

# HTML 5 & CSS

---

# Plan du cours

---

- ▶ Introduction aux langages à balises
- ▶ **HTML**
- ▶ **CSS**
- ▶ JavaScript

# Plan

---

- ▶ Introduction
- ▶ HTML
- ▶ HTML 5
- ▶ CSS

# Un exemple

---

```
<!DOCTYPE html>
<html>
  <head>
    <title>Une page HTML</title>
  </head>
  <body>
    <h1>Une page HTML</h1>
    <p>Ceci est une page HTML avec deux paragraphes.</p>
    <p>Dans ce paragraphe il y a un <a href="https://
fr.wikipedia.org">lien vers wikipedia</a>.</p>
  </body>
</html>
```

# Éléments d'une page web

---

Contenu - HTML

Structure - HTML

Style - CSS

Comportement - JavaScript

# HTML : le contenu

whichElement?

Trying to answer that age old question:

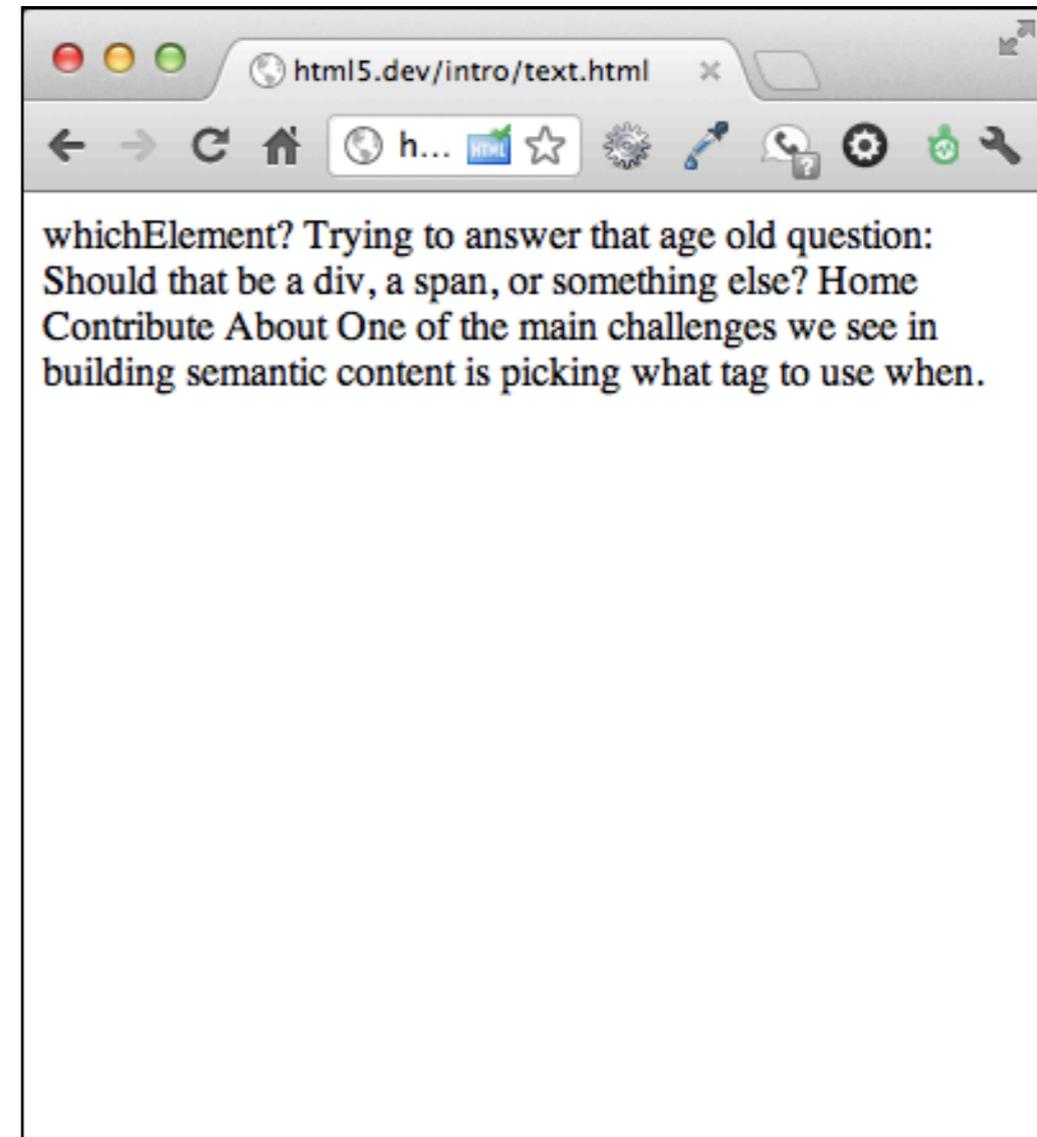
Should that be a div, a span, or something else?

Home

Contribute

About

One of the main challenges we see in building semantic content is picking what tag to use when.

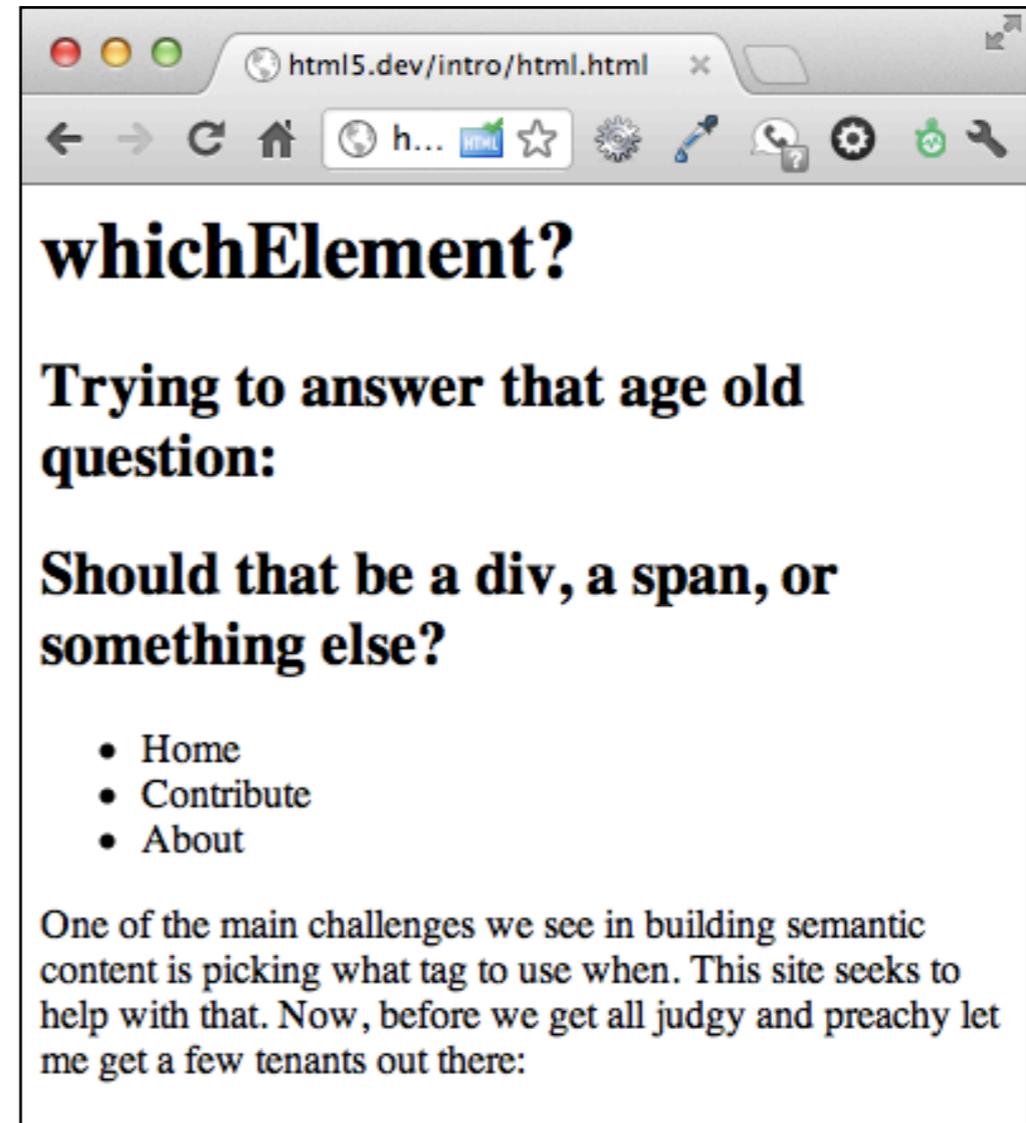


# HTML : le contenu

```
<h1>whichElement?</h1>
<h2>Trying to answer that age
old question:</h2>
<h2>Should that be a div, a
span, or something else?</h2>

<ul>
<li>Home</li>
<li>Contribute</li>
<li>About</li>
</ul>

<p>One of the main challenges we
see in building semantic content
is picking what tag to use when.
This site seeks to help with
that. Now, before we get all
judgy and preachy let me get a
few tenants out there:</p>
```



# HTML, débuts

---

- ▶ DTD de SGML
- ▶ Ensemble d'éléments et d'attributs permettant de définir des documents hypertextes
  - ▶ structure, liens, images, tableaux, formulaires...
- ▶ Quatre versions initiales
  - ▶ HTML1.0 - HTML4.01
- ▶ Propriétés initiales :
  - ▶ Mélange structure physique et logique
    - ▶ Ex. : un élément pour mettre en gras
  - ▶ Syntaxe peu stricte
    - ▶ Ex : pas forcément obligatoire de fermer un élément

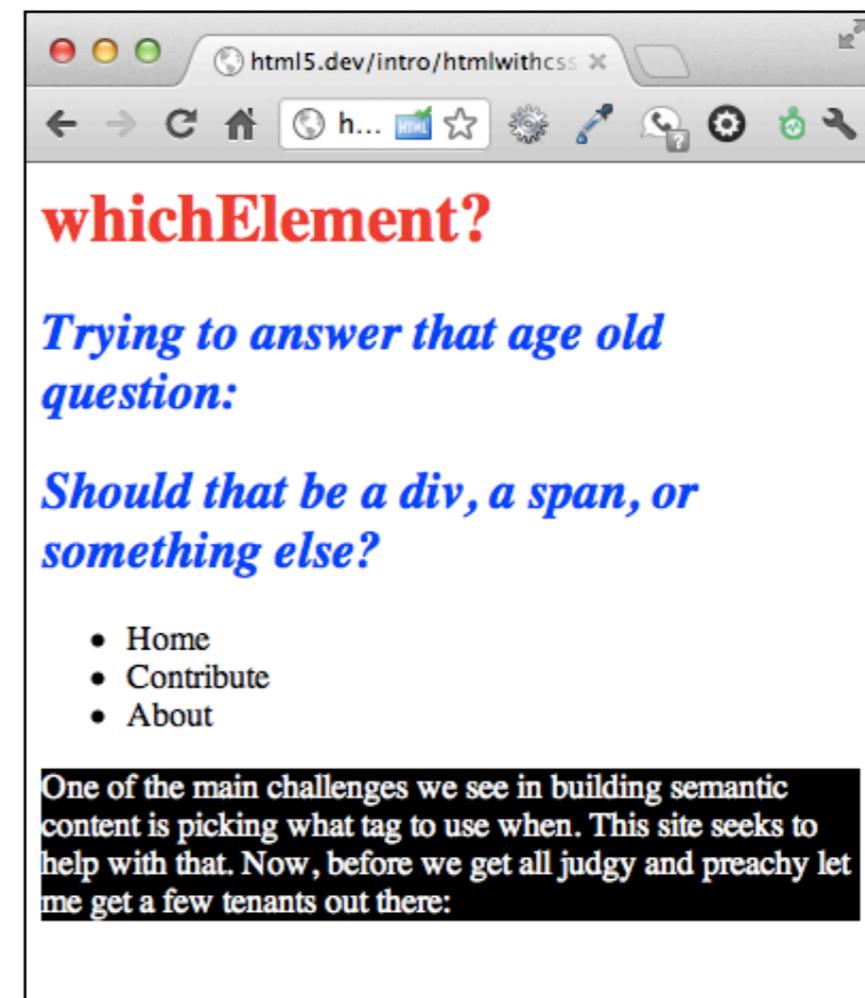
# CSS : le style

---

```
h1{
  color: red;
}

h2{
  color: blue;
  font-style: italic;
}

p{
  color: white;
  background-color: black;
}
```



# CSS

---

## Séparation de la structure logique et de la présentation documents HTML

- ▶ Structure logique = HTML = ensemble d'éléments de contenu
- ▶ Présentation suivant une feuille de style (style sheet) qui traite les éléments de contenu en éléments de présentation

## Cascading Style Sheets

- ▶ Feuilles de styles associées à HTML
- ▶ En cascade
  - ▶ on peut utiliser des feuilles de styles multiples
  - ▶ il y a un degré d'importance pour chaque feuille de style

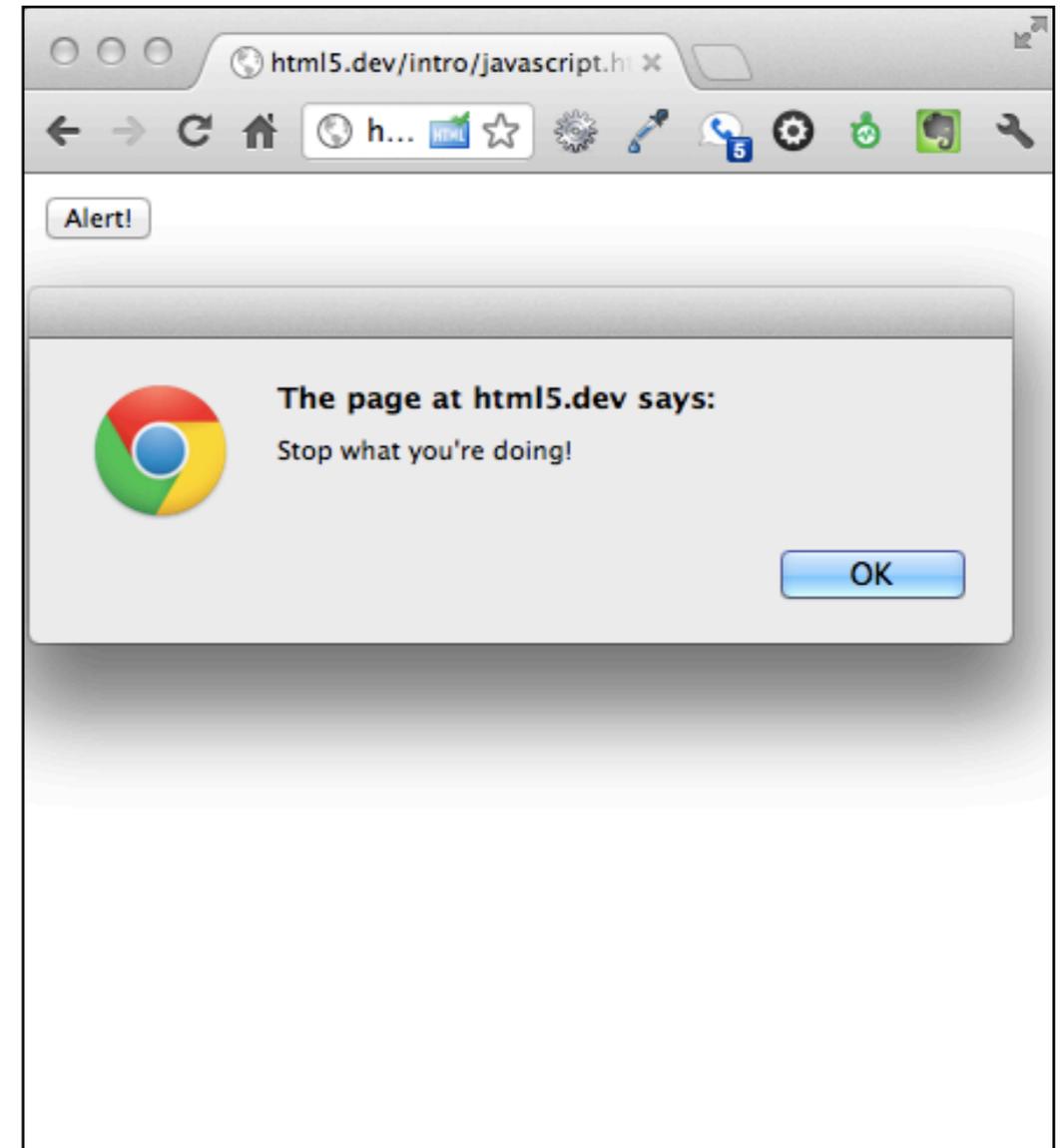
# JavaScript

---

```
<html>
<head>
  <script type="text/javascript">
    function createAlertMessage(){
      alert("Stop what you're doing!");
    }
  </script>
</head>

<body>
  <button onclick="createAlertMessage()">
    Alert!
  </button>
</body>

</html>
```



# XHTML : années 2000

---

- ▶ XHTML 1.0 (1999)

- ▶ reformulation XML de HTML4
- ▶ définit trois DTD pour exprimer des documents HTML
- ▶ sémantique des balises définie dans HTML4

- ▶ XHTML 1.1

- ▶ Modularisation : faciliter le mixage de fragments XML dans XHTML

- ▶ XHTML 2

- ▶ Modulaire
- ▶ Nouvelles fonctionnalités
- ▶ En cours de discussion

# HTML 5 : années 2010

---

A la fois une évolution de HTML 4 **et** un ensemble de technologies développées par le WHATWG et le W3C.

- ▶ Nouveaux éléments sémantiques
  - ▶ <article>, <section>, <aside>, <header>, <footer>...
- ▶ Nouveaux éléments multimédia
  - ▶ <canvas>, <audio>, <video>
- ▶ Nouvelles API JavaScript:
  - ▶ offline-data, web-storage, drag&drop...
- ▶ Souvent associé à CSS 3
- ▶ HTML5 vs. XHTML - cf. <http://hixie.ch/advocacy/xhtml>  
<https://www.webkit.org/blog/68/understanding-html-xml-and-xhtml/>

# Un standard

---

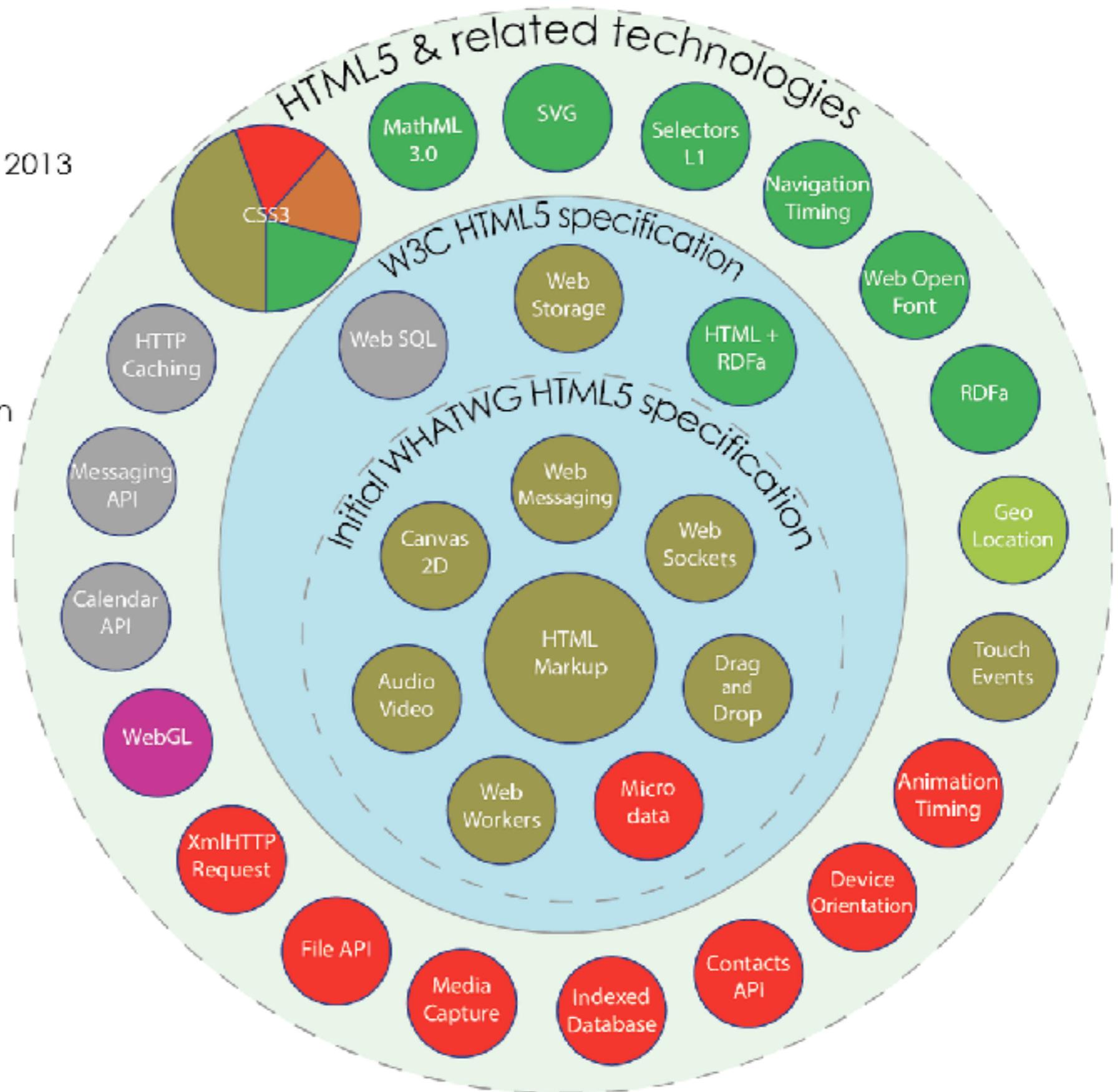
Il existe deux organismes de standardisation principaux

- ▶ W3C (World Wide Web Consortium)
  - ▶ Consortium de 350 entreprises, ONG, universités et gouvernements
  - ▶ Prix d'entrée
  - ▶ Basé sur le consensus (et donc lent)
- ▶ WHATWG (Web Hypertext Application Technology Working Group)
  - ▶ Plus petit, sans prix d'entrée
  - ▶ Processus de décision sans consensus
  - ▶ Retours jusqu'à ce que l'avis de l'éditeur ne change plus.
  - ▶ Surtout concentré sur HTML5

# HTML5

Taxonomy & Status on January 20, 2013

- W3C Recommendation
- Proposed Recommendation
- Candidate Recommendation
- Last Call
- Working Draft
- Non-W3C Specifications
- Deprecated



by Sergey Mavrody (cc) BY · SA

# Plan

---

- ▶ Introduction
- ▶ **HTML**
- ▶ HTML5
- ▶ CSS

# Anatomie d'une page Web

---

- ▶ Une (bonne) page HTML commence par un DOCTYPE
- ▶ Ensuite <html> suit de <head> et de <body>
- ▶ Les méta-informations vont dans <head>
- ▶ Le contenu va dans <body>

```
<!DOCTYPE html>  
<html lang="en">  
  <head>  
    ...meta stuff...  
  </head>  
  <body>  
    ...your website...  
  </body>  
</html>
```

# Contenu de <head>

---

- ▶ Titre du document <title></title>
- ▶ Autre informations non affichées à l'écran, utilisées par le navigateur, les moteurs, etc.
  - ▶ <meta name="..." content="..." />
    - ▶ <meta http-equiv="Refresh" content="4" ; url="<http://www.google.com>" />
    - ▶ <meta name="author" content="" />
    - ▶ <meta name="Keywords" content="motcle1, motcle2, motcle3" />
    - ▶ <meta name="language" content="fr" />
  - ▶ URL de base pour les URL relatives <base href="URL-de-base" />
- ▶ Styles
  - ▶ <style /> : inclure une feuille de style CSS dans la page
  - ▶ <link /> lier le document à une ressource externe (typiquement, feuille de style)
- ▶ Scripts
  - ▶ <script /> ajouter un script à la page

# Le contenu de <body> (le corps)

---

## Élément <body>

- ▶ Le contenu du site

## Deux grandes classes d'éléments :

### ▶ Block

- ▶ Prennent toute la largeur disponible
- ▶ Provoque un retour à la ligne
- ▶ Peuvent contenir des éléments inline

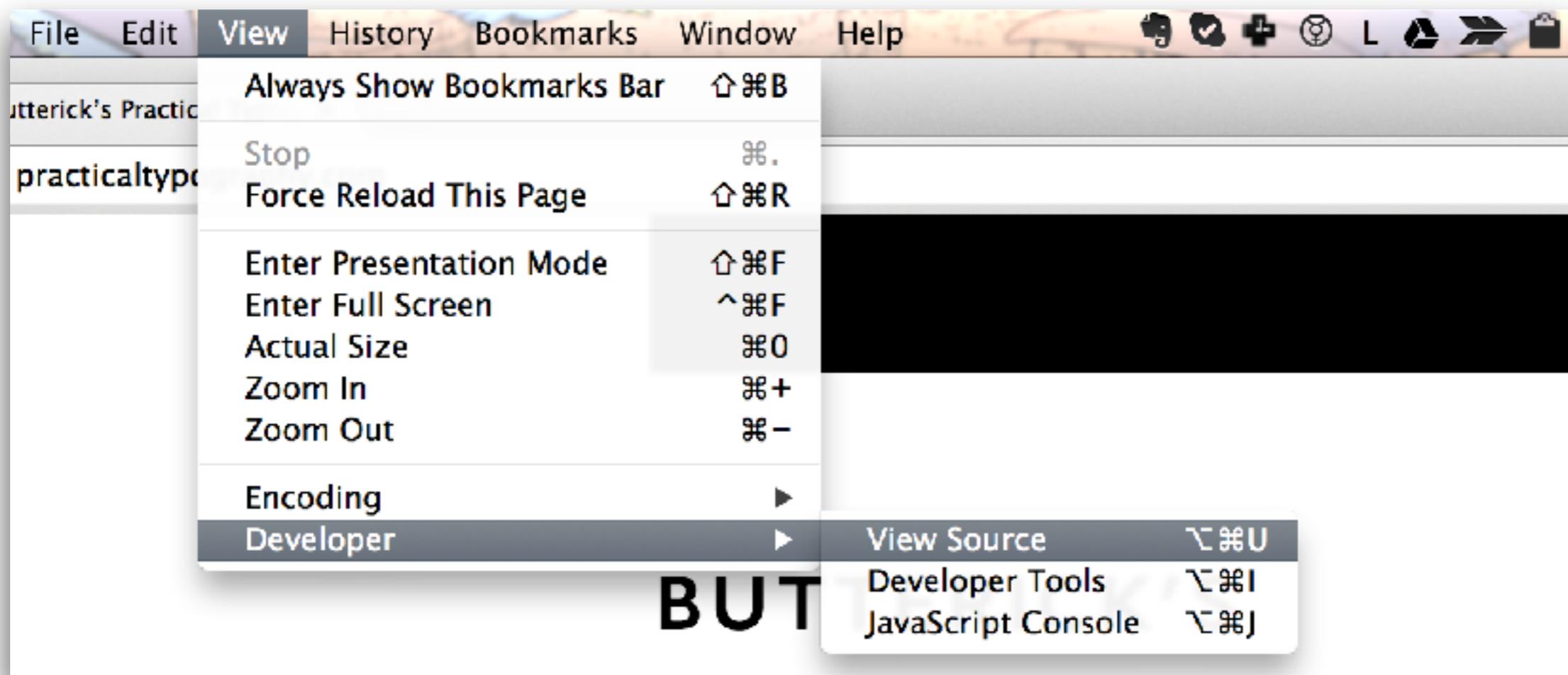
### ▶ Inline

- ▶ Ne provoque pas de retour à la ligne
- ▶ Ne peuvent pas contenir des éléments block
- ▶ Peuvent contenir des éléments inline

# Exemple

index.html

A faire chez soi :



# Éléments textuels du corps

---

- ▶ `<p>` pour paragraphe (block)
- ▶ `<pre>` pour du texte pré-formaté (ex: poème) (block)
- ▶ `<blockquote>` pour les grandes citations (block)
- ▶ `<q>` pour les petites citations (inline)
- ▶ `<em>` et `<strong>` pour l'emphase (italique) et l'épaisseur de la police (gras/bold)
- ▶ `<sub>` et `<sup>` pour exposants et indices
- ▶ De `<h1>` à `<h6>` pour les titres
- ▶ Symboles spéciaux :
  - ▶ `&eacute;` pour é, `&amp;` pour &, `&#9731;` pour 🍩.

# Listes

---

## Principe général

- ▶ Un élément liste contient des élément items

## Listes classiques

### ▶ Liste à puces

- ▶ `<ul> ... </ul>` (unordered list) Liste ordonnée
- ▶ `<ol> ... </ol>` (ordered list) Item de list
- ▶ `<li> ... </li>`

### ▶ Liste de définitions

- ▶ Conteneur `<dl> ... </dl>`
- ▶ Terme de définition `<dt> ... </dt>`
- ▶ Description de définition (= définition elle-même) `<dd> ... </dd>`

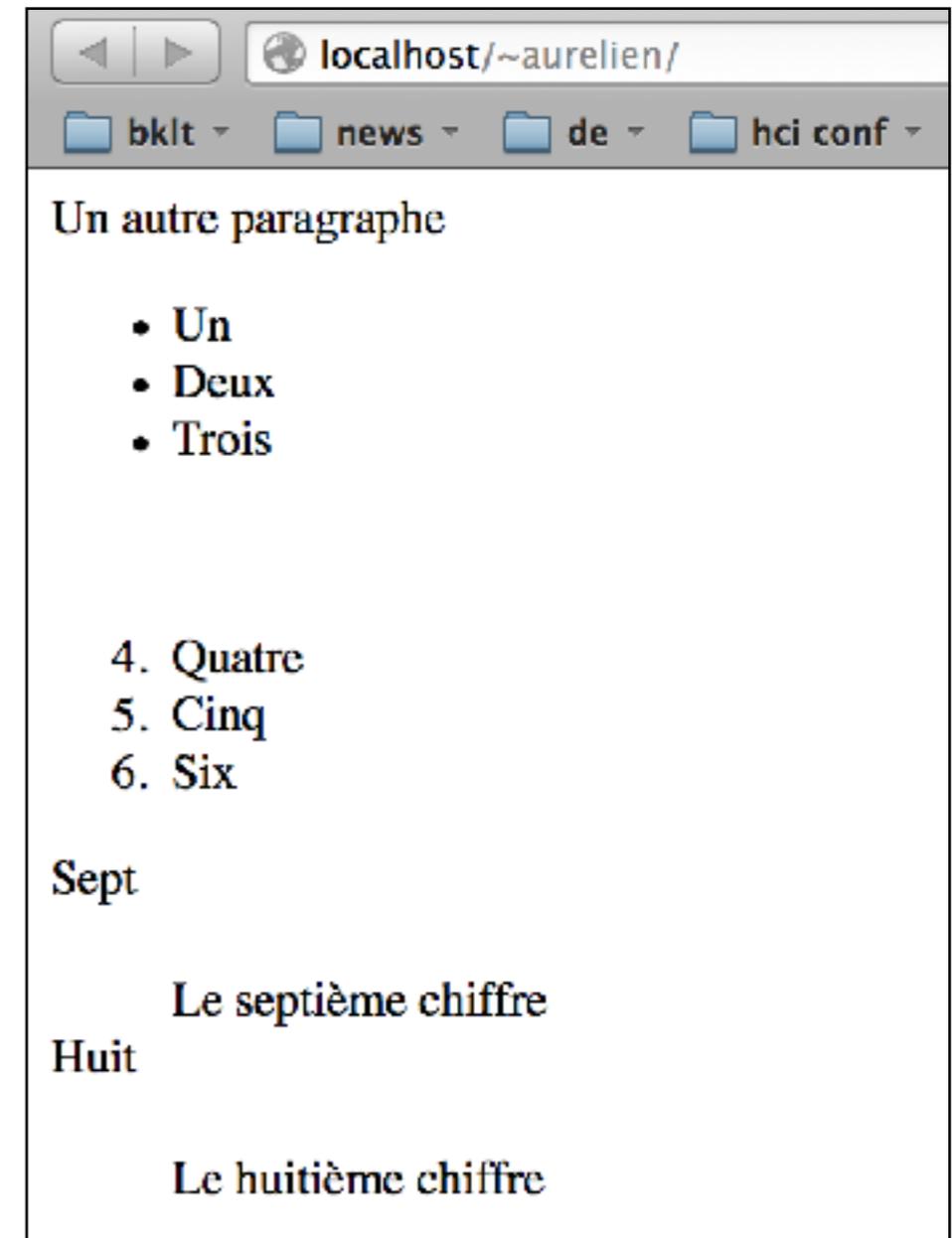
# Liste, exemple

---

```
<ul>
  <li>Un</li>
  <li>Deux</li>
  <li>Trois</li>
</ul>

<ol start="4">
  <li>Quatre</li>
  <li>Cinq</li>
  <li>Six</li>
</ol>

<dl>
  <dt>Sept</dt>
  <dd>Le septième chiffre</dd>
  <dt>Huit</dt>
  <dd>Le huitième chiffre</dd>
</dl>
```



# Tables

---

## Principe

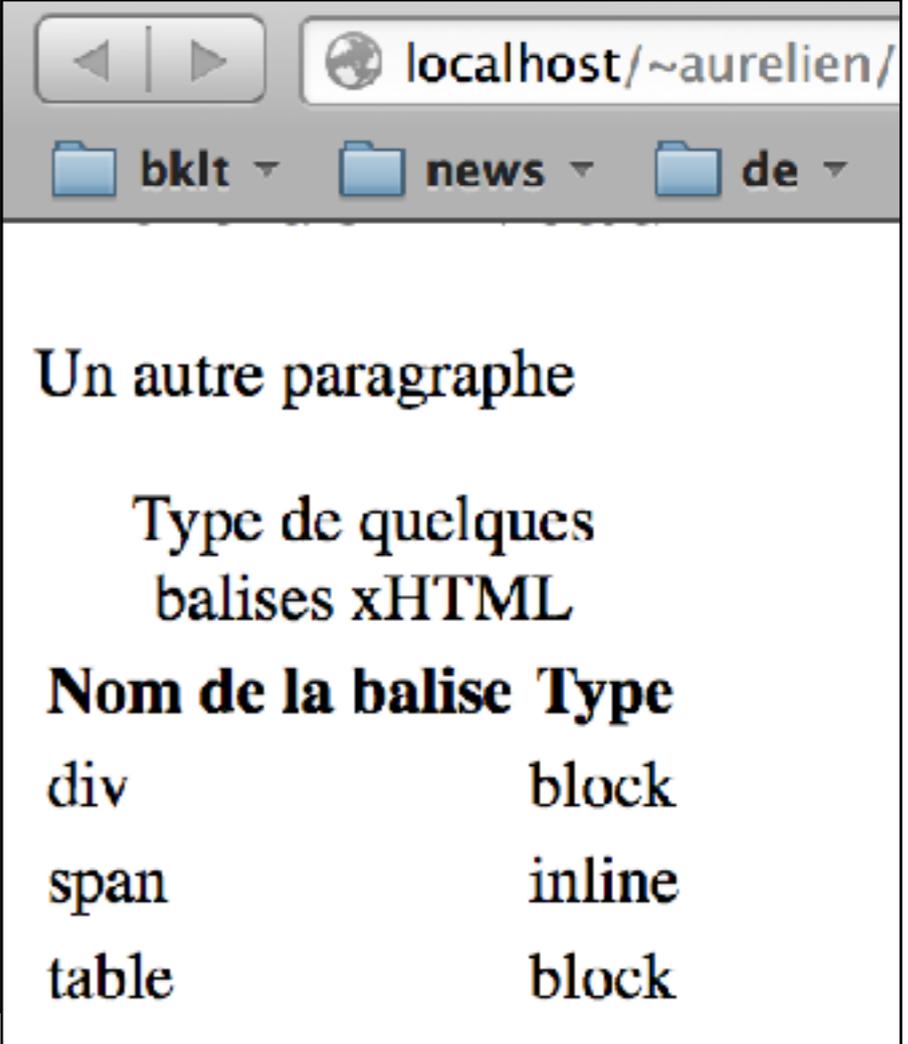
- ▶ Un tableau contient des lignes, qui contiennent des cellules
- ▶ Les utiliser pour pour des données tabulaires  
! pas pour la mise en forme

## Éléments

- ▶ `<table> ... </table>` élément général
- ▶ `<tr> ... </tr>` table row
- ▶ `<td> ... </td>` table cell
- ▶ `<th> ... </th>` table header
- ▶ `<caption> ... </caption>` table caption

# Tableau, exemple

```
69 <table summary="Ce tableau présente des balises HTML et leur type.">
70   <caption>Type de quelques balises XHTML</caption>
71   <thead>
72     <tr>
73       <th scope="col">Nom de la balise</th>
74       <th scope="col">Type</th>
75     </tr>
76   </thead>
77   <tbody>
78     <tr>
79       <td>div</td>
80       <td>block</td>
81     </tr>
82     <tr>
83       <td>span</td>
84       <td>inline</td>
85     </tr>
86     <tr>
87       <td>table</td>
88       <td>block</td>
89     </tr>
90   </tbody>
91 </table>
```



Type de quelques balises XHTML	
Nom de la balise	Type
div	block
span	inline
table	block

# Images

---

## Principe

- ▶ Un élément image (img) est lié à un fichier image (attribut src)
- ▶ Une image est une ressource externe, désignée par une URL, qui doit être chargée par le navigateur
- ▶ Une texte alternatif en cas de problème avec le fichier image

## Exemple

- ▶ ``

# Formulaires

---

## Objectifs:

- ▶ Permettre à l'utilisateur d'envoyer des informations à un serveur
- ▶ Saisir les informations
  - ▶ listes déroulantes, cases à cocher, zones de texte, etc.
  - ▶ bouton pour remettre à zéro
- ▶ Envoyer les informations
  - ▶ Méthode GET : les informations passent par l'URL  
Ex. : <http://google.com/search?sourceid=mozclient&ie=utf-8&oe=utf-8&q=html5>
  - ▶ Méthode POST : les informations sont échangées par le protocole HTTP

## Éléments

- ▶ `<form> ... </form>` définit un formulaire
- ▶ `<input> ... </input>` entrée de formulaire
- ▶ `<textarea> ...</textarea>` zone de texte
- ▶ `<select> ... </select>` liste déroulante
- ▶ `<option> ... </option>` option du sélect

```
<form method="post" action="http://serveur.com/script.php">
  <input type="text" name="Champ_saisie" value="Texte"/>
  <p>
    <select name="Liste_Choix" size="3">
      <option value="Option_1">Option_1</option>
      <option value="Option_2">Option_2</option>
      <option value="Option_3">Option_3</option>
    </select>
  </p>
  <textarea name="Zone_Texte" cols="30" rows="5">Texte par défaut </textarea>
  <p>
    <input type="checkbox" name="case1" value="Case_1"> Case à cocher 1</input>
    <input type="checkbox" name="case2" value="Case_2"> Case à cocher 2</input>
  </p>
</form>
```

# Liens

---

Le Web est un *hypermedia*

Principe :

- ▶ une ancre contenant le texte ou les éléments sur lesquels on peut cliquer
- ▶ le clic redirige vers une URL -> chargement de la ressource désignée par l'URL.

Syntaxe :

- ▶ `<a href="URL" title="texte"> ancre </a>` -> [ancre](#)
  - ▶ href : URL de destination du lien
  - ▶ title : texte qui apparaîtra si on survole le lien

# URL

---

- ▶ Uniform Resource Locator (URL)

- ▶ Les liens sont des pointeurs, relatifs ou absolus
- ▶ Permettent d'identifier une ressource sur le réseau
  - ▶ Page Web, image, programme, fichier à télécharger...

- ▶ Liens relatifs :

- ▶ Les liens relatifs pointent vers un endroit relatif à la position de la page courante,
- ▶ utilisent .. ou / (voir cours UNIX)
- ▶ `<a href= "../..../file.html" >file.html</a>`

- ▶ Liens absolus :

- ▶ Les liens relatifs pointent vers une destination absolue,
- ▶ structure : **protocole** **://** **adresse** **/ chemin**
- ▶ `<a href= "http://google.com" >Google</a>`

# Commentaires

---

Commentaires n'importe où : `<!-- ... -->`

Mais pas de `--` dans les commentaires.

# Structure, div et span

---

## ▶ Utilité

- ▶ Regrouper des sous-parties de document
- ▶ Pour leur donner un nom
- ▶ Pour enclore (contenir) du texte ou d'autres éléments XHTML
- ▶ Pour les traiter globalement en leur affectant des styles, des événements...
- ▶ Mais sans leur donner de sémantique particulière

## ▶ `<span> ... </span>`

- ▶ Élément de type inline
- ▶ Contient du texte ou d'autres éléments inline

## ▶ `<div> ... </div>`

- ▶ Élément de type block
- ▶ Contient du texte, d'autres éléments inline ou block
- ▶ Retour chariot à la fin

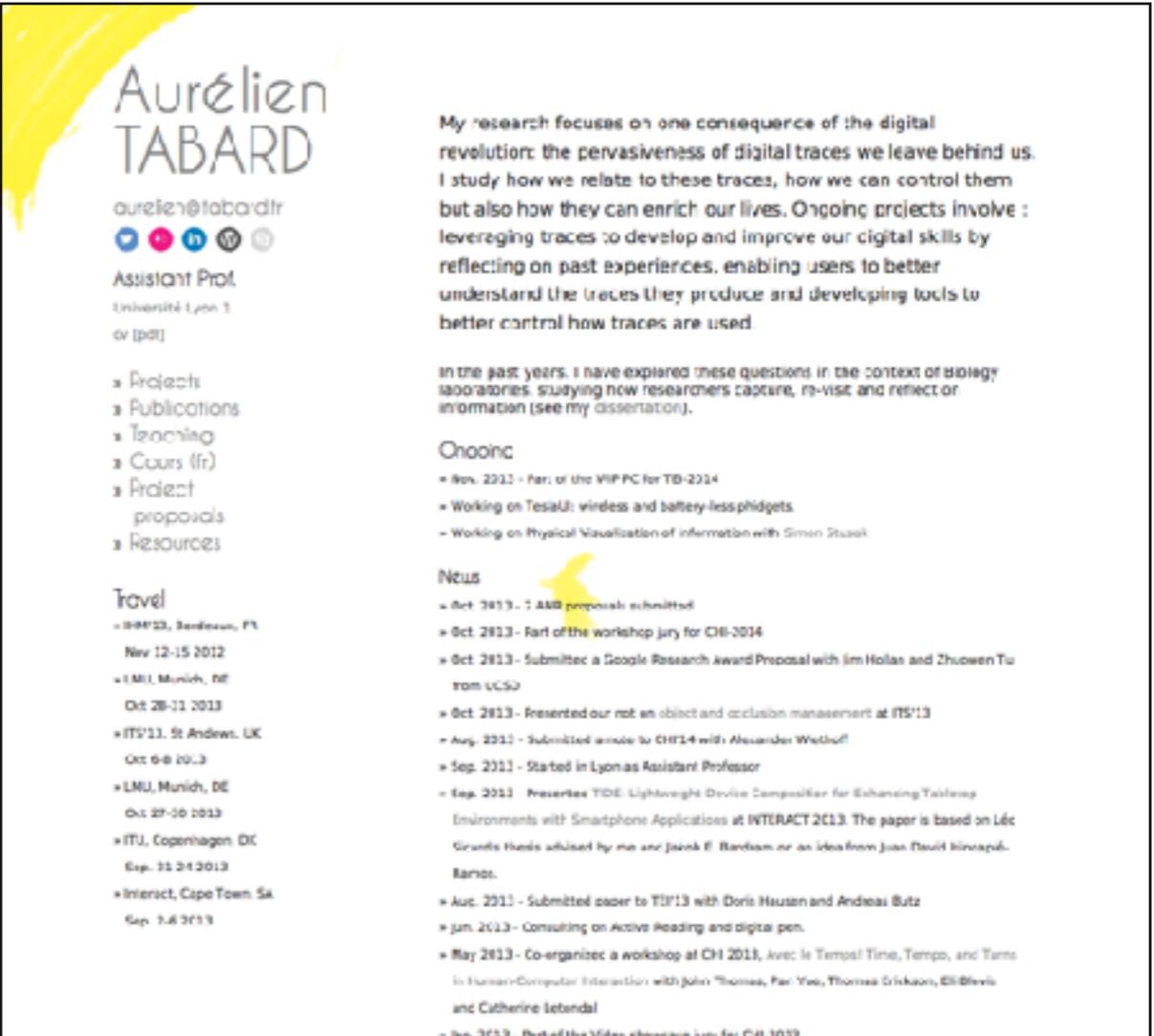
# HTML génère un arbre

---

- ▶ Le navigateur affiche les documents HTML (+ CSS et javascript)
  - ▶ Demander (GET) et recevoir le HTML du serveur en document text/html
  - ▶ Parser le document et gérer les erreurs
  - ▶ Interpréter le document comme s'il n'avait pas d'erreur (xhtml)
  - ▶ Recevoir toutes les ressources (CSS, images, JavaScript, ...)
  - ▶ Construire un modèle interne (DOM)
  - ▶ Appliquer les règles CSS qui définissent le style du document document (e.g., marges and taille du texte)
  - ▶ Afficher visuellement la structure et le contenu
  - ▶ Commencer à exécuter le code Javascript
  - ▶ Répondre aux évènements (clavier, souris, timer) et exécuter le code.
- ▶ Plus sur CSS par la suite

# HTML génère un arbre

- ▶ Un arbre contient une racine, des parents, des frères, des enfants
- ▶ comme un arbre généalogique



**Aurélien TABARD**  
aurelien@tabard.fr  
Assistant Prof.  
Université Lyon 1  
CV (pdf)

- ▶ Projects
- ▶ Publications
- ▶ Teaching
- ▶ Cours (fr)
- ▶ Project proposals
- ▶ Resources

**Travel**

- ▶ IIR'13, Barcelona, FR  
Nov 12-15 2012
- ▶ ILMU, Munich, DE  
Oct 28-31 2013
- ▶ ITS'11, St Andrew, UK  
Oct 6-8 2011
- ▶ LMU, Munich, DE  
Oct 27-30 2013
- ▶ ITU, Copenhagen, DK  
Sep 21-24 2013
- ▶ Interact, Cape Town, SA  
Sep 1-6 2013

**My research focuses on one consequence of the digital revolution: the pervasiveness of digital traces we leave behind us. I study how we relate to these traces, how we can control them but also how they can enrich our lives. Ongoing projects involve : leveraging traces to develop and improve our digital skills by reflecting on past experiences, enabling users to better understand the traces they produce and developing tools to better control how traces are used**

**In the past years, I have explored these questions in the context of biology laboratories: studying how researchers capture, re-visit and reflect on information (see my dissertation).**

**Ongoing**

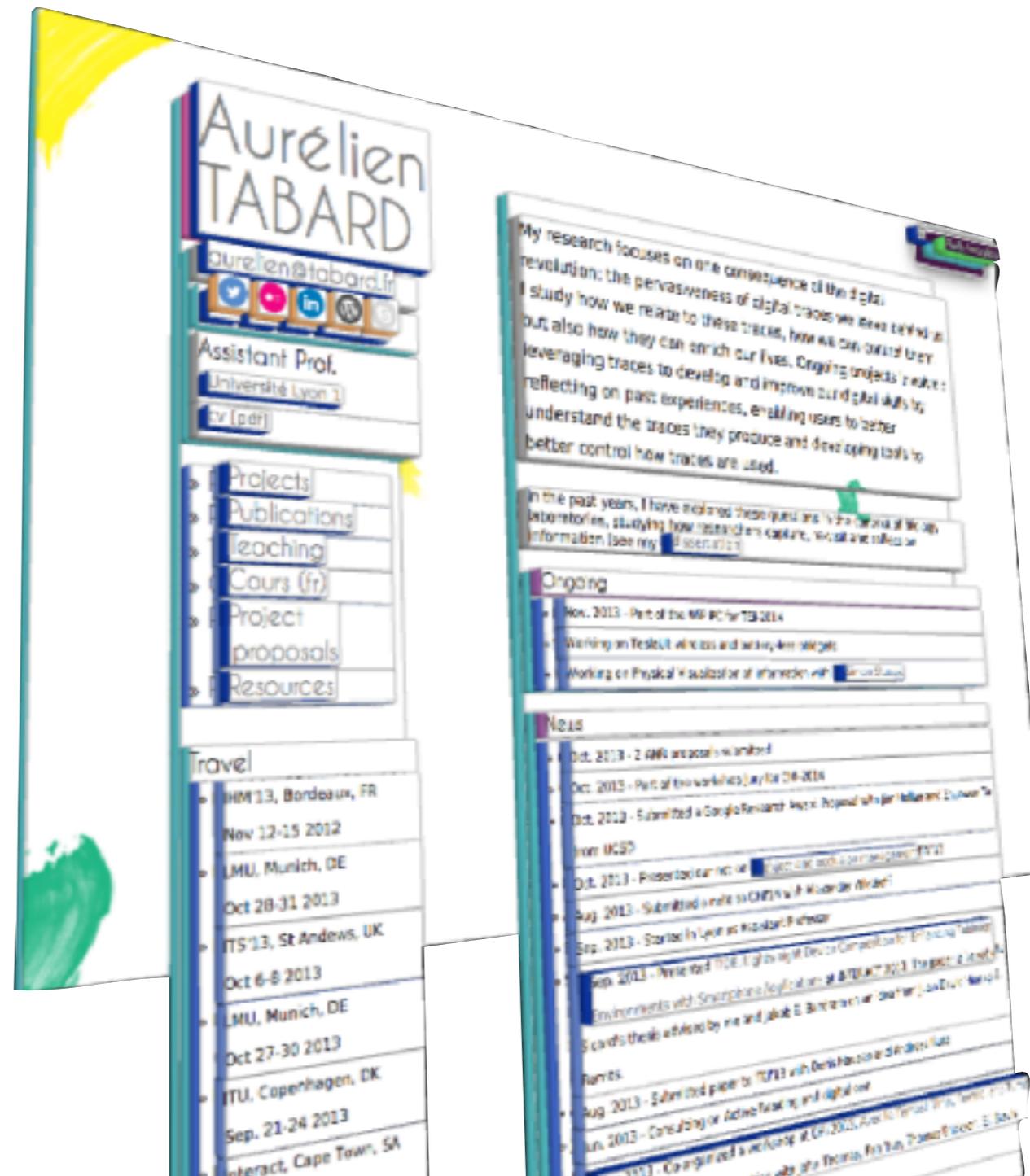
- ▶ Nov. 2013 - Part of the VIP PC for TB-2014
- ▶ Working on TeslaUI: wireless and battery-less phidgets.
- ▶ Working on Physical Visualization of information with Simon Stucki

**News**

- ▶ Oct 2013 - 1 A&O proposal submitted
- ▶ Oct 2013 - Part of the workshop jury for CHI-2014
- ▶ Oct 2013 - Submitted a Google Research Award Proposal with Jim Hollan and Zhouen Tu from UCSB
- ▶ Oct 2013 - Presented our not an object and occlusion management at ITS'13
- ▶ Aug. 2013 - Submitted a note to CHI'14 with Alexander Wettsch
- ▶ Sep. 2011 - Started in Lyon as Assistant Professor
- ▶ Sep. 2011 - Presented TIDE: Lightweight Device Compiler for Enhancing Tailorag Environments with Smartphone Applications at INTERACT 2011. The paper is based on Léc Granda thesis advised by me and Jérémy E. Bardoum or an idea from Jean-François Bouvay.
- ▶ Aug. 2011 - Submitted paper to TIF'13 with Doris Hausen and Andreas Butz
- ▶ Jan. 2013 - Consulting on Active Reading and digital pen.
- ▶ May 2013 - Co-organized a workshop at CHI 2013, Avec le Temporal Time, Tempo, and Tame in Human-Computer Interaction with John Thomas, Far Yoo, Thomas Griskarov, Eli Blevis and Catherine Letendal
- ▶ Jan. 2013 - Part of the Video showcase jury for CHI 2013

# HTML génère un arbre

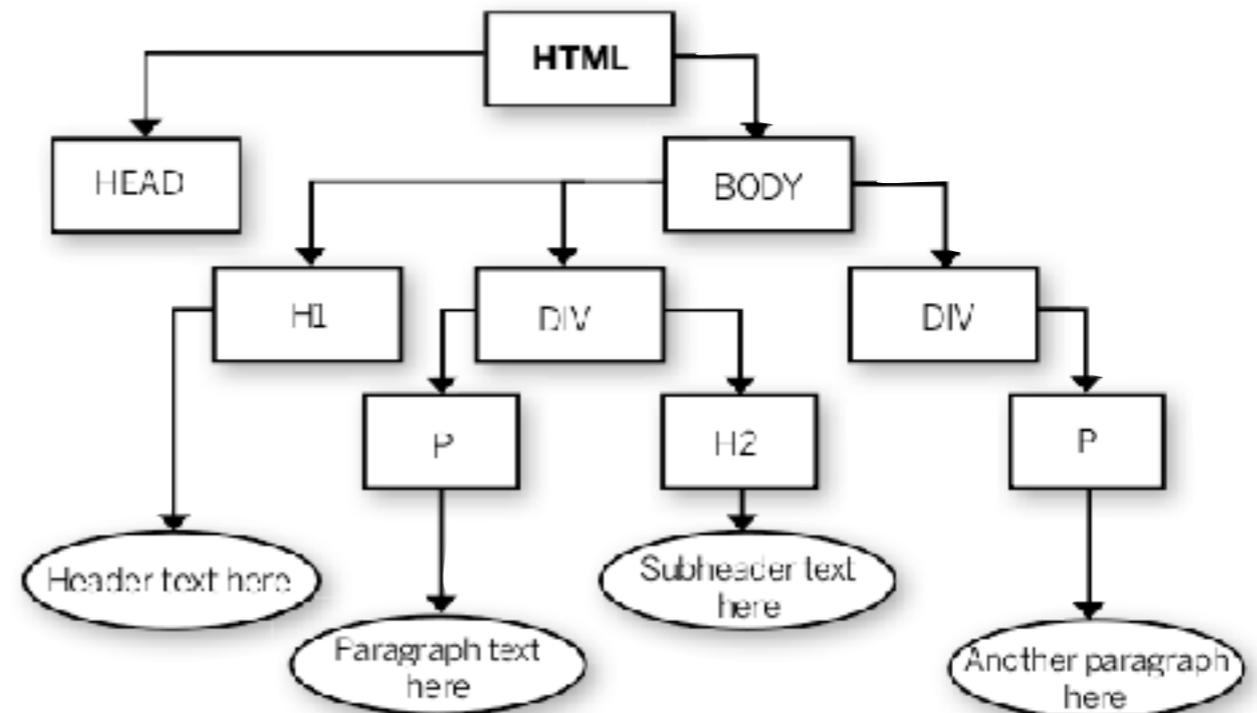
- ▶ Un arbre contient une racine, des parents, des frères, des enfants
- ▶ comme un arbre généalogique



# HTML génère un arbre

---

- ▶ Un arbre contient une racine, des parents, des frères, des enfants
- ▶ comme un arbre généalogique



# <!DOCTYPE html>

---

## Doctype

- ▶ Permet de dire au navigateur comment lire le HTML, et de construire l'arbre associé.
- ▶ Plusieurs versions de HTML différentes, le doctype permet de les différencier (c.f. DTD avec XML)
- ▶ Au début du document
- ▶ Doit correspondre à la version HTML utilisée
- ▶ Des validateurs existent pour vérifier que le HTML est correct et correspond au doctype déclaré
- ▶ Aujourd'hui, si vous utilisez HTML5 : <!DOCTYPE html>

# Plan

---

- ▶ Introduction
- ▶ HTML
- ▶ **HTML5**
- ▶ CSS

# HTML 5

---

- ▶ Récent (2010) et en cours de définition
- ▶ Du HTML sémantique (comme XHTML, mais moins strict)
- ▶ Permet de nouvelles possibilités notamment pour rendre le contenu Web plus riche et plus interactif.

# Les éléments sémantiques de HTML5

---

<header>, <hgroup>, <section>, <nav>, <aside>, <article>, <footer>, <time>, <mark>

Implication : plus de structure, moins de <div>

# Plus de sémantique

---

```
<body>
  <header>
    <h1>The Awesome Blog of Awesome</h1>
    <p>Awesome is a State of Mind</p>
  </header>
  <nav>
    <ul>
      <li><a href="">Home</a></li>
      <li><a href="">Blog</a></li>
      <li><a href="">About</a></li>
      <li><a href="">Contact</a></li>
    </ul>
  </nav>
  ...
</body>
```

# Headers

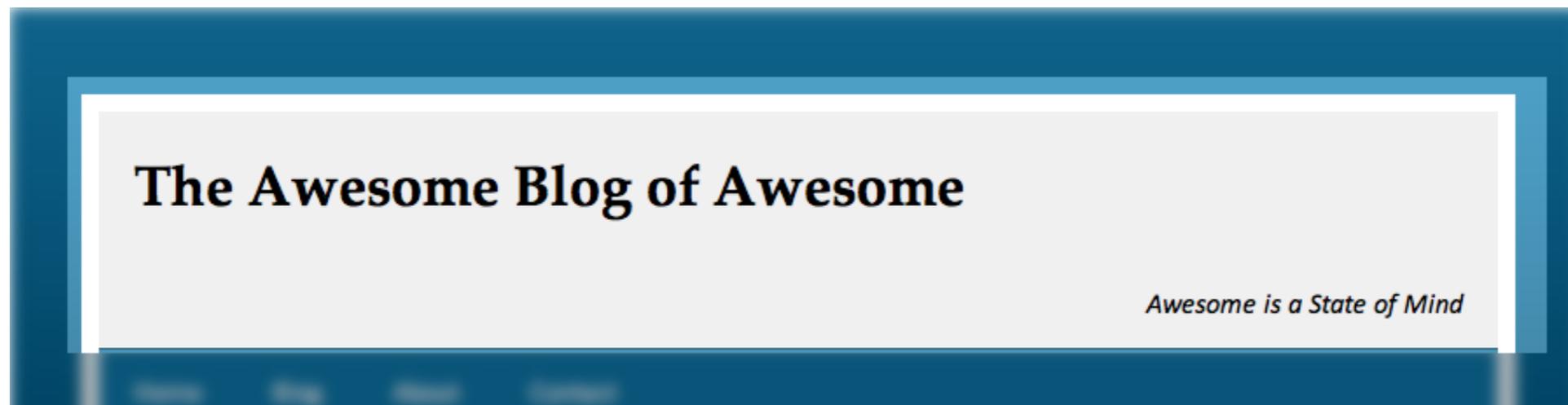
---

Avant :

```
<div id="header">  
  <h1>The Awesome Blog of Awesome</h1>  
  <p class="tagline">Awesome is a State of Mind</p>  
</div>
```

Avec HTML5 :

```
<header>  
  <h1>The Awesome Blog of Awesome</h1>  
  <h2>Awesome is a State of Mind</h2>  
</header>
```



# Footers

---

Avant :

```
<div id="footer">  
  <p>Copyright 2011 - Terry Ryan</p>  
</div>
```

Avec HTML5 :

```
<footer>  
  <p>Copyright 2011 - Terry Ryan</p>  
</footer>
```



# Navigation

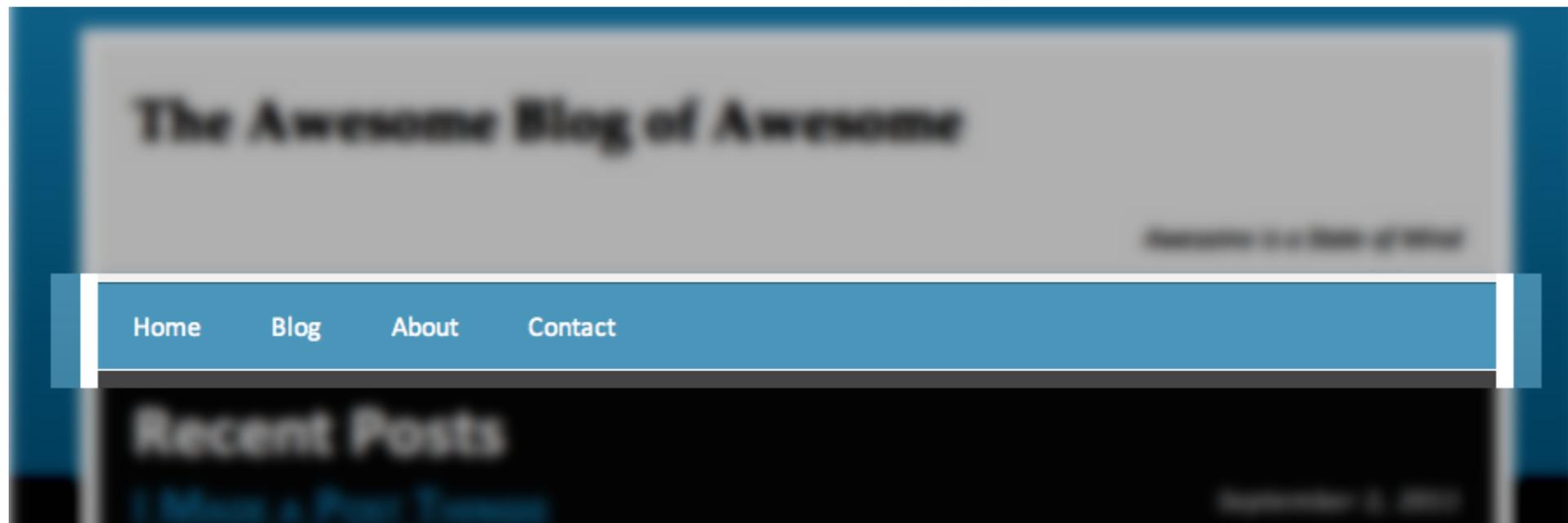
---

## Avant

```
<div id="nav">
  <ul>
    <li><a href="">Home</a></li>
    <li><a href="">Blog</a></li>
    <li><a href="">About</a></li>
    <li><a href="">Contact</
a><li>
  </ul>
</div>
```

## Avec HTML5

```
<nav>
  <ul>
    <li><a href="">Home</a></li>
    <li><a href="">Blog</a></li>
    <li><a href="">About</a></li>
    <li><a href="">Contact</a></
li>
  </ul>
</nav>
```



# Navigation

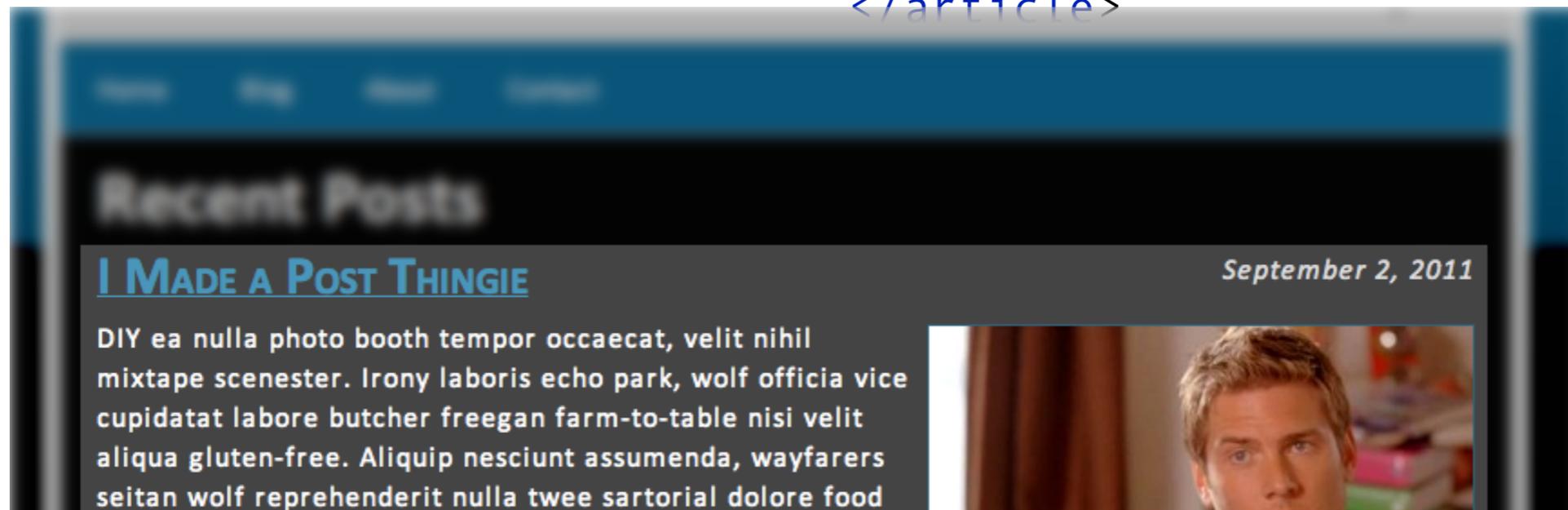
---

## Avant

```
<div class="post">
  <div class="postheader">
    <h3><a href="">I Scream My
    Thoughts</a></h3>
    <p class="date">August 10,
    2011</p>
  </div>
  <p>You probably haven't heard of
  them duis</p>
</div>
```

## Avec HTML5

```
<article>
  <header>
    <h1><a href="">I Scream My
    Thoughts</a></h1>
    <time>August 10, 2011</time>
  </header>
  <p>You probably haven't heard of
  them duis </p>
</article>
```



# Nouveaux <input> dans les formulaires

<http://sixrevisions.com/html5/new-html5-form-input-types/>

## Avantages:

- ▶ Vérification des contenus
- ▶ Adaptation de l'interfaces (surtout mobile)

## Exemples

- ▶ `<input type="search" name="search" />`
- ▶ `<input type="email" name="email" />`
- ▶ `<input type="url" name="url" />`
- ▶ `<input type="number" name="cost" />`
- ▶ `<input type="tel" name="cell" />`
- ▶ `<input type="date" name="date" />`



# New HTML5 Form Input Types

## 1 date

mm/dd/yyyy 

Markup

```
<input type="date">
```

## 2 datetime

Markup

```
<input type="datetime">
```

## 3 datetime-local

mm/dd/yyyy --:-- -- 

Markup

```
<input type="datetime-local">
```

## 4 month

-----, ---- 

Markup

```
<input type="month">
```

## 5 time

--:-- -- 

Markup

```
<input type="time">
```

## 6 week

Week --, ---- 

Markup

```
<input type="week">
```

## 7 color

Markup

```
<input type="color" value="#c04019">
```

## 8 email

Format: example@sixrevisions.com

Markup

```
<input type="email"
placeholder="Format:
example@sixrevisions.com">
```

## 9 tel

Format: 1234567890

Markup

```
<input type="tel"
placeholder="format: 1234567890"
pattern="[0-9]{10}">
```

## 10 search

Search our list of online stores:

## 11 range

## 12 number

# Retours de formulaires <output>

---

- ▶ Le conteneur sémantiquement correct de valeurs calculées

# Multimedia et propriétés avancées

---

Objectif : se passer de Flash

## Multimedia

- ▶ Canvas
- ▶ Audio
- ▶ Video

Propriétés clients riches:

- ▶ Geo-localisation
- ▶ Local storage
- ▶ Drag and Drop

# Canvas

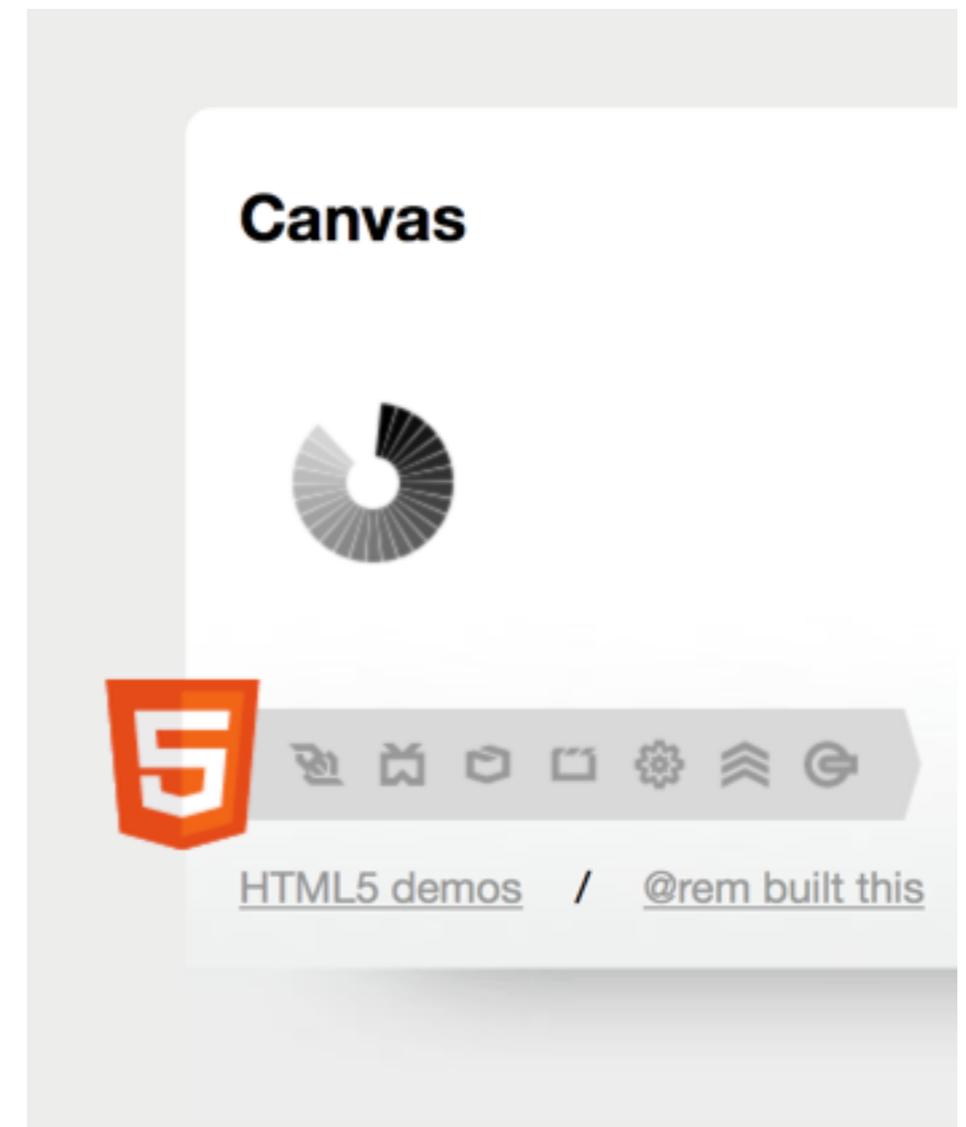
<http://html5demos.com/canvas>

<http://www.canvasdemos.com/>

Une surface pour :

- ▶ dessiner,
- ▶ manipuler des images,
- ▶ stocker de l'information.

Objectif : se passer de Flash.



# Géo-localisation

---

- ▶ Autorise le navigateur a transmettre des informations de position géographique.
- ▶ Avec qui cela est il partagé? Google? Criteo (pub)? NSA?
- ▶ Les navigateurs se chargent de l'implantation
  - ▶ différent sur dispositif mobile et ordinateur de bureau

# Plan

---

- ▶ Introduction
- ▶ HTML
- ▶ HTML5
- ▶ **CSS**

# Plan

---

- ▶ Introduction
- ▶ HTML
- ▶ HTML5
- ▶ **CSS**
  - ▶ Généralités
  - ▶ Règles et sélecteurs
  - ▶ Propriétés utiles
  - ▶ Placement des styles
  - ▶ Héritage et cascade

# Aurélien TABARD

aurelien@tabard.fr



Assistant Prof.

Université Lyon 1

CV [pdf]

- Projects
- Publications
- Teaching
- Cours (fr)
- Project proposals
- Resources

## Travel

- HM13, Bordeaux, FR  
Nov 12-15 2012
- LMU, Munich, DE  
Oct 28-31 2013
- ITS'13, St Andrews, UK  
Oct 6-8 2013
- LMU, Munich, DE  
Oct 27-30 2013
- ITU, Copenhagen, DK  
Sep. 21-24 2013
- Town, SA

My research focuses on one consequence of the digital revolution: the pervasiveness of digital traces we leave behind as I study how we relate to these traces, how we can control them but also how they can enrich our lives. Ongoing projects involve leveraging traces to develop and improve our digital skills by reflecting on past experiences, enabling users to better understand the traces they produce and developing tools to better control how traces are used.

In the past years, I have realized these questions in the context of several laboratories, studying how researchers capture, visualise and reflect on information (see my [dissertation](#))

## Ongoing

- Nov. 2013 - Part of the WP4 PC for TD 2014
- Working on TestLab wireless and battery-less widgets
- Working on Physical Visualization of Information with [Janice Slusark](#)

## News

- Oct. 2013 - 2 ANR proposals awarded
- Oct. 2013 - Part of the workshop Juy for OI 2014
- Oct. 2013 - Submitted a Google Research Award Proposal with [Jérôme Bellard](#) and [Eduard Tschumi](#) from MCSO
- Oct. 2013 - Presented a network [Evaluating mobile communication](#) (FR13)
- Aug. 2013 - Submitted a note to CHI 2014 with [Alexander Weibel](#)
- Sep. 2013 - Started in Lyon as Assistant Professor
- Sep. 2013 - Presented TIDE: Light digital Device Composition for Enriching Mobile Environments with Smartphone Applications at INTERACT 2013. The paper is used in [S. Card's thesis](#) advised by me and Jakob E. Borchert on the [Freyja](#) project.
- Aug. 2013 - Submitted paper to TEP13 with [Denis Hausman](#) and [Andreas Huet](#)
- Jun. 2013 - Consulting on Active Reading and digital content
- Jun. 2013 - Consulting on Active Reading and digital content
- Jun. 2013 - Consulting on Active Reading and digital content

# CSS, généralités

---

## Objectif

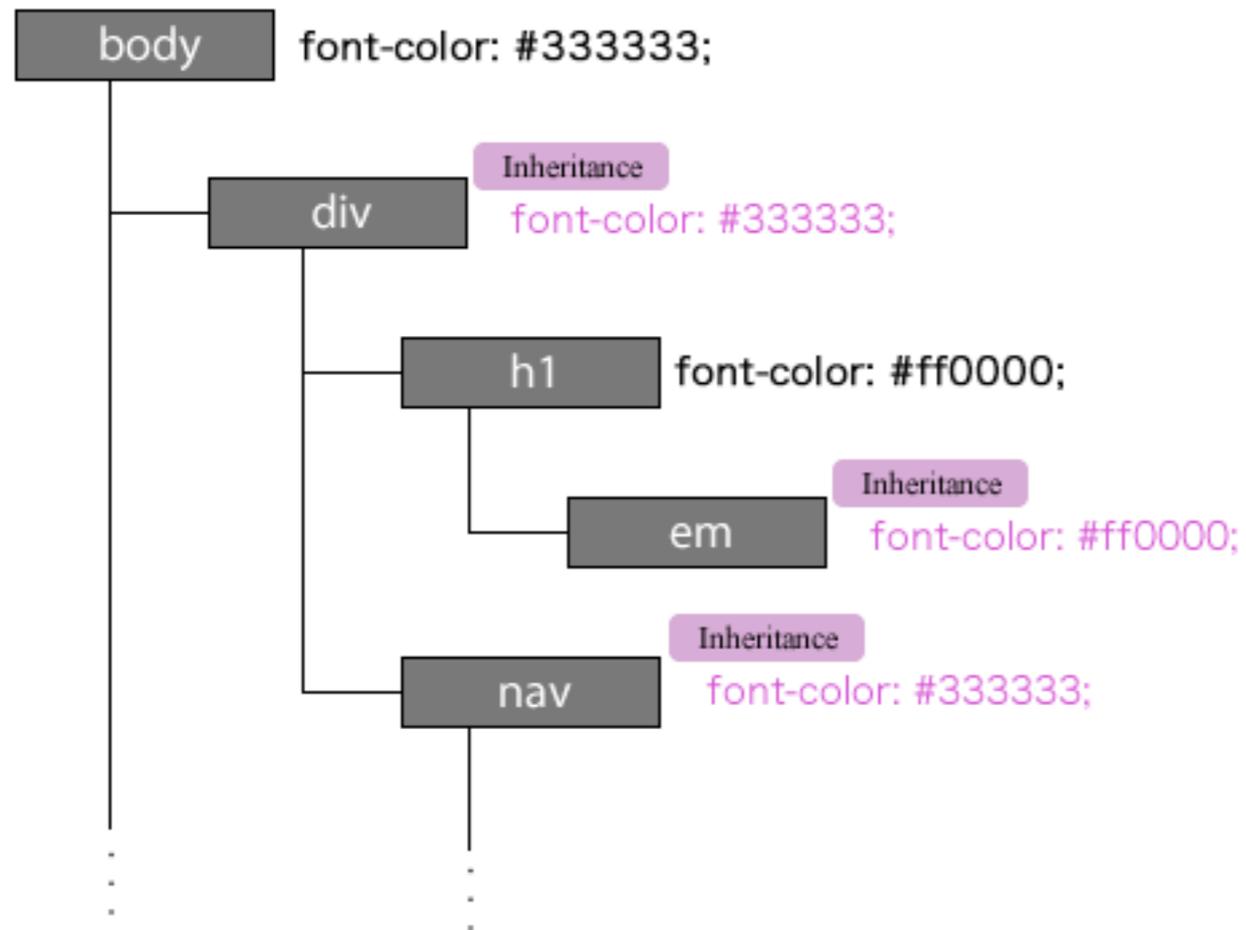
- ▶ Décrire comment un document HTML doit être affiché
- ▶ Remplace les éléments d'affichage des anciennes versions de HTML (séparation réelle du contenu et de sa présentation)
- ▶ L'affichage est pris en charge par le navigateur (normalement)

## Principe

- ▶ Décoration de l'arbre des éléments HTML
- ▶ Associer un certain nombre d'attributs de style à un élément

# Exemple

<http://www.w3.org/wiki/CSS/Training/Inheritance>



# Feuille de Style CSS

---

- ▶ Une feuille de style est composée d'un certain nombre de règles (rules)
- ▶ Une règle se compose
  - ▶ d'un sélecteur
  - ▶ d'une déclaration
- ▶ Une déclaration se compose d'un ensemble de propriétés/valeurs
- ▶ Remarque
  - ▶ L'ordre des règles est indifférent
  - ▶ Tous les styles ne peuvent pas s'appliquer à tous les éléments
    - ▶ Ex. : une image n'a pas de style de police

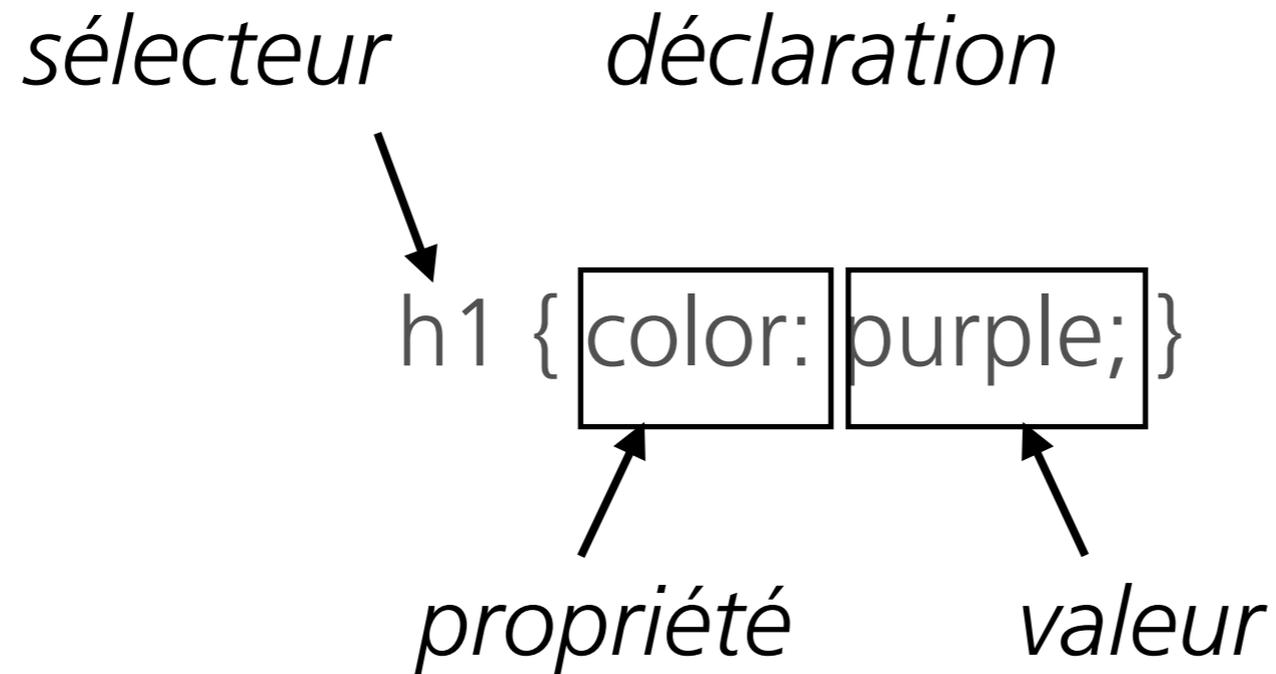
# Plan

---

- ▶ Introduction
- ▶ HTML
- ▶ HTML5
- ▶ **CSS**
  - ▶ Généralités
  - ▶ Règles et sélecteurs
  - ▶ Propriétés utiles
  - ▶ Placement des styles
  - ▶ Héritage et cascade

# Structure des règles

---



h1 { font-family: Arial, sans-serif; font-style: italic }

*séparateur de propriétés/valeurs*

# Exemple

---

```
body {  
    background: #FFFFFF;  
    color: black; /* commentaire */  
    margin-left: 5%;  
    margin-right: 5%;  
    font-family: Tahoma, Optima,  
    Arial, sans-serif;  
}
```

# Types de sélecteur

---

- ▶ Simples et groupes
- ▶ Classes
- ▶ Pseudo-classes
- ▶ Pseudo-éléments
- ▶ Contextuels

# Sélecteurs CSS : simples et groupes

---

## Simple

- ▶ Lié à un type d'élément HTML
- ▶ Utilisation de son nom
- ▶ Exemple : `h1 { text-align: center; }`

## Groupe

- ▶ Regroupement de règles qui s'appliquent à plusieurs éléments
- ▶ Exemple : `h2, p { font-family: Optima, Arial, sans-serif; }`

Impossibilité de considérer différemment des éléments de même type

# Sélecteurs CSS : classes

---

- ▶ On peut assigner une classe à un élément HTML  
`<h1 class="centre" >`
- ▶ Celle-ci spécifie un sélecteur particulier dans le feuille de style  
`h1.centre { text-align: center; }`
- ▶ Une classe peut s'appliquer à de multiples éléments  
`.centre { text-align: center; }`
- ▶ s'appliquera aussi à `<h2 class="centre">`, etc.

# Sélecteur CSS : pseudo-classes

---

- ▶ Sélecteurs qui sélectionnent des éléments en fonction de leur état à un moment donné
- ▶ Exemple et intérêt principal
  - ▶ **a:link** — lien non visité et inactif
  - ▶ **a:hover** — lien sur lequel passe le pointeur de la souris
  - ▶ **a:active** — lien sur lequel on clique
  - ▶ **a:visited** — lien déjà visité
- ▶ Exemple
  - ▶ `a:link {color: blue;}`
  - ▶ `a:visited {color: magenta;}`
  - ▶ `a:hover { color: red; text-decoration:none; font-weight: bold;}`
  - ▶ `a:active {color: red;}`

# Sélecteurs CSS : pseudo-éléments

---

## :first-letter

- ▶ première lettre dans un élément bloc (ex. p, h1, ...)

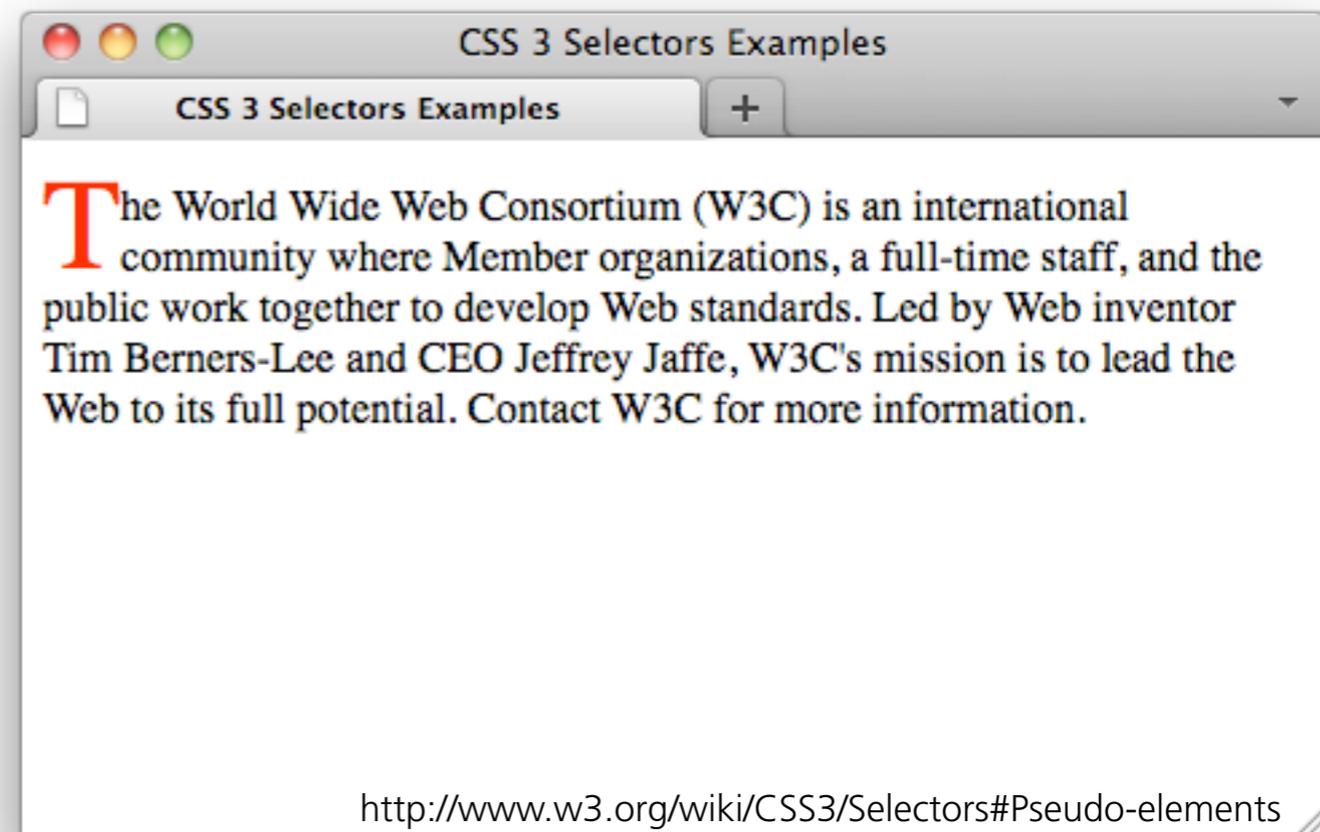
## :first-line

- ▶ première ligne dans un élément bloc (ex. p, h1, ...)

## :after

## :before

- ▶ avant et après du contenu



# Sélecteurs CSS : contextuels

---

Sélecteurs qui ne sélectionnent que des éléments dans un certain contexte

- ▶ Les plus complexes mais aussi les plus riches.

Exemple de style contextuel :

CSS :

```
h1 em { color: red; }
```

HTML :

```
<h1>Ceci est un texte de header  
<em>ce texte est mis en évidence</em>  
celui-ci ne l'est pas.</h1>  
<p>Dans ce paragraphe,  
<em>ceci est mis en évidence</em>  
</p>
```

**Ceci est un texte de header *ce texte est mis en évidence* celui-ci ne l'est pas.**

Dans ce paragraphe, *ceci est mis en évidence*

# Plan

---

- ▶ Introduction
- ▶ HTML
- ▶ HTML5
- ▶ **CSS**
  - ▶ Généralités
  - ▶ Règles et sélecteurs
  - ▶ Propriétés utiles
  - ▶ Placement des styles
  - ▶ Héritage et cascade

# Propriétés de texte

---

- ▶ font-size: **small** | **medium...** | % | **x pt**
- ▶ font-family: **fontname1, fontname2** (*si la première n'est pas disponible*), **familyname** (serif, sans-serif, etc.)
- ▶ font-weight: **bold** | **lighter**
- ▶ font-style: **italic**, **oblique**
- ▶ text-align: **left** | **center** | **right** | **justify**
- ▶ text-indent: (*retrait de première ligne*) % | **2 px**

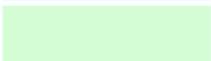
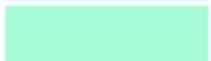
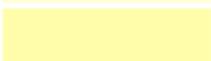
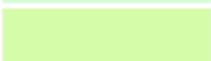
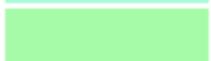
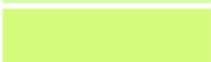
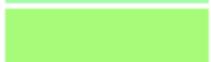
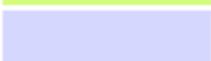
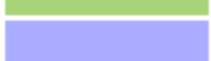
# Les couleurs

color:

▶ red | blue...| hexcode

background-color:

▶ red | blue...| hexcode

Hex Code	Color	Hex Code	Color	Hex Code	Color
#FFFFFF		#CCFFFF		#99FFFF	
#FFFFCC		#CCFFCC		#99FFCC	
#FFFF99		#CCFF99		#99FF99	
#FFFF66		#CCFF66		#99FF66	
#FFCCFF		#CCCCFF		#99CCFF	
#FFCCCC		#CCCCCC		#99CCCC	
#FFCC99		#CCCC99		#99CC99	
#FFCC66		#CCCC66		#99CC66	
#FF99FF		#CC99FF		#9999FF	
#FF99CC		#CC99CC		#9999CC	

Hex Code	Color	Hex Code	Color	Hex Code	Color
#FF9999		#CC9999		#999999	
#FF9966		#CC9966		#999966	
#FF66FF		#CC66FF		#9966FF	
#FF66CC		#CC66CC		#9966CC	
#FF6699		#CC6699		#996699	
#FF6666		#CC6666		#996666	
#FF6633		#CC6633		#996633	
#FF6600		#CC6600		#996600	
#FF33FF		#CC33FF		#9933FF	
#FF33CC		#CC33CC		#9933CC	

# Types d'éléments

---

## Éléments blocs

- ▶ Prennent la forme d'un bloc dans la page
  - ▶ ensemble de lignes
  - ▶ ne peuvent être contenus que dans d'autres éléments blocs
- ▶ Exemple : p, img, ul, table, h1, div...

## Éléments inline

- ▶ S'inscrivent dans la continuité des éléments
  - ▶ ne forcent pas un changement de ligne
  - ▶ peuvent être inclus dans n'importe quel élément
- ▶ Exemple : a, em, span...

## Éléments de listes

- ▶ éléments HTML qui ont un marqueur (ex: bullet) et un ordre

# Rappel sur div et span

---

## div

- ▶ élément contenant d'autres éléments, servant à définir un bloc
- ▶ servira essentiellement à faire du positionnement de bloc

## span

- ▶ élément contenant d'autres éléments, inline
- ▶ servira essentiellement à regrouper des caractéristiques globales pour des éléments textuels

# Propriétés de positionnement

---

S'appliquent aux éléments blocs

Attribut position

- ▶ static

- ▶ bloc placé en fonction de sa position dans l'ordre des balises,
- ▶ ne peut pas être modifiée dynamiquement

- ▶ absolute

- ▶ bloc placé précisément par rapport aux bords de la fenêtre d'affichage
- ▶ la position peut être modifiée dynamiquement (déplacement)

- ▶ relative

- ▶ bloc positionné par rapport à sa position normale, dans le flux (ex. décalage de 20 pts à droite)

# Propriétés graphiques : arrière plan

---

## Arrière-plan

- ▶ background-color (couleur de fond)
- ▶ background-image (image de fond)
- ▶ background-repeat (types de répétition de l'image)
- ▶ background-attachment (arrière-plan fixe quand on utilise l'ascenseur)
- ▶ background-position (position par rapport au coin supérieur gauche)
- ▶ background (attribut de résumé)

Exemple : `p { background: image.gif fixed repeat-y }`

# Propriétés graphiques : visibilité

---

## Attribut visibility et display

- ▶ indiquent si (visible|hidden) et comment un élément est affiché

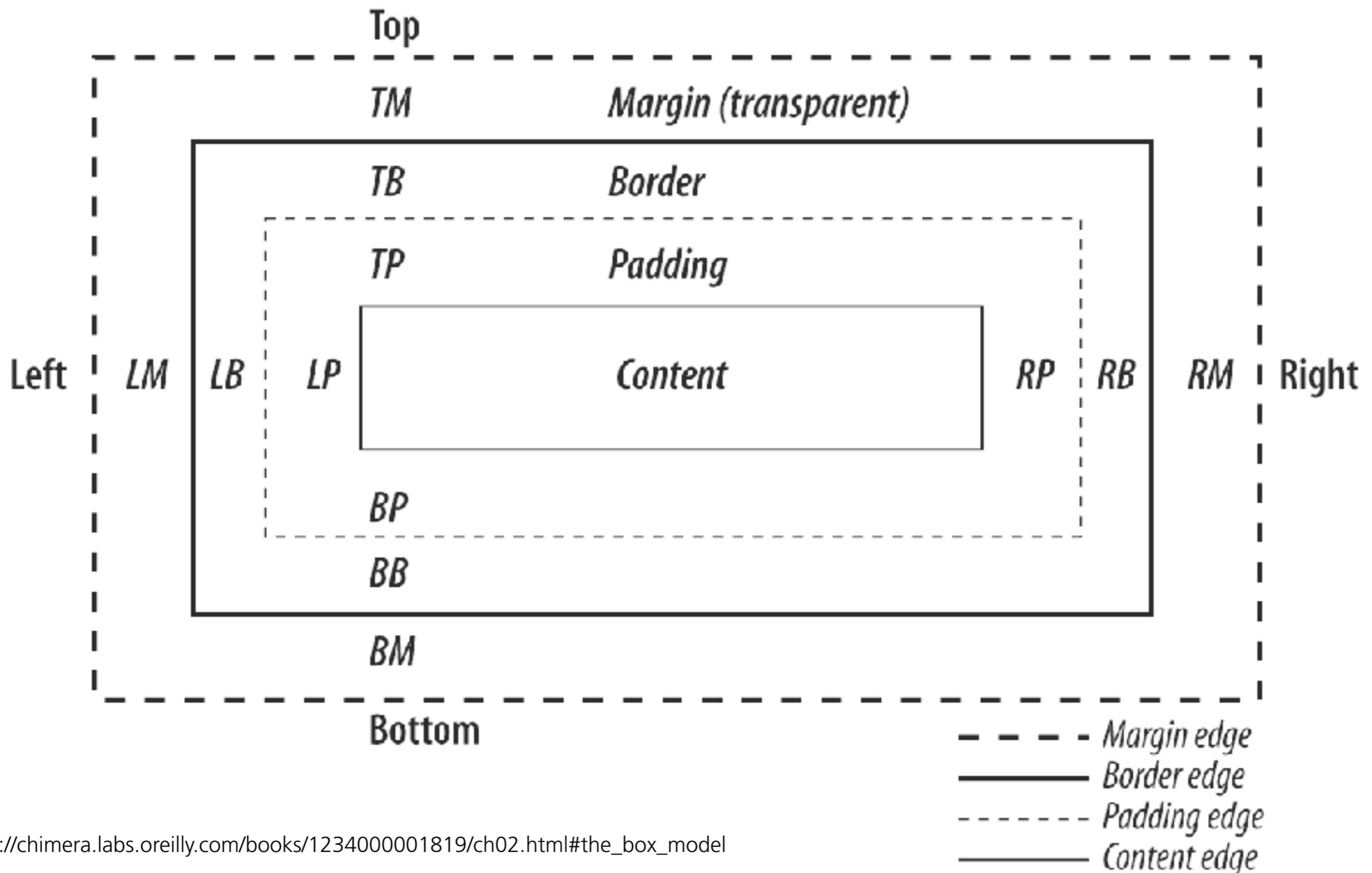
## Attribut z-index

- ▶ en cas de superposition de blocs d'affichage, indique l'ordre dans lesquels le navigateur doit les afficher (cf. couches/layers dans les logiciels de dessin)

## Remarques

- ▶ les distances s'expriment en pixels (px), points (pt), unités métriques (cm, mm), ou pourcentages de la taille de la fenêtre (%) ou en **unité relative (em, rem)**

# Le modèle de boîte (box model)



# Plan

---

- ▶ Introduction
- ▶ HTML
- ▶ HTML5
- ▶ **CSS**
  - ▶ Généralités
  - ▶ Règles et sélecteurs
  - ▶ Propriétés utiles
  - ▶ Placement des styles
  - ▶ Héritage et cascade

# Styles intégrés

---

Déclarés comme attribut style d'un élément

Exemple

```
<h1 style="font-family: Arial; font-style: italic;">
```

```
  Un texte qui se retrouvera en arial italique
```

```
</h1>
```

# Styles intégrés

---

- ▶ La feuille de style est déclarée dans l'en-tête (head) du document XHTML
- ▶ Elle s'appliquera aux éléments dans body
- ▶ Exemple

```
<html>
<head>
  <title>Texte du titre</title>
  <style type="text/css">
    .important {color: red;}
    h1 {font-family: Arial; font-style: italic; }
    em { color: green }
  </style>
</head>
<body>... </body>
</html>
```

# Feuilles de styles liées

---

Les règles se trouvent dans un fichier extérieur

On indique au navigateur où se trouve cette ressource de style, en utilisant un élément link dans l'en-tête <head>

- ▶ `<link rel="stylesheet" type="text/css" href="fichier-de-style.css" >`

Intérêt

- ▶ Plusieurs documents XHTML peuvent faire appel à la même feuille de style

# Plan

---

- ▶ Introduction
- ▶ HTML
- ▶ HTML5
- ▶ **CSS**
  - ▶ Généralités
  - ▶ Règles et sélecteurs
  - ▶ Propriétés utiles
  - ▶ Placement des styles
  - ▶ Héritage et cascade

# Héritage de styles

---

## ▶ Par défaut

- ▶ Les styles d'un élément sont héritées par ses éléments descendants

### ▶ Exemple :

```
<p style="color: red;"> Du texte <em>mis en évidence</em>, pas mis en évidence.</p>
```

**Du texte *mis en évidence*, pas mis en évidence.**

## ▶ A condition que l'héritage ait un sens

- ▶ i.e. que les caractéristiques soient applicables à l'élément enfant

- ▶ un positionnement de bloc n'a pas d'intérêt pour un élément em qui y est contenu

- ▶ si un style est défini spécialement pour un élément

- ▶ ex: em { color: blue; }, l'héritage ne se fait pas

**Du texte *mis en évidence*, pas mis en évidence.**

# Cascading style sheets : cascade

---

On peut avoir concurrence entre plusieurs styles définis dans de multiples endroits

- ▶ styles par défaut (1- navigateur)
- ▶ fichiers CSS externes (2- spécifications globales au site)
- ▶ élément head du document XHTML (3- spécification globales au doc.)
- ▶ attributs style des éléments (4- spécification locales)
- ▶ style utilisateur (5- spécification de l'utilisateur)

# Cascading style sheets : priorités

---

## Notion de cascade ou ordre de priorité des styles

- ▶ trouver toutes les déclarations qui s'appliquent à un élément
- ▶ les classer par spécificité
  - ▶ système de poids qui s'ajoutent
- ▶ les classer par ordre d'apparence
  - ▶ plus une déclaration apparaît tard, plus elle a de poids

## Exemple

- ▶ style (4) > style (4 hérité) > style (3) > style (2) > style (1)