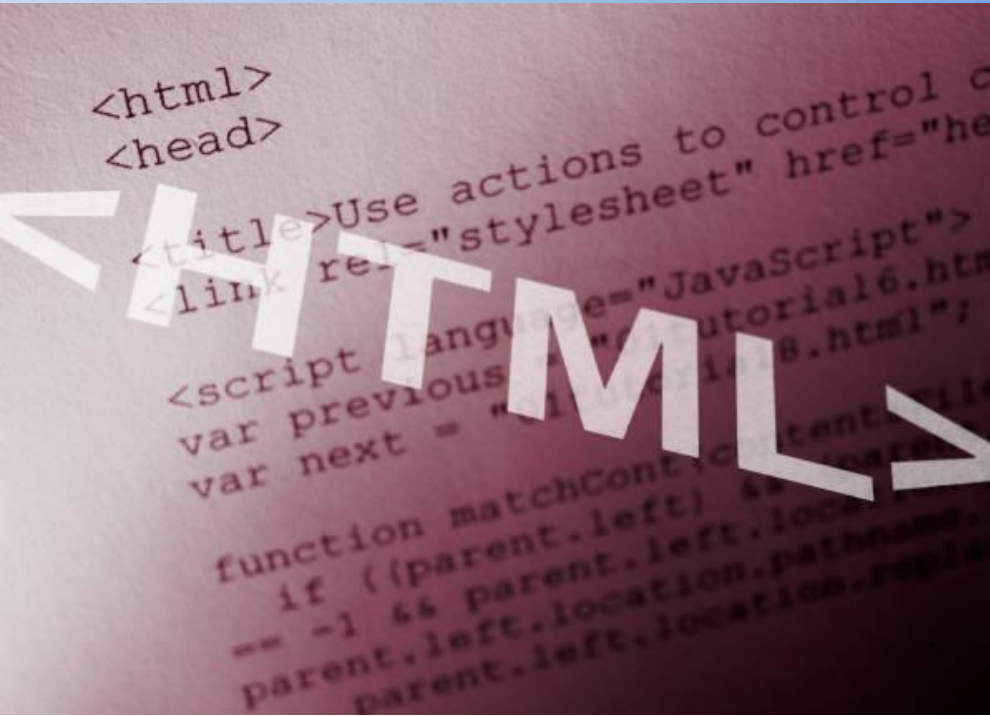


MODULE

CONCEPTION DE DOCUMENTS WEB



1. Langages pour l'IHM
2. HTML, CSS
3. Javascript
4. JQuery

C'est un module... HTML + CSS + Javascript + JQuery

**TOUT AUTRE langage, bibliothèque, API, utilitaire...
... est interdit !**

PAS DE PHP : trouvez les solutions en HTML ou en JS !

**PAS d'AVANCE DE PHASE : si en cours on n'a vu que
HTML, la réponse au TP n'est PAS en Javascript : il
faut la trouver en HTML**

HTML permet de définir une IHM.....

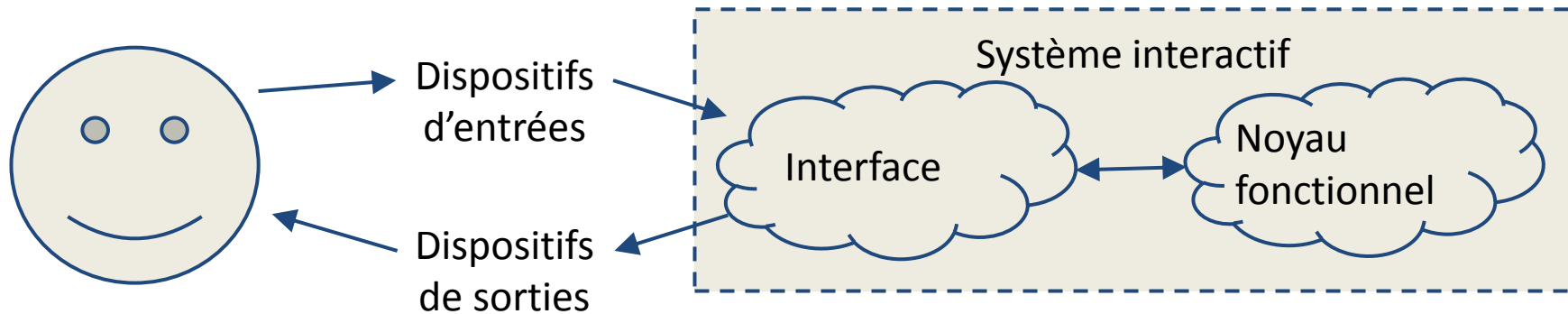
IHM?

Interaction **H**umain - **M**achine

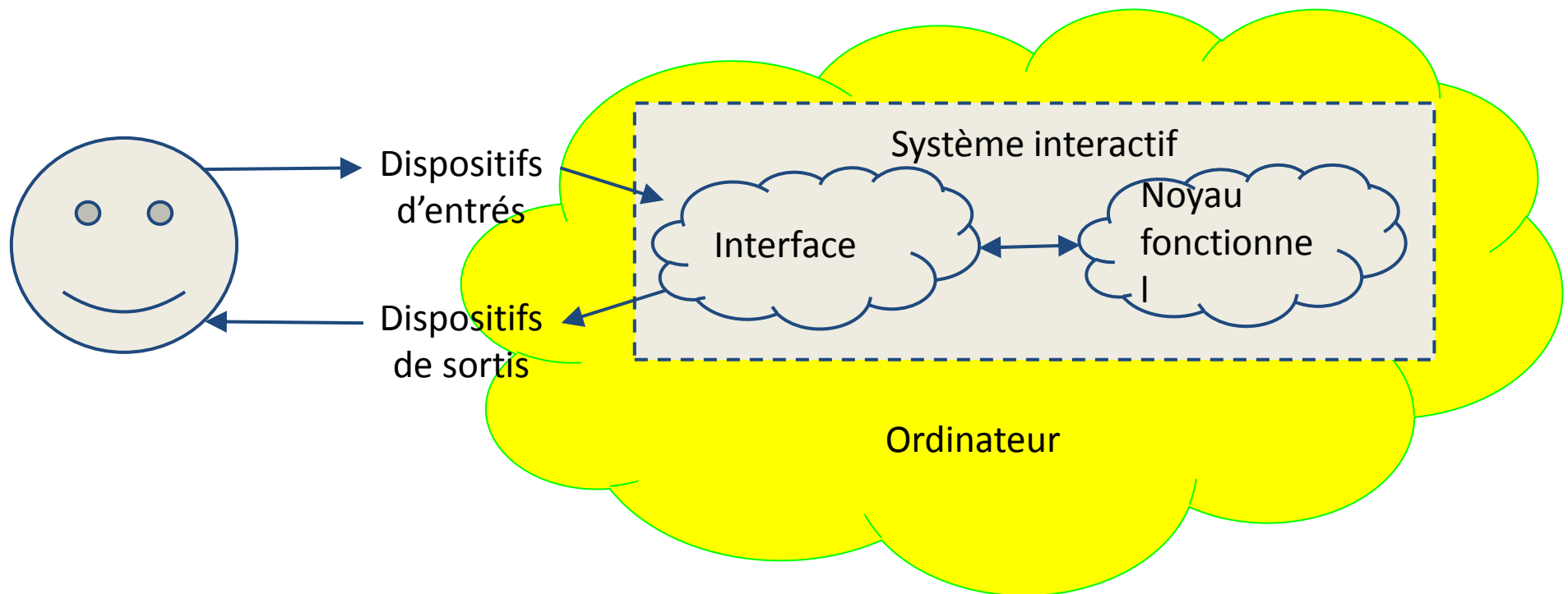
Décrire l'interaction entre une machine et ses utilisateurs :

- Conducteur - Bus
- Ouvrier - Machine
- ...
- Personnes - Ordinateur

Modifier / Agir sur l'état du système



Percevoir / Comprendre l'état du système



Une interface est une zone, réelle ou virtuelle qui sépare deux éléments.

L'interface désigne ainsi ce que chaque élément a besoin de **connaître** de l'autre **pour** pouvoir **fonctionner correctement**.

Types d'interactions

CLI : Command Line Interface

- + Flexible
- + adapté aux experts
- + très peu couteux (réseaux...)
- Mémorisation
- Apprentissage
- Gestion des erreurs
- Rebutte les non experts

```
FCS transmogriifier
Version: 1.0b Build: 2011042001
Framework Version: 1.0b Build: 2011042001

Copyright (C) 2009-2011 Beau Hunter, 318 Inc.

Usage:
transmogriifier.py [options] [target]
transmogriifier.py [-f configfile] [-d supportdir] [-a action] [-t mediatitle]

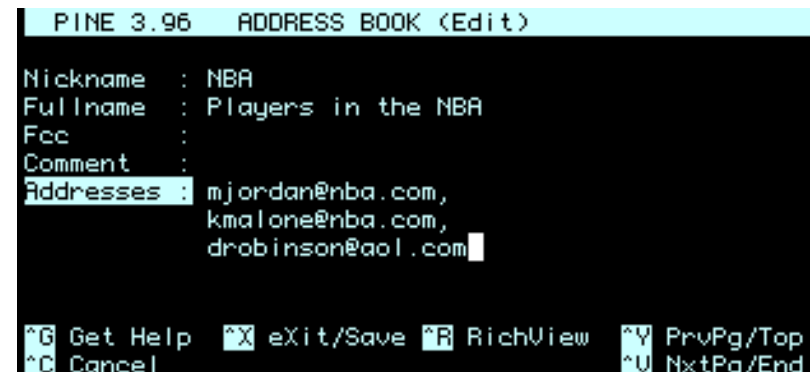
Working with assets:
transmogriifier.py --setField="Keywords" --value="MyAsset" -t MyAsset
transmogriifier.py --setField="Keywords" --value="MyAsset" -i /fcxmlinfile.xml
transmogriifier.py --appendField="Status" --value="Update!" -i /fcxmlinfile.xml

transmogriifier.py --getAssetID --assetPath="/FCS/Media/myfile.mpg"
transmogriifier.py --getEntityPath --assetID=1
transmogriifier.py --getFilePath --assetID=1
transmogriifier.py --getArchiveFilePath --assetID=1
transmogriifier.py --getThumbnailPath --assetID=1
transmogriifier.py --getProxyPath --assetID=1
transmogriifier.py --getPosterFramePath --assetID=1
transmogriifier.py --archive --assetID=1
transmogriifier.py --restore --assetID=1
transmogriifier.py --analyze --assetID=1
```

Types d'interactions

Formulaires

- + Simplifie la saisie
- + Guide
- Occupe de la place
- Rigide



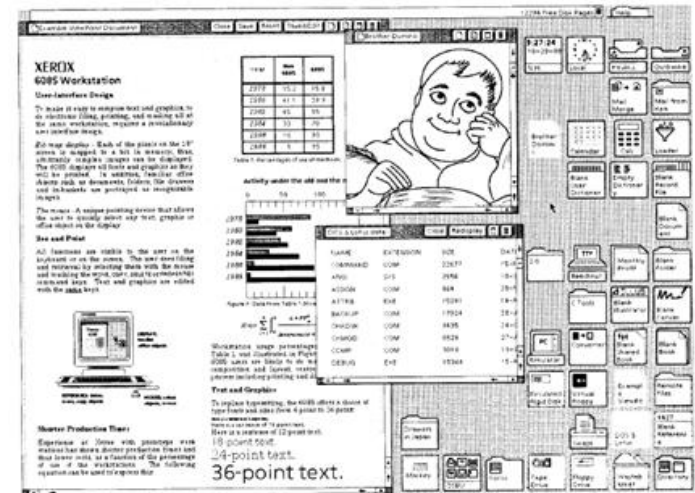
```
PINE 3.96 ADDRESS BOOK (Edit)
Nickname : NBA
Fullname : Players in the NBA
Fcc :
Comment :
Addresses : mjordan@nba.com,
            kmalone@nba.com,
            drobinson@aol.com
^G Get Help ^X eXit/Save ^R RichView ^Y PrevPg/Top
^C Cancel ^U NxtPg/End
```

Notion de widget, d'interacteur...

Types d'interactions

WIMP : Window Icon Menu Pointer

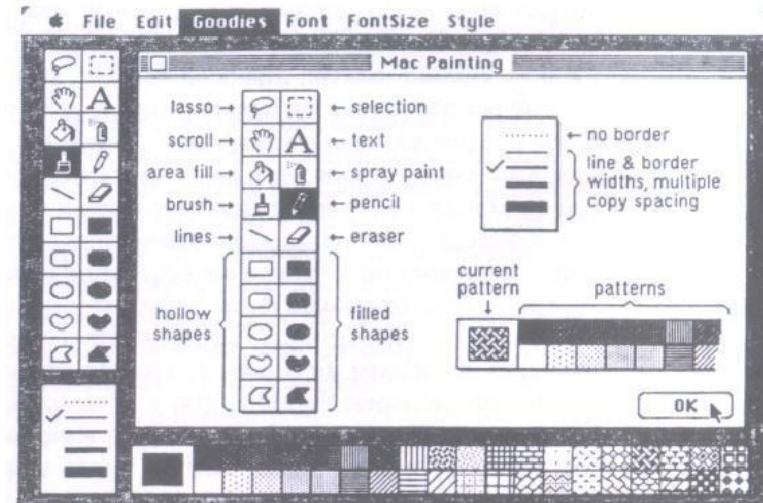
- + Fenêtrage, plusieurs écrans en un
- + Icones comme raccourcis
- + Menu, structuration
- + Pointeur
 - * précision
 - * indique où se porte l'attention
- + Standardisation
- Trop spécialisé au contexte PC
- Indirections (souris, manip...)



Types d'interactions

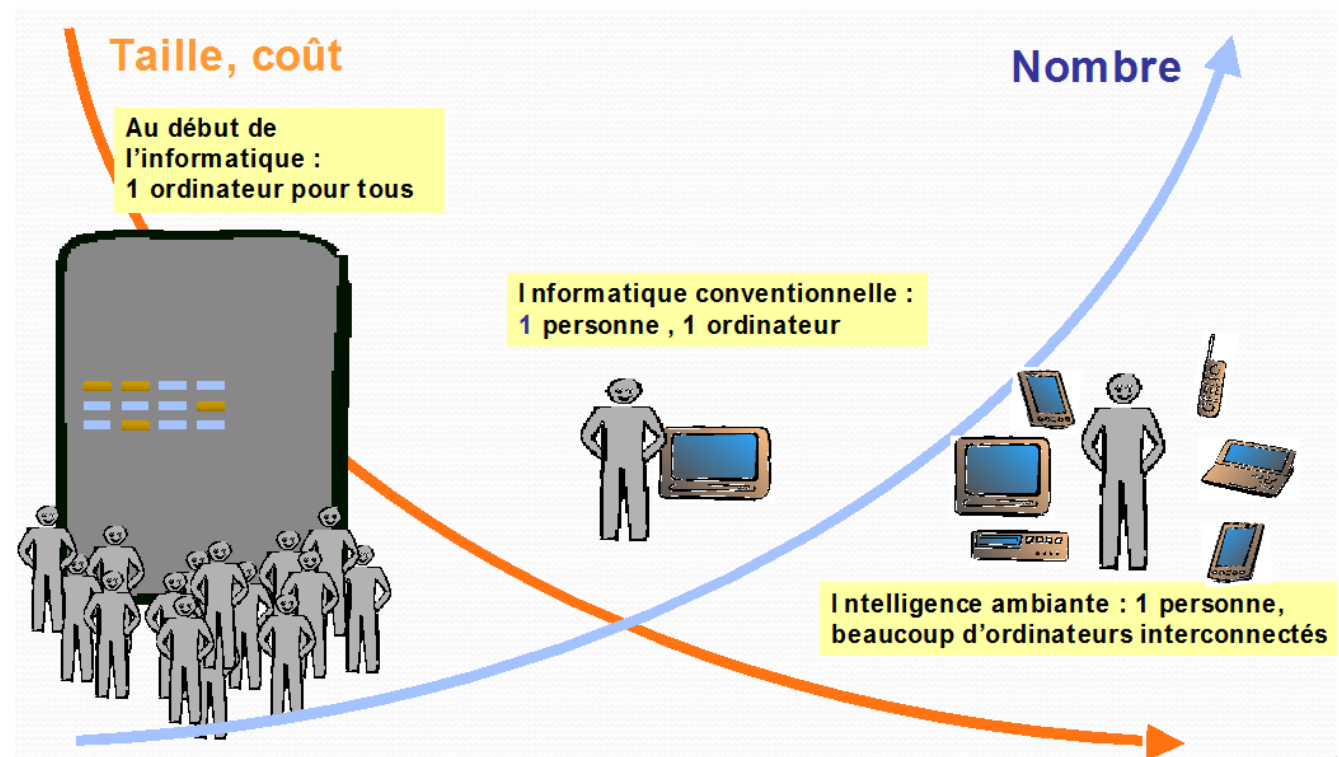
Manipulation directe

- + Représentation des concepts
- + Utilisation de métaphores
- + Apprentissage, exploration
- + Réduction des erreurs
- Représentation parfois non pertinente
- Métaphores parfois trompeuses

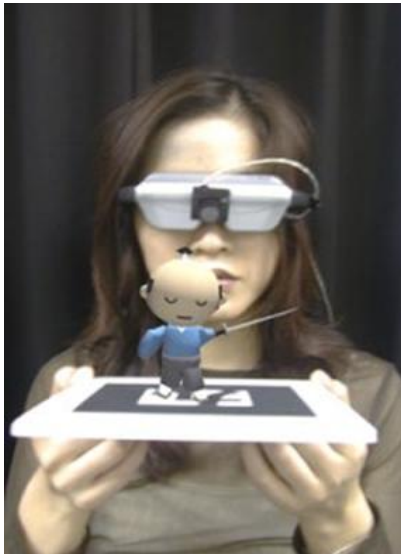


Types d'interactions Post-WIMP

- Mobile
- Gestuelle
- Ambiante
- ...



Types d'interactions



Réalité augmentée



Tableau blanc interactif



Interface tangible

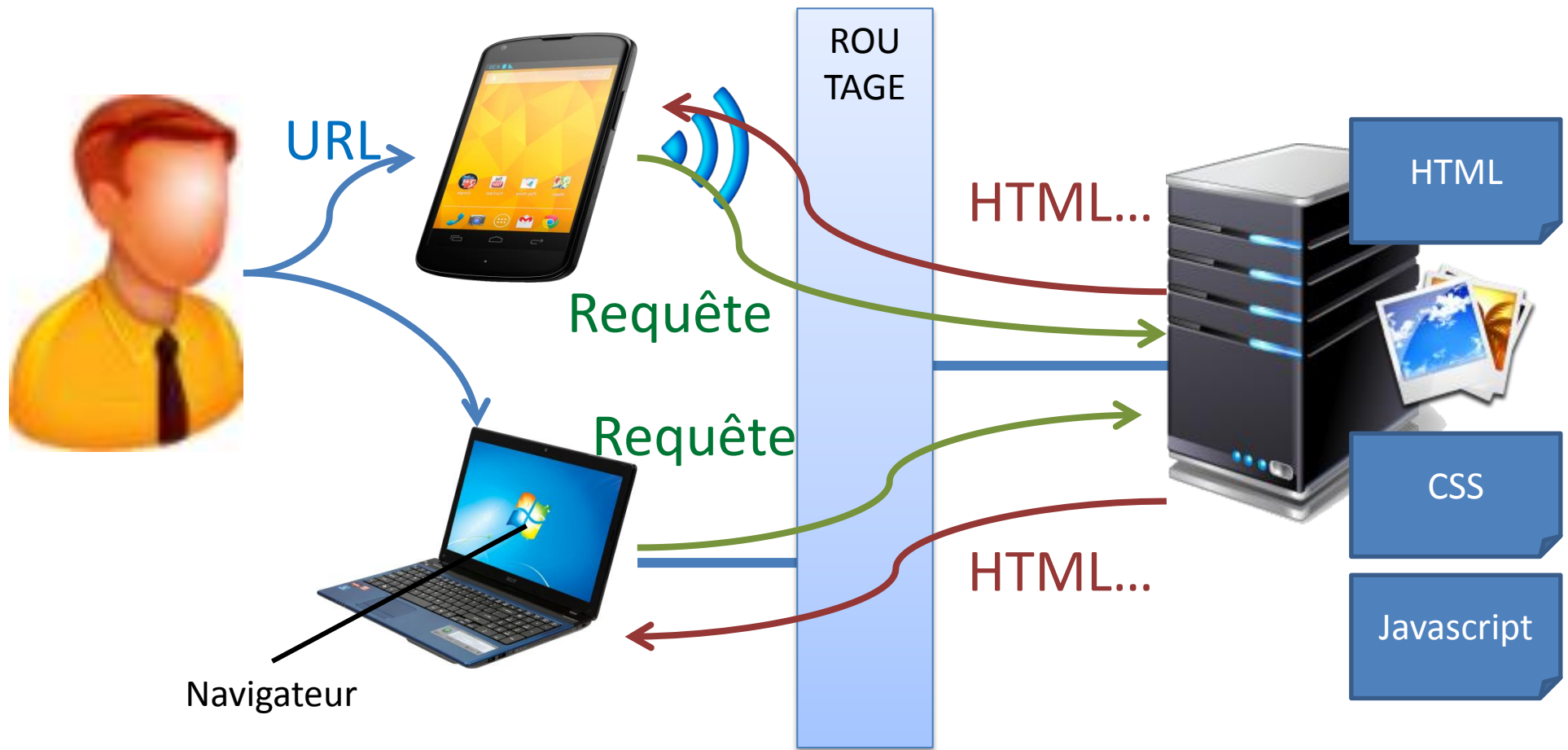
Quel que soit le type d'interaction...

Réaliser des logiciels utiles et utilisables

- Adéquation entre fonctionnalités proposées et besoins des utilisateurs
- Adéquation entre l'interface et les utilisateurs

... EN HTML/CSS/Javascript/Jquery : AUSSI !

HTML : schéma de principe



HTML

- Présentation
- Les balises, les attributs
- Structure d'un document
- Typologie des éléments
- Les chemins
- Bonnes pratiques

Lectures recommandées

- *Premiers pas en CSS3 et HTML5*, de F. Draillard
- *Introduction à HTML5*, de B. Lawson, R. Sharp
- *HTML 5 pour les Web Designers*, de J. Keith
- *Responsive Web Design avec HTML5 et CSS3*, de B. Frain

Une page web c'est...

The screenshot shows the Open Source Initiative (OSI) website. A red oval highlights the navigation menu on the left, which includes links like 'About the OSI', 'The Open Source Definition', 'Open Source Licenses', 'Trademark and Logo Usage', 'FAQ', 'Open Standards', 'Open Source Education', 'Mailing lists', 'Getting Help', 'Donate to the OSI', 'Terms of Service', 'OSI Affiliate Membership', 'Contact OSI', and 'OSI Individual Membership'. A red oval also highlights the top green header bar. A blue oval highlights the main content area, which includes the 'Mission' section, 'OSI Opening For Membership' section, and a 'JOIN OSI NOW!' button. A blue oval highlights the 'Recent blog posts' section on the right, which lists several articles. A blue oval highlights the 'Du Contenu' (Content) section at the bottom, which features a list of logos for various open source projects and a 'Subscribe to blog' button.

Mission | Open Source Initiative

De la mise en page

Search this site: Search

Navigation

- ▶ About the OSI
- ▶ The Open Source Definition
- ▶ Open Source Licenses
- ▶ Trademark and Logo Usage
- FAQ
- ▶ Open Standards
- Open Source Education
- ▶ Mailing lists
- ▶ Getting Help
- ▶ Donate to the OSI
- Terms of Service
- ▶ OSI Affiliate Membership
- Contact OSI
- OSI Individual Membership

Mission

The Open Source Initiative (OSI) is a non-profit corporation with global scope formed to educate about and advocate for the benefits of open source and to build bridges among different constituencies in the open source community.

Open source is a development method for software that harnesses the power of distributed peer review and transparency of process. The promise of open source is better quality, higher reliability, more flexibility, lower cost, and an end to predatory vendor lock-in.

One of our most important activities is as a standards body, maintaining [the Open Source Definition](#) for the good of the community. The Open Source Initiative Approved License trademark and program creates a nexus of trust around which developers, users, corporations and governments can organize open source cooperation.

OSI Opening For Membership

The OSI Board is in the process of progressing governance reform to transform OSI into a member-based organisation.

JOIN OSI NOW!

September 6, 2012: **Five New Affiliates** join from around the world. [Read more...](#)

July 18, 2012: **Individual Membership**: The next step of OSI's transformation into a member organization starts today! You can become an Individual Member.

June 18, 2012: **Affiliate Membership**: OSI is now accepting applications for Affiliate Membership. Additionally, OSI invites you to become an Affiliate.

Recent blog posts

- [OSI Welcomes Five More Affiliates](#)
- [OSI Signs Declaration of Internet Freedom](#)
- [Open Source on G+](#)
- [OSI Welcomes New Affiliates, Opens For Affiliate Applications](#)
- [OSI Board Elects New Officers](#)
- [Survey Results Re Individual Memberships](#)
- [FRAND and Open Standards](#)
- [Thank you, Michael Tiemann!](#)
- [OSI Welcomes Debian and CENATIC](#)
- [OSI's new Board](#)

[more](#)
Subscribe to blog

Du Contenu

document

OSI Affiliates, July 18, 2012

Logos for various open source projects: Apache, CC, Debian, Docker, Fedora, Inkscape, Linux, MySQL, OpenOffice, PostgreSQL, Ubuntu, etc.

zotero

Le contenu : texte, images, liens, tableaux... que le visiteur va voir concrètement. Le créateur de la page indique librement le contenu qu'il souhaite.

La mise en page : ce qui indique au navigateur comment présenter le contenu. Pour cela on utilise des langages de présentation : HTML, XHTML, HTML5,...

Une page, c'est donc un (des) fichier(s) qui décrivent la structure du document Web et le contenu à afficher.

Extension **.html** ou htm

Fichiers au **format texte** permettant la description de **documents Web**

Sous l'autorité du **W3C**
World Wide Web Consortium
<http://www.w3c.org>

Langages non propriétaires

HTML : HyperText Markup Language

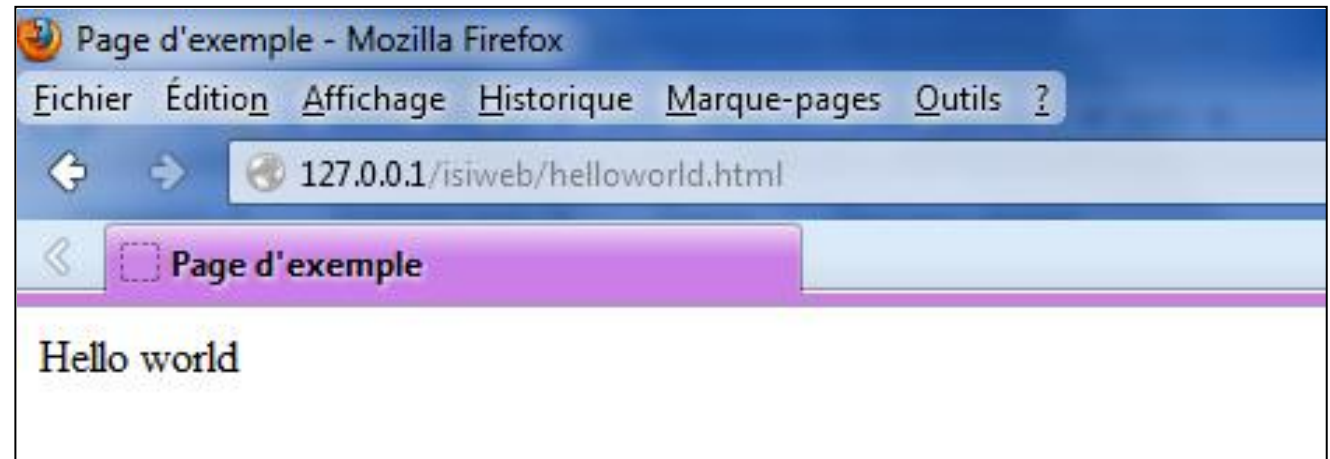
Structuration de documents dédiés aux sites Web

XHTML : eXtensible HyperText Markup Language

- Equivalent à HTML
- Mais reformulation de HTML en **XML**
- Ecriture plus rigoureuse

HTML : exemple

```
<html>  
  <head>  
    <meta charset="ISO-8859-1">  
    <title>Page d'exemple</title>  
  </head>  
  <body>  
    <p>Hello world</p>  
  </body>  
</html>
```



HTML : structure générale

```
<html>  
  <head>  
    <meta charset="ISO-8859-1"/>  
    <title>Page d'exemple</title>  
  </head>  
  <body>  
    <p>Hello world</p>  
  </body>  
</html>
```

1- Des codes entre
< et > : les **balises** !

HTML : structure générale

```
<html>  
  <head>  
    <meta charset="UTF-8">  
    <title>Page d'exemple</title>  
  </head>  
  <body>  
    <p>Hello world</p>  
  </body>  
</html>
```

2- Du contenu

Langage de balisage composé de **balises (ou marqueurs ou tags)**

- Syntaxe :

<mot_cle>*texte***</mot_cle>**

Ou **<mot_cle/>**

- Les balises sont prédéfinies par le langage
- Structurer les documents : définir les zones et les widgets de la page

Le chevauchement des balises est illégal

Valide :



`<h1>Texte<i>sans chevauchement</i></h1>`

Invalide :



~~`<h1>Texte<i>avec chevauchement</h1></i>`~~

Les attributs : permettent de préciser une valeur dans les balises ouvrantes ou uniques, structure en paires : **nom**="valeur"

<p id="unique">Paragraphe unique**</p>**

<p class="special">Paragraphe spécial**</p>**

Apostrophes doubles (le plus courant) ou simples optionnelles mais (fortement) recommandé

Attributs standards : **id**="id432" **ou** **class**="typeUrgent"

Attention :

- **XHTML : case sensitive ; HTML5 : non**
- **Recommandation : noms d'attributs en minuscules**

Certains attributs sont obligatoires sur certaines balises:

``

~~``~~

Les attributs standards peuvent avoir des valeurs contraintes :

```
<input type="checkbox" name="rouge" value="ok"  
checked="checked"/>
```

```
<input type="map" name="grenoble" value="grenoble"  
checked="known"/>
```

Un attribut standard ne peut être utilisé que sur les balises correspondantes :

``

~~`<p src="url_de_l_image"/>`~~

Il est possible d'ajouter des attributs non prévus dans le langage, en faisant précéder leur nom de "data-"

`<p data-couleurDuCiel="bleu">texte</p>`

est valide.

`<p data-couleurDuCiel="bleu">texte</p>`

Nota : "data-" non requis en HTML4, son absence est supportée par les navigateurs, mais non conforme au langage HTML5.

Rappel : couleurDuCiel = couleurduciel
couleurDuCiel = non recommandé, mais + facile

<!-- Un commentaire -->

*Non affiché par les navigateurs
Mais comme tout le reste du document
lisible en affichant la source*

On ne met pas de tiret double
à l'intérieur d'un commentaire

<!-- Un commentaire -->

<!--=====-->

sont valides.

~~**<!-- Un autre -- commentaire -->**~~

~~**<!--=====-->**~~

ne sont pas valides.

HTML – Structure des documents

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="UTF-8">
    <title>Page d'exemple</title>
  </head>
  <body>
    <p>Hello world</p>
  </body>
</html>
```

2- Des blocs : le document **Html** qui contient un **en-tête** et un **corps**

HTML – Structure des documents

L'imbrications des éléments

```
<!doctype html>
```

```
<html lang="fr">
```

```
<head>
```

```
<meta charset="l
```

```
<title>Page d'ex
```

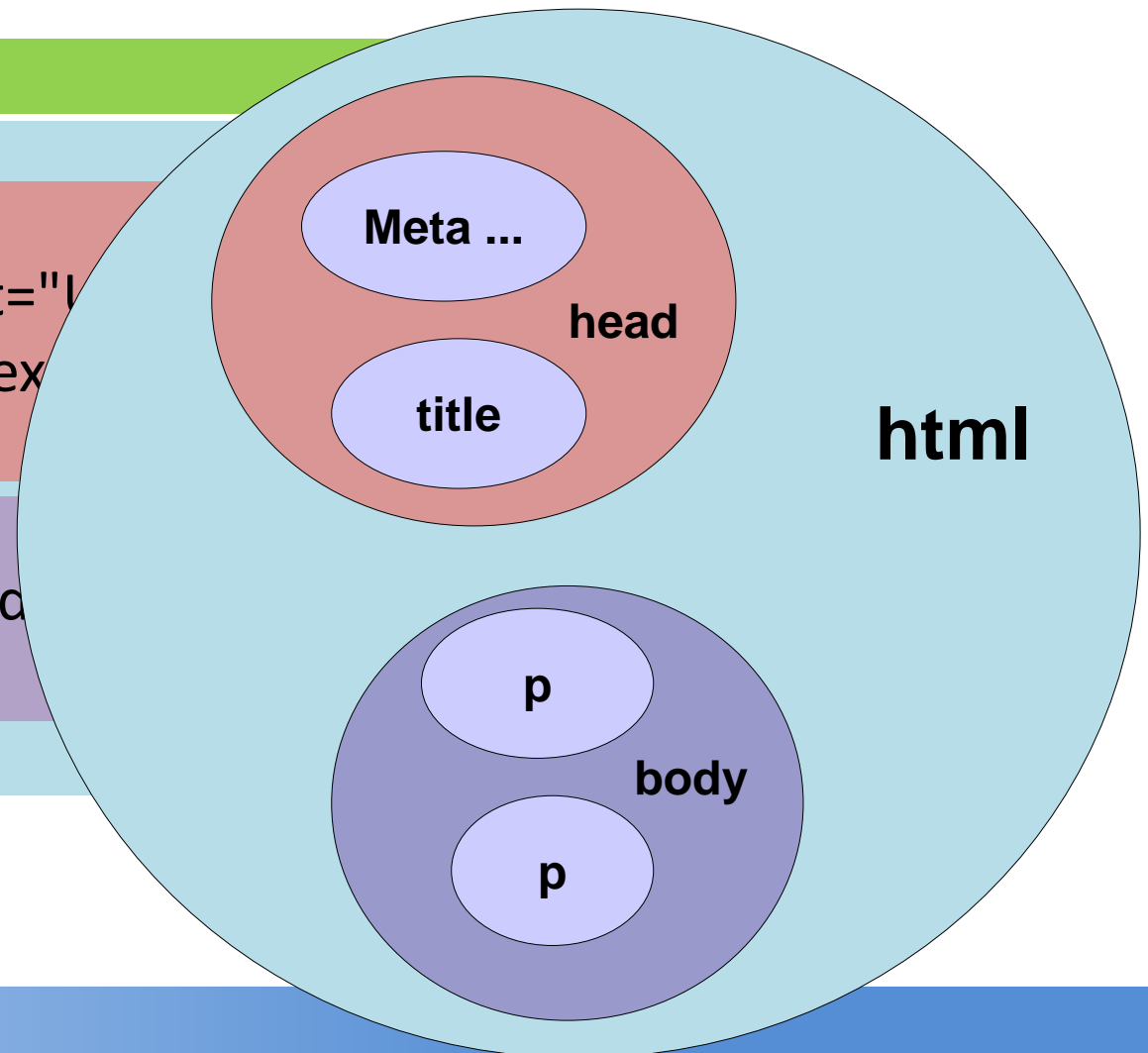
```
</head>
```

```
<body>
```

```
<p>Hello world
```

```
</body>
```

```
</html>
```



Un document HTML comprend plusieurs parties

- Un doctype
- Un élément racine html
 - Un en-tête
 - Un corps

HTML – Structure des documents

- doctype
- html
 - Un en-tête
 - Un corps

```
<!doctype html >  
<html>  
    <head lang="fr">  
    </head>  
    <body>  
    </body>  
</html>
```

HTML – Structure des documents

Déclaration du type de document

Le tag doctype indique au navigateur dans quel type de HTML la page a été écrite (HTML-3.2 «classique», HTML-4 de transition ou strict, XHTML, etc...).

`<!doctype html>` signifie : document en HTML5.

Utile pour les navigateurs, les lecteurs de votre code ou les validateurs de code

HTML – Structure des documents

Langue

La déclaration de la langue se fait dans la balise html.

```
<html lang="fr">
```

Nous déclarons ici que la langue est le français, ce qui sert au référencement (ex : Google vous propose les pages en français ou une traduction) et aux synthétiseurs vocaux.

- Délimité par les balises **<head>** et **</head>**
- Contient des informations non affichées par les navigateurs courants
 - Le titre
 - Les méta-données
 - Les références à d'autres ressources
 - Le type d'encodage des caractères

L'en-tête : le titre

<title>Mon document</title>

Le titre n'est pas directement affiché dans le document, mais souvent par le navigateur, dans l'interface (nom de l'onglet, par exemple)

HTML – Structure des documents

L'en-tête : les méta-données

Ce sont :

- **mots clés et description** : décrire le contenu du document et faciliter le référencement
- **informations / instructions aux moteurs de recherche** : indiquent s'il faut suivre les liens lors de l'indexation d'un site, la fréquence de relecture souhaitée
- D'autres méta données permettent de préciser **le nom de l'auteur, la version...**

HTML – Structure des documents

L'en-tête : les méta-données

Exemples :

- `<meta name="description" content="Cours SIL-2, IUT2 de Grenoble"/>`
- `<meta name="author" content="Pierre Dupont">`
- `<meta name="keywords" content="sil2, html, css">`
- `<meta name="robots" content="[[no]index][no[flollow]]"/>`

L'en-tête : encodage

Le type d'encodage des caractères de la page

<meta charset="encodage"/>

ATTENTION : doit être cohérent avec l'encodage physique de la page.

Pour le français on dispose des encodages suivants :

iso-8859-1 : encodage classique pour les langues de l'Europe occidentale (aussi appelé Latin-1)

iso-8859-15 : même encodage comportant quelques caractères supplémentaires comme le signe €...

utf-8 : encodage pour les caractères de la majorité des langues mondiales

A noter : **L'en-tête : encodage**

En utilisant l'encodage iso-8859-1 ou iso-8859-15 les caractères ASCII 7-BIT (codes 32 à 127) sont valides, avec 4 exceptions car ce sont des caractères du langage XHTML.

Ces exceptions sont codées avec des entités :

" (codé **"**;))

& (codé **&**;))

< (codé **<**;))

> (codé **>**;))

L'en-tête : encodage

En utilisant l'encodage iso-8859-1 ou iso-8859-15 :

Les caractères, en dehors de la classification ASCII 7-BIT (donc les codes de 128 à 255), sont codés par **des entités ou des références numériques** :

Par exemple :

é (codé **é** ou **é**;)

è (codé **è** ou **è**;)

à (codé **à** ou **à**;)

ô (codé **ô** ou **,**;)

etc.

Remarque :
le blanc
insécable s'écrit
** **;

L'en-tête : les références

Les références permettent de... référencer des ressources utilisées par le document : feuilles de style CSS, icône, fichiers de scripts externes Javascript...

Exemples :

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

```
<link rel="shortcut icon" type="image/x-icon" href="favicon.ico" />
```


HTML – Structure des documents

Le corps du document

Le corps d'un document est délimité par les balises `<body>` et `</body>`. Il peut contenir divers éléments :

- du texte (titres, paragraphes, listes, etc.)
- des images
- des hyperliens
- des tableaux
- des formulaires
- des cadres
- des objets externes (applets Java, Flash ...)
- etc.

HTML – Typologie des éléments

Structure et rendu

Chaque élément a une double identité :

- Sa **structure** (mot clé, attributs standards, ...)
- Son **rendu** (ou apparence), qui est défini par défaut pour chaque navigateur (et qui peut être modifié par CSS).

HTML – Typologie des éléments

Block et inline

2 grands mode de rendu des éléments

Les rendus de type **block**

Les rendus de type **inline**

Cette typologie dicte le comportement en terme de **positionnement** et d'**affichage**

HTML & XHTML – Typologie des éléments

Les éléments de type Block

Ce sont :

- Des blocs dans les documents – Exemples : paragraphes, listes...
 - Apparaissent les **uns en dessous des autres**
 - Ont des **dimensions et des marges externes ou internes fixées** par défaut, à l'exception des blocs **<div>**
 - Sont **positionnables** (avec les feuilles de style CSS)

HTML & XHTML – Typologie des éléments

Les éléments de type Block

Ils peuvent :

- **contenir d'autres blocs**
sauf les blocs de paragraphes (<p>) et de **titres** (<h1>, <h2>, ... <h6>) **qui ne peuvent contenir d'autres blocs**
- **Contenir des éléments inline**

HTML & XHTML – Typologie des éléments

Les éléments de type Block

Exemples

- `<h1></h1>`
- `<h2></h2>`
- ...
- `<h6></h6>`
- `<p></p>`
- `<table></table>`
- ``
- ``
- `<blockquote></blockquote>`
- `<dl></dl>`
- `<div></div>`
- etc.

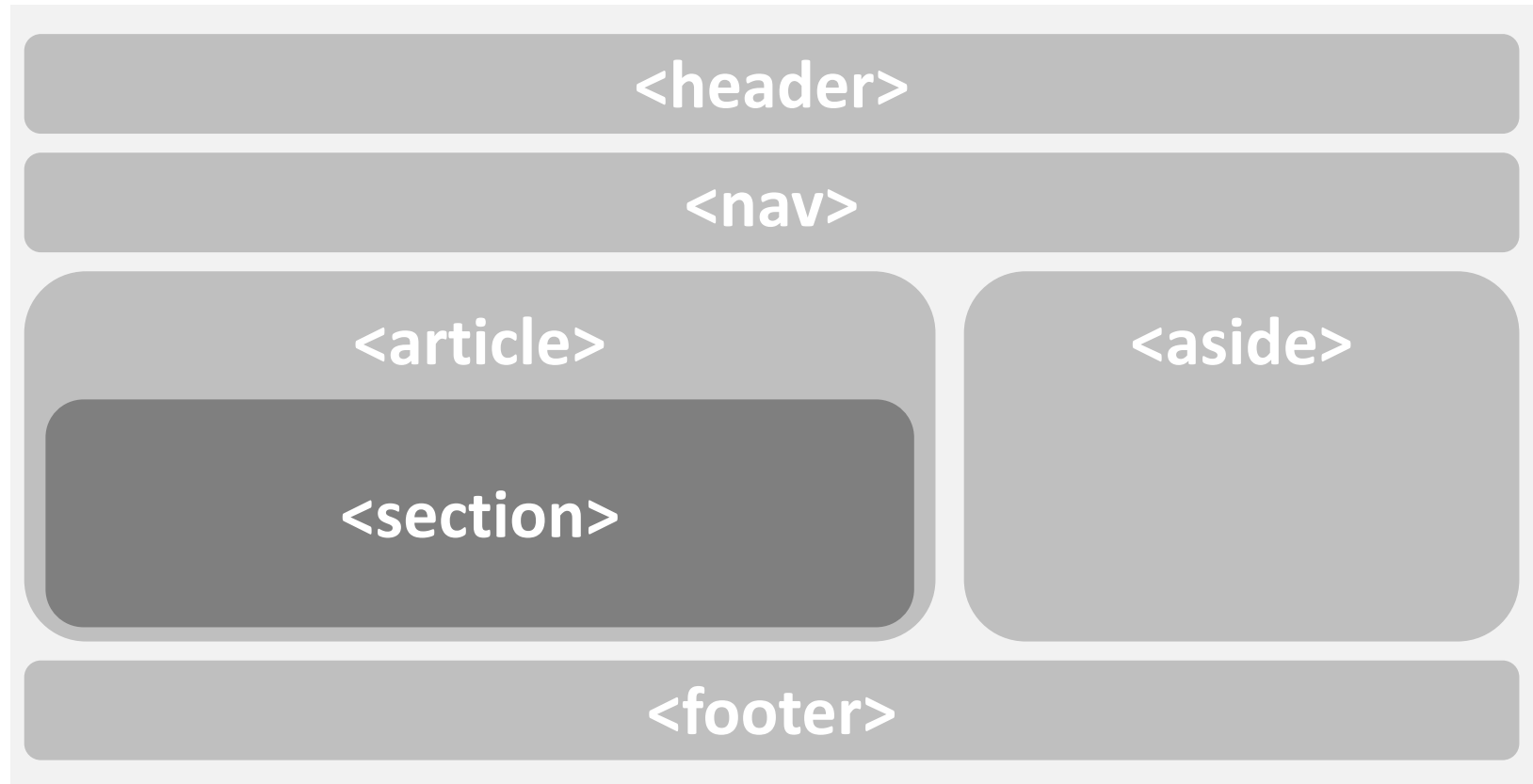
Éléments de type block pour balisage sémantique

N.B. :
proximité
des notions
d'article et de
section.

Section : regrouper
du contenu
apparenté

Article : contenu
apparenté
autonome (?)

Aside : contenu indirectement apparenté

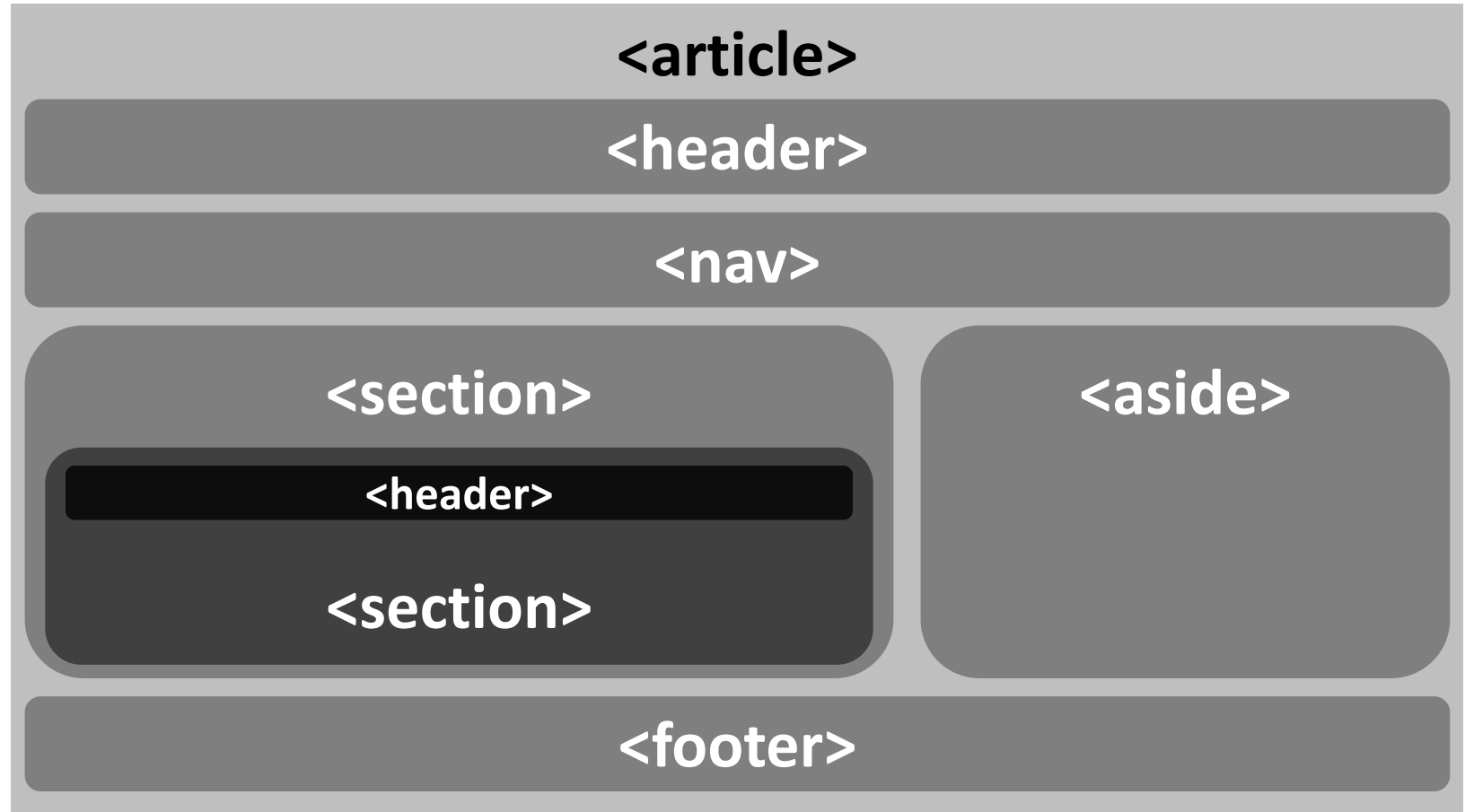


HTML – Structure des documents

Imbrication des éléments

Sections et articles peuvent contenir les autres éléments, y compris des sous-sections et des sous-articles

==> définir la sémantique de ces éléments au regard du contenu du site.



HTML – Structure des documents

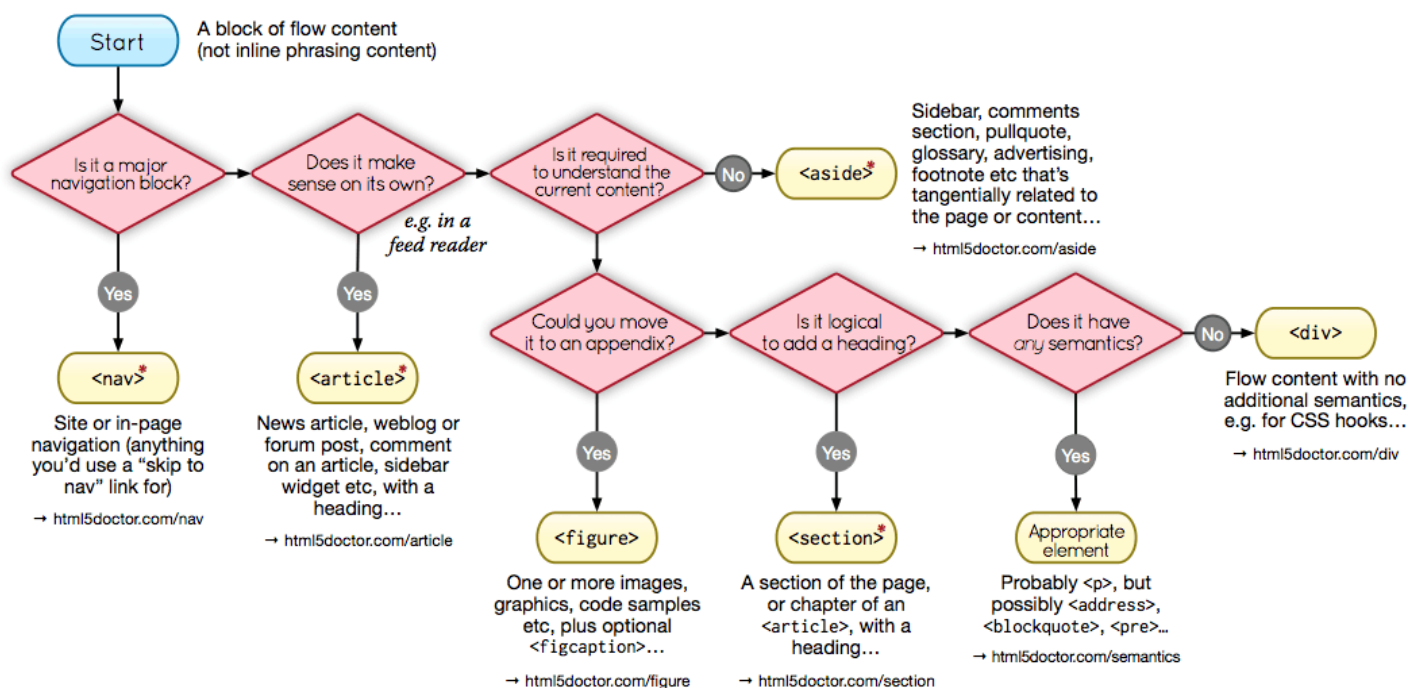
Éléments de type block pour balisage sémantique



HTML5 Element Flowchart

Sectioning content elements and friends

By @riddle & @boblet
www.html5doctor.com



* Sectioning content element
These four elements (and their headings) are used by HTML5's outlining algorithm to make the document's outline
→ html5doctor.com/outline

2011-07-22 v1.5
For more information:
www.html5doctor.com/semantics

Les éléments inline

- Apparaissent **au fil du texte**, ils ne sont pas placés les uns au dessus des autres (ils restent à l'emplacement défini).
- N'ont **pas de marges internes ou externes** par défaut
- Ne sont **pas dédiés à un positionnement précis** (même si cela est possible avec les CSS)
- Servent à modifier, enrichir des portions de textes, apporter du sens

HTML & XHTML – Typologie des éléments

Les éléments inline

Les éléments inline :

- Ne peuvent **contenir que des éléments inline**
(→ **pas de block**)
- Un **élément inline** doit être contenu dans un **élément de type block**



- **time** : date et/ou heure
 - Accompagnée (ou non) de datetime au format ISO AAAA-MM-JJThh:mm:ss (complètement ou non)
 - S'il n'y a pas d'attribut datetime, le contenu doit être au format ISO

```
<time datetime="18:00">18 h</time>  
<time datetime="2010-11-15">15 novembre</time>  
<time>2010-11-15</time>
```

HTML & XHTML – Typologie des éléments

Les éléments inline

Exemples :

- `<a>`
- ``
- ``
- ``
- `<i></i>`
- ``
- ``

HTML – Structure des documents

Les éléments inline

Note : en HTML4, certaines balises définissaient un rendu visuel (ex : `` = texte en gras. Ce n'est plus le cas en HTML5.

- **b** (avant : bold) ne signifie pas « en gras », mais stylistiquement décalé sans avoir une importance supplémentaire (sinon, utiliser strong)
- **i** (avant : italic) ne signifie pas « en italique », mais « dans une voix ou une humeur alternative » sans emphase ni importance particulière (sinon, utiliser em)
- **hr** (avant : ligne horizontale) prend un sens sémantique de séparation entre paragraphes (potentiellement rendu comme une ligne horizontale)
- **small** ne signale plus un texte en petite taille, mais caractérise des éléments « en petits caractères » comme un copyright ou une référence écrites « en petits caractères »

Les structures universelles

div et span = éléments pour structurer des documents
Web (en association avec les CSS)

div = élément de type bloc

span = élément en ligne.

n'apportent aucune contrainte de présentation, ils sont
« neutres » à cet égard. **Ils servent à « ajouter » de la
structure.**

*Attention, ces éléments n'ont pas de sens particulier, ils sont neutres
également sur le plan de la sémantique. Par conséquent, ils ne doivent pas
remplacer systématiquement les autres éléments.*

Un document est positionné dans une **arborescence de répertoires et de fichiers**.

Le répertoire de plus haut niveau (contenant tous les autres documents ou fichiers), est appelé la « **racine** ».

Les chemins absolus

On peut exprimer la destination d'un lien de manière absolue (depuis la racine) :

/rep/sousrep/index.html

*Problème : si on déplace l'ensemble des documents, par exemple pour les inclure dans un répertoire placé différemment par rapport à la racine, **les chemins ne seront plus valides.***

HTML – Les chemins

Les chemins relatifs

- Descendre dans l'arborescence, vers un sous répertoire :
nom_sous_repertoire/ ou **./nom_sous_repertoire/**
./ représentant le positionnement courant
- Remonter dans l'arborescence :
../
- Remonter de plusieurs niveaux :
../../../
- Remonter puis de redescendre dans l'arborescence :
../../autre_repertoire/autre_sous_repertoire

- **Pas destiné à remplacer le positionnement !**
- **Balises `<table>` `</table>`**
- Il est souhaitable d'ajouter à la balise `<table>` l'attribut **summary** = indiquer un résumé du tableau (`<table summary="ce que contient le tableau">`).

- Chaque **ligne** est encadrée par `<tr> </tr>`
- Les cellules d'en-tête sont encadrées par `<th> </th>`
- Les cellules de valeur sont encadrées par `<td> </td>`

- Les balises **<thead></thead>** **<tfoot></tfoot>** **<tbody></tbody>** permettent de structurer les tableaux
- La balise **<caption></caption>** permet d'indiquer la légende du tableau.

Exemple

```
<table>
  <caption>L'acutegende</caption>
  <thead>
    <tr>
      <th>Cellule d'en t&ecirc;te A</th>
      <th>Cellule d'en t&ecirc;te B</th>
    </tr>
  </thead>
  <tfoot>
    <tr>
      <td>Cellule de pied de tableau A</td>
      <td>Cellule de pied de tableau B</td>
    </tr>
  </tfoot>
```

...suite

```
<tbody>
  <tr>
    <td>Valeur A ligne 1</td>
    <td>Valeur B ligne 1</td>
  </tr>
  <tr>
    <td>Valeur A ligne 2</td>
    <td>Valeur B ligne 2</td>
  </tr>
</tbody>
</table>
```

Légende du tableau	
Cellule d'en tête A	Cellule d'en tête B
Valeur A ligne 1	Valeur B ligne 1
Valeur A ligne 2	Valeur B ligne 2
Cellule de pied de tableau A	Cellule de pied de tableau B

Colonnes étendues

```
<table>
  <tr>
    <th colspan="2">Cellule d'en t&ecirc;te &eacute;tendue en
    largeur</th>
  </tr>
  <tr>
    <td>Valeur A ligne 1</td>
    <td>Valeur B ligne 1</td>
  </tr>
  <tr>
    <td>Valeur A ligne 2</td>
    <td>Valeur B ligne 2</td>
  </tr>
</table>
```

L'attribut **colspan** crée des cellules qui s'étendent sur plusieurs cellules d'un tableau, en ligne

Cellule d'en tête étendue en largeur	
Valeur A ligne 1	Valeur B ligne 1
Valeur A ligne 2	Valeur B ligne 2

Lignes étendues

```
<table>
  <tr>
    <th>Cellule d'en t&ecirc;te A</th>
    <th>Cellule d'en t&ecirc;te B</th>
  </tr>
  <tr>
    <td rowspan="2">Valeur A ligne 1 et 2 (&eacute;tendu)</td>
    <td>Valeur B ligne 1</td>
  </tr>
  <tr>
    <td>Valeur B ligne 2</td>
  </tr>
</table>
```

L'attribut **rowspan** crée des cellules qui s'étendent sur plusieurs lignes d'un tableau

Cellule d'en tête A	Cellule d'en tête B
Valeur A ligne 1 et 2 (étendu)	Valeur B ligne 1
	Valeur B ligne 2

Balise ****. Ses principaux attributs sont :

- src : emplacement du fichier source de l'image
- width : largeur
- height : hauteur
- alt : texte qui apparaît lorsque l'image ne s'affiche pas et comme info bulle de l'image

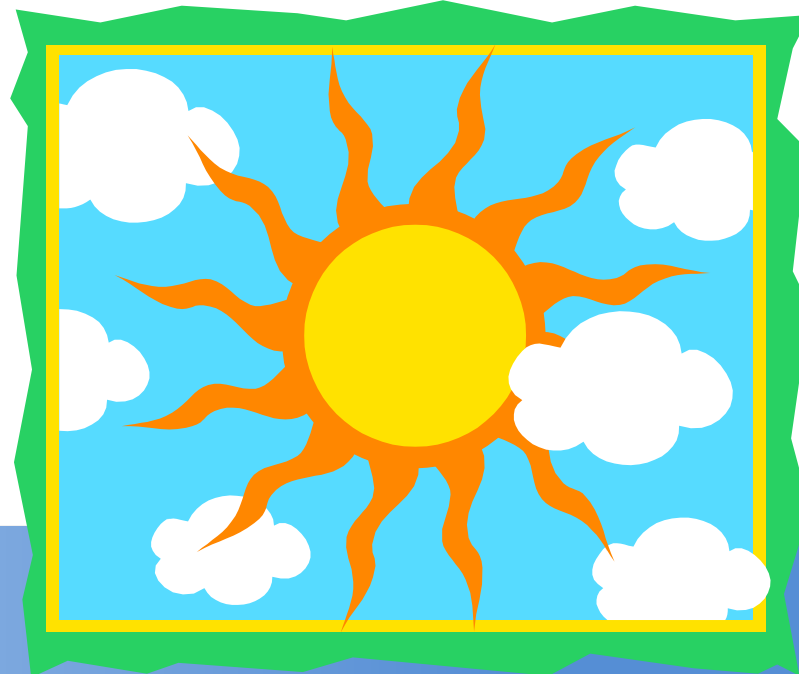
```

```

Les principaux formats d'images

GIF (Graphic Interchange Format)

- Pour les graphismes aux tracés simples, sans dégradé de couleurs
- Limité à une palette de 256 couleurs (choisies parmi des millions).
- Peut être animé.
- Gère la transparence.



JPG OU JPEG (Joint Picture Expert Group)

- Adapté aux photos ou aux images présentant de nombreux dégradés
- Gère la compression (avec perte)



PNG (Portable Network Graphic)

- A l'origine : format alternatif au GIF
- Peut remplacer le GIF (éventuellement le JPG)
- Méthode de compression améliorée
- Attention, il n'est pas supporté par les anciennes versions des navigateurs.
- La transparence peut poser quelques soucis avec certaines versions de navigateurs

- Lien vers une autre page - même site - même répertoire
`Titre du lien`
- Lien vers une autre page - même site - autre répertoire
`Titre`
- Lien vers une autre page - nouvelle fenêtre du navigateur
`Lien`

- Liens vers une page du même site
Suite
- Liens vers une page d'un autre site
Un site
- Lien vers un fichier
Document PDF
Document PDF

- Lien vers un fragment de document

- Définition d'un fragment :

`Sommet`

- Accès au fragment :

`Lien`

`Lien`

- Lien sous forme d'image vers une autre page
``
- Lien spécifique pour téléchargement de fichier (tous formats)
``
le nom de fichier par défaut n'est pas requis : on peut écrire
``

Valider ses documents ?

La validation des documents, même si elle est **nécessaire**, n'est **pas suffisante** : seule la validité de la syntaxe est vérifiée, en aucun cas la qualité structurelle ou sémantique du document.

validator.w3.org

Les CSS

Cascading Style Sheets

- **Rôle, usages**
- **Sélecteurs**
- **Syntaxes de regroupement**
- **Intégration dans les documents HTML et XHTML**
- **L'héritage**
- **Présentation de quelques propriétés**
- **Le positionnement**
- **Les types de média**
- **Le principe de cascade**
- **Bonne pratiques**

- **CSS= Cascading Style Sheets**
= feuilles de style en cascade
 - Ensemble de règles
 - Qui sélectionnent les éléments HTML
 - Qui leur associent des **caractéristiques de style.**
- **4 versions** : CSS 1, CSS 2, CSS 3 et CSS 4, qui est en cours d'élaboration

- **Séparation** du contenu et de la présentation
- **Positionnement** et **style**
- Construction d'une **charte graphique**
(et d'une cohérence globale)
- Factorisation des règles de style

- Séparation du contenu et de la présentation :
 - **Conserver la lisibilité des documents même avec des navigateurs ne supportant pas les CSS**
 - Permettre leur **consultation avec d'autres supports** (assistants personnels, synthèse vocale, navigateurs braille...)
 - Améliorer l'**accessibilité** des documents
 - **Améliorer l'impression** des documents

- Une **règle** se compose de :
 - Un **sélecteur**
 - Une **déclaration** entre **accolades** { }, composée de propriétés séparées par des ;
 - Une propriété est constituée de
 - Un **mot-clé**, suivi de ":"
 - La (les) **valeur(s)** associée(s)

Exemple (pour appliquer une taille de 18 pixels aux titres de niveau 1), la règle est :

```
h1 { font-size:18px; }
```

- Les commentaires dans les feuilles de style s'écrivent entre `/*` et `*/`

Par exemple :

```
/* commentaires */
```

Commenter son code est toujours conseillé, notamment lorsque les feuilles de style deviennent longues / complexes

Sélecteur = « lien » entre le document HTML et la feuille de style (ou les feuilles de style).

Permet au CSS d'identifier (sélectionner) le(s) élément(s) au(x)quel(s) appliquer une règle.

3 types simples de sélecteurs :

- les balises
- les classes
- les identificateurs

Toutes les balises HTML peuvent servir de sélecteurs à des règles CSS.

Exemples

- Pour agir sur les paragraphes

```
p { font-size: 14px; }
```

- Pour agir sur les liens

```
a { font-family: tahoma, arial, verdana; }
```

Les classes HTML = familles d'éléments librement définies

```
<p id="p1" class="maclasse">...</p>  
<span id="s2" class="maclasse">...</span>
```

Sélection en CSS : nom de la classe préfixé d'un point

Exemple

```
.maclasse { color: green; }
```

Les classes peuvent être **restreintes à un élément**

- Le point est précédé de l'élément auquel la classe peut être appliquée
- La classe ne s'applique qu'aux éléments correspondants

```
p.maclasse {color : green;}
```

- La règle s'appliquera à **tout paragraphe** de classe maclasse
- Mais PAS aux autres éléments même s'ils sont de la classe maclasse (div, h1, span, table ou autre)

Les identificateurs ne peuvent porter que sur un seul élément, **unique** et identifié (de fait) dans un document HTML

Le sélecteur d'un identifiant est écrit avec un dièse (#) comme préfixe

```
#monidentificateur { ... }
```

Exemple :

```
#grandtitre {  
    font-weight: bold;  
}
```

Côté HTML :

```
<h1 id="grandtitre">Mon titre</h1>
```

Quelques remarques en guise de conclusion

Utiliser des classes ou des identificateurs ?

- Lorsque c'est possible, lorsqu'un **élément est identifié de manière unique** dans un document, on privilégie l'utilisation d'un **identificateur** : le code est ainsi plus facile à lire et à maintenir.
- Les **classes** sont adaptées pour des **éléments génériques**, (éventuellement) appelés à être utilisés plusieurs fois dans un même documents : « types » de paragraphes, éléments redondants (mise en valeur de portions de texte) ...

4 façons d'incorporer les directives de style au XHTML.

- Données incorporées dans les balises
- Données incorporées dans l'en-tête du document
- Feuille externe attachée ou liée
- (Feuille de style importée)

Intégration des CSS dans les documents

Directives de style dans les balises

Données incorporées dans les balises

Avec l'attribut style dans le corps du document.

```
<p style="color:red; font-weight:bold;">
```

Rouge gras

```
</p>
```


Intégration des CSS dans les documents

Directives de style dans l'en-tête

Données incorporées dans l'en-tête du document

Avec l'élément `<style>` dans l'en-tête du document.

Les règles s'appliquent alors à l'ensemble du document.

```
<head>
```

```
...
```

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

```
    h4 { color:green }
```

```
</style>
```

```
</head>
```

Intégration des CSS dans les documents

Feuille externe attachée ou liée

Avec l'élément **<link>** vers une feuille de style externe, placé **dans l'en-tête du document**

```
<head>
```

```
...
```

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

```
</head>
```

```
...
```

Les règles de style se trouvent dans un fichier monstyle.css auquel le document fait appel ; Il est possible et parfois utile de lier plusieurs feuilles de styles successivement.

Remarque : en appliquant un style « par défaut » à un élément englobant, dans le cas où ce style n'est pas redéfini, les éléments « enfants » de body, et les « enfants des enfants » hériteront de ce style

```
<div style="color:red; font-weight:bold;">
```

Rouge gras

```
<div>
```

Rouge gras aussi

```
</div>
```

```
</div>
```

La div englobée hérite des propriétés de style

Note : définir un style sur **body** = l'appliquer par défaut à tous les éléments de la page

NB : Certaines propriétés de style ne sont pas transmises de l'élément parent à l'élément enfant, c'est le cas de margin, padding (et d'autres propriétés de bloc). Il y aussi certains bugs de navigateurs anciens

Positionnement "dans le flux"

- Par défaut, le navigateur construit l'affichage au fur et à mesure que les descriptions des éléments lui parviennent : un bloc est placé en dessous du précédents, les éléments inline sont placés les uns à côté des autres.
- Ces éléments sont alors dits "**dans le flux**", sa position dépend alors de ses propres marges et des marges internes du conteneur et des éléments adjacents.

Le positionnement

Positionnement dans le flux

```
div {  
  width: 600px ;  
  padding-top: 20px ;  
  border: solid 1px black;  
}  
p {  
  margin-left: 20px ;  
  margin-bottom: 20px ;  
  width: 300px;  
  border: solid 1px black;  
}
```

```
<div>  
  <p>paragraphe</p>  
  <p>paragraphe</p>  
</div>
```

paragraphe

paragraphe

Le positionnement

Positionnement relatif


- Le **positionnement relatif** permet d'inscrire un contenu en flux normal, puis de le décaler horizontalement et/ou verticalement. Le contenu suivant n'est pas affecté par ce déplacement, qui peut donc entraîner des chevauchements.
- Les propriétés **top**, **right**, **left**, **bottom**, permettent de fixer la position relative.

Le positionnement

Positionnement relatif

```
<div>
<p id="premier">paragraphe</p>
<p id="second">paragraphe</p>
</div>
```

```
...
div {
  width : 600px ;
  padding-top : 20px ;
  border: solid 1px black;
}
p#premier {
  margin-left : 20px ;
  width: 300px;
  border: solid 1px black;
}
p#second {
  margin-left : 20px ;
  width: 300px;
  border: solid 1px black;
  position: relative;
  left: 4px;
  bottom: 22px;
}
```



paragraphe

paragraphe

Le positionnement

Positionnement relatif

Autre exemple : le décalage est relatif à la position normale de l'élément dans le bloc parent

Un paragraphe avec un élément décalé du reste du texte.

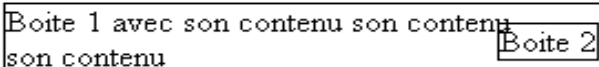
```
.decale {  
  position: relative;  
  bottom: 5px;  
  border: solid 1px black;  
}  
...  
<p>Un paragraphe avec  
<span class="decale">un  
&eacute;l&eacute;ment  
d&eacute;cal&eacute;.</span>  
du reste du texte.</p>  
...
```

Le positionnement

Positionnement absolu

- La position de l'élément est déterminée de manière **absolue** dans son conteneur parent **positionné** le plus proche, ou à défaut, dans la fenêtre du navigateur
- On utilise la propriété **position**, avec la valeur **absolute**, pour positionner un élément de manière absolue.
- Les propriétés **top**, **right**, **left**, **bottom**, permettent alors de fixer la position.

```
#boite1 {  
    position: relative;  
    width: 300px;  
    border: solid 1px black;  
}  
#boite2 {  
    position: absolute;  
    top: 10px;  
    right: 30px;  
    border: solid 1px black;  
}  
...  
<div id="boite1">  
    <p>Boite 1</p>  
    <div id="boite2">Boite 2</div>  
</div>
```



Boite 1 avec son contenu son contenu
son contenu

Boite 2

Le positionnement

Positionnement fixé

Le **positionnement fixé** est très comparable au positionnement absolu, sauf que l'élément fixé **reste à sa place sur l'écran** même lorsque l'utilisateur fait défiler le contenu.

Un élément fixé est comme « ancré » à sa place.

On utilise la propriété **position**, avec la valeur **fixed**.

Les propriétés **top**, **right**, **left**, **bottom**, permettent alors de définir la position.

Le positionnement

Positionnement flottant

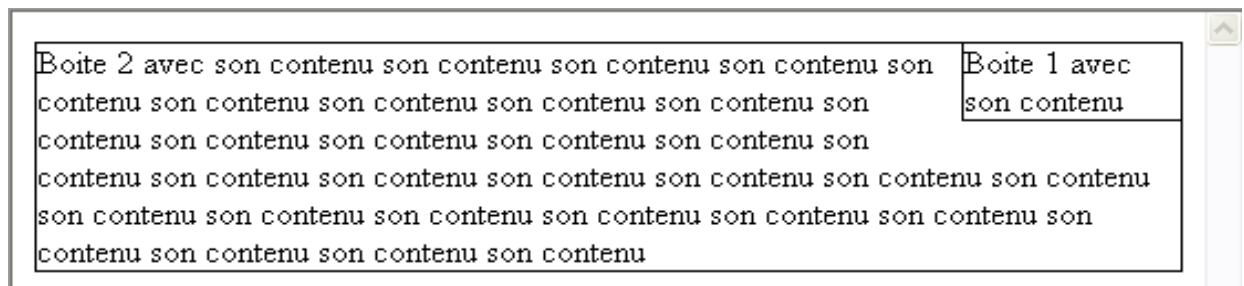
- On utilise la propriété **float**, avec la valeur **left** ou **right**, pour positionner un élément de manière flottante.
- L'élément prend **place à gauche (ou à droite) dans l'élément conteneur**.
- L'élément suivant occupera la **place laissée libre**.
- La boîte 1 se place de manière flottante à droite, la boîte 2 occupe l'espace restant.

```
#boite1 {
    float: right;
    width: 100px;
    border: solid 1px black;
}

#boite2 {
    border: solid 1px black;
}

...

<div id="boite1">Boite 1</div>
<div id="boite2">Boite 2</div>
```

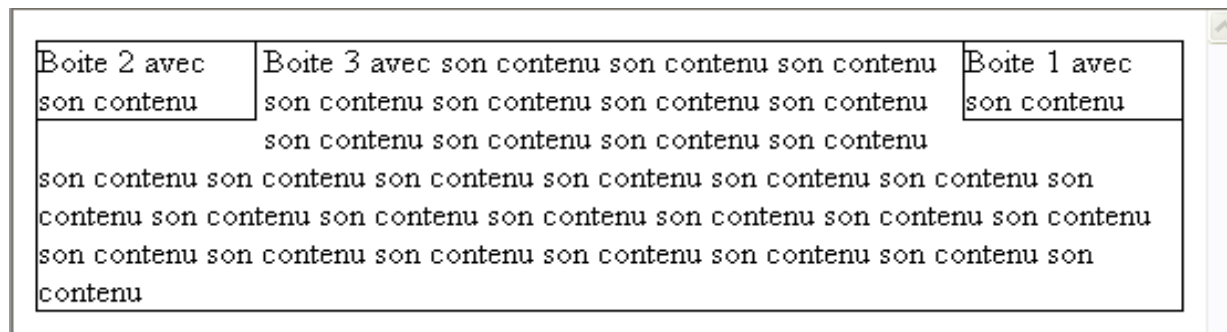


Le positionnement

Exemple avec 2 boîtes flottantes

```
<div id="boite1">Boite 1</div>
<div id="boite2">Boite 2</div>
<div id="boite3">Boite 3</div>
```

```
...
#boite1 {
    float: right;
    width: 100px;
    border: solid 1px black;
}
#boite2 {
    float: left;
    width: 100px;
    border: solid 1px black;
}
#boite3 {
    border: solid 1px black;
}
```

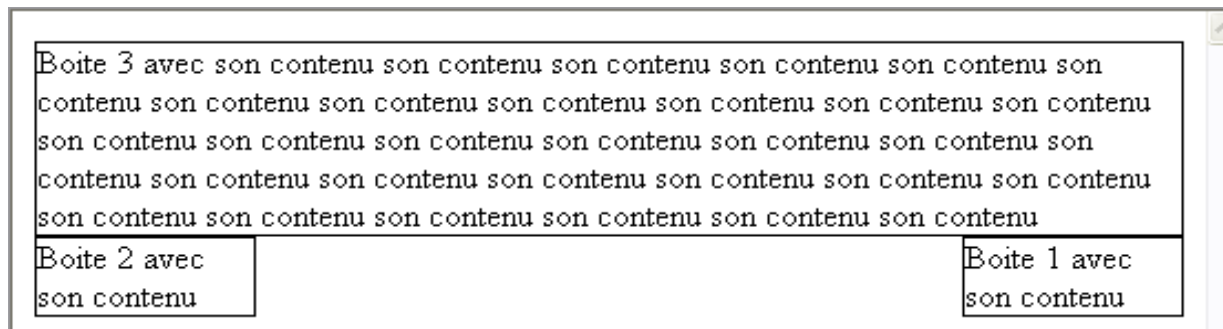


Le positionnement

Remarque : attention, si dans le flux du document, la boîte 3 était inscrite avant les autres boîtes, celle-ci, non positionnée, prendrait toute la largeur du document. Les autres boîtes occuperaient donc leur emplacement « flottant », mais en dessous.

```
<div id="boite3">Boite 3</div>
<div id="boite1">Boite 1</div>
<div id="boite2">Boite 2</div>
```

```
#boite1 {
    float: right;
    width: 100px;
    border: solid 1px black;
}
#boite2 {
    float: left;
    width: 100px;
    border: solid 1px black;
}
#boite3 {
    border: solid 1px black;
}
```



Le positionnement

Positionnement flottant

- La propriété **clear** permet d'interdire le « voisinage » d'un élément « flottant ».
- Elle accepte 3 valeurs
 - left** : interdit le « voisinage » d'un élément « flottant » à gauche
 - right** : interdit le « voisinage » d'un élément « flottant » à droite
 - both** : interdit le « voisinage » d'un élément « flottant » des deux cotés

Le positionnement

Positionnement flottant

```
#boite2 {  
    float: left;  
    width: 100px;  
    border: solid 1px black;  
}  
#boite3 {  
    clear: both;  
}  
<div id="boite2">Boite 2  
</div>  
<div id="boite3">Boite 3  
</div>
```

Boite 2 avec
son contenu

Boite 3 avec son contenu son
contenu son contenu son cor
son contenu son contenu son
contenu son contenu son cor
son contenu son contenu son

- Par défaut, le dernier élément (dans l'ordre d'écriture du code) se place au dessus des éléments précédents.
- La propriété **z-index** permet de changer ce placement : dans un même élément conteneur, les éléments ayant un z-index supérieur sont placés au dessus des éléments ayant un z-index inférieur.
- Ceci permet de placer les éléments les uns au dessus des autres.

Les CSS permettent de gérer des **directives de style alternatives pour différents médias** (périphériques de consultation, d'impression, périphériques vocaux et braille...)

Types de media

Attribut media

- Lors du lien fait avec une feuille de style l'attribut **media** (balise **<link>**) permet de spécifier un ou plusieurs média(s) visé(s) par la feuille de style.
- Cet attribut est valable aussi lorsque l'on ouvre un **bloc de déclaration de style** dans l'en tête d'un document

```
<link href="default.css" rel="stylesheet" type="text/css" media="screen">  
<link href="default.css" rel="stylesheet" type="text/css" media="print">  
<style type="text/css" media="screen,print">  
...  
</style>
```

Par défaut, si l'attribut média n'est pas spécifié, les directives s'appliquent pour tous les médias (all)

@media

- Indique un bloc de règles qui ne s'appliquent qu'aux médias spécifiés.
- Cette règle est utilisable dans les feuilles de style liées ou importées, ou dans les directives fixées dans l'en tête des documents XHTML.

```
@media print {  
    body {  
        background-color: #ffffff;  
        color: #000000;  
    }  
}
```

Il est possible d'indiquer une liste de médias en les séparant par des virgules

Types de media @-règles

•all	Tous médias
•aural	Parole et synthétiseurs de sons
•braille	Interface tactiles braille
•embossed	Impression braille
•handheld	Petits dispositifs ou dispositifs tenus en main
•print	Impression
•projection	Projection
•screen	Ecrans d'ordinateurs
•tty	Teletypes, terminaux...
•tv	TV

Quelques remarques en guise de conclusion

Erreurs classiques

- L'élément `div` ne remplace pas tous les éléments : Il vaut mieux privilégier les éléments HTML ayant une valeur sémantique.
- Il n'est pas indiqué de créer des divisions « à outrance » dans les documents.
- Tous les éléments n'ont pas besoin de « `class` ». Multiplier les classes et par conséquent les attributs d'éléments HTML (pour les rattacher à une classe) rend le code beaucoup plus lourd. Il est préférable d'utiliser la sélection hiérarchique plutôt que de vouloir « typer » tous les éléments par des classes.

Quelques remarques en guise de conclusion

Règles d'écriture, code réutilisable

Ne pas négliger les commentaires pour , par exemple :

Donner en introduction des informations sur l'auteur, les évolution de la feuilles de style, la todo list...
(voir <http://cssdoc.net/> par exemple)

Proposer un sommaire de la feuille de style (ses sections)

Normaliser les codes couleur utilisés

Diviser la feuilles de style en sections et sous-sections clairement visibles et identifiées