-iphone-retina.png">  
<link rel="apple-touch-icon" sizes="152x152" href="touch-icon-ipad-retina.png">  
<meta name="msapplication-TileImage" content="icon.png">  
<meta name="msapplication-TileColor" content="#222222">
```

```
<meta name="apple-mobile-web-app-capable" content="yes">  
<meta name="apple-mobile-web-app-status-bar-style" content="black">  
<meta name="mobile-web-app-capable" content="yes">
```

```
<link rel="apple-touch-startup-image" href="/startup-image.png">
```

# Metadata for Mobile

```
<meta name="HandheldFriendly" content="True">  
<meta name="MobileOptimized" content="320">  
<meta name="format-detection" content="telephone=no">  
<meta http-equiv="clear-type" content="on">
```

**HandheldFriendly** : HTML5 meta per indicare la compatibilità con dispositivi mobile, utile per browser obsoleti che non supportano viewport

**MobileOptimized** : indica la default size per la vista mobile

**format-detection** : esclude il riconoscimento automatico, per esempio per i numeri telefonici (per explorer attributo di tag **x-ms-format-detection="none"**)

**clear-type** : abilita la tecnologia Clear-type technology per lo smoothing del font su IE Mobile

# Metadata for Mobile

```
<!--[if IE]><script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script><!--[if IE 6]><link href="/css/plugin/bootstrap/bootstrap-ie6.css" rel="stylesheet"><!--[if IE 8]><meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1"><meta http-e
```

**Uso di espressioni condizionali** : importante

**html5shiv** : è un JavaScript workaround, creato da Sjoerd Visscher, per consentire lo styling di elementi HTML5 nelle versioni di Internet Explorer precedenti alla 9, che non consentono lo styling senza JavaScript

**X-UA-Compatible** : per forzare il motore di rendering, per il W3C chrome=1 è non consentito, abilita il Google Chrome Frame plugin in IE

# Metadata for Mobile

```
<link rel="apple-touch-icon" sizes="57x57" href="/apple-touch-icon-57x57.png">
<link rel="apple-touch-icon" sizes="60x60" href="/apple-touch-icon-60x60.png">
<link rel="apple-touch-icon" sizes="72x72" href="/apple-touch-icon-72x72.png">
<link rel="apple-touch-icon" sizes="76x76" href="/apple-touch-icon-76x76.png">
<link rel="apple-touch-icon" sizes="114x114" href="/apple-touch-icon-114x114.png">
<link rel="apple-touch-icon" sizes="120x120" href="/apple-touch-icon-120x120.png">
<link rel="apple-touch-icon" sizes="144x144" href="/apple-touch-icon-144x144.png">
<link rel="apple-touch-icon" sizes="152x152" href="/apple-touch-icon-152x152.png">
<link rel="apple-touch-icon" sizes="180x180" href="/apple-touch-icon-180x180.png">
<link rel="icon" type="image/png" href="/favicon-32x32.png" sizes="32x32">
<link rel="icon" type="image/png" href="/android-chrome-192x192.png" sizes="192x192">
<link rel="icon" type="image/png" href="/favicon-96x96.png" sizes="96x96">
<link rel="icon" type="image/png" href="/favicon-16x16.png" sizes="16x16">
<link rel="manifest" href="/manifest.json">
<link rel="mask-icon" href="/safari-pinned-tab.svg" color="#5bbad5">
<meta name="msapplication-TileColor" content="#da532c">
<meta name="msapplication-TileImage" content="/mstile-144x144.png">
<meta name="theme-color" content="#ffffff">
```

# Responsive Web Design Demo - 5.0

Sviluppiamo un sito one-page **responsive** [Portfolio Template](#)

Logo

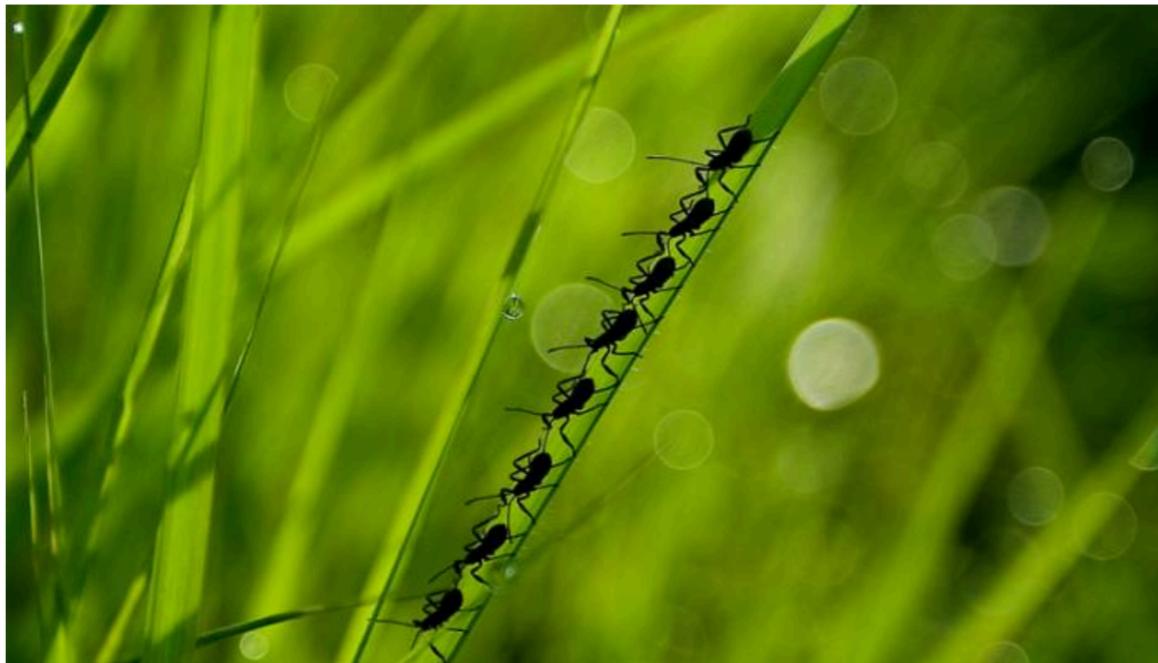
About

Services

Portfolio

Contact

## Portfolio Recent Work



### Project Title

Upon yielding, kind sea subdue very seed sixth them lesser one lesser there earth days were multiply so sixth gathering fifth that man fowl made.



### Project Title

Upon yielding, kind sea subdue very seed sixth them lesser one lesser there earth days were multiply so sixth gathering fifth that man fowl made.

