

Internet et HTML

Cours maj en 2022 – Attention, les choses évoluent rapidement...

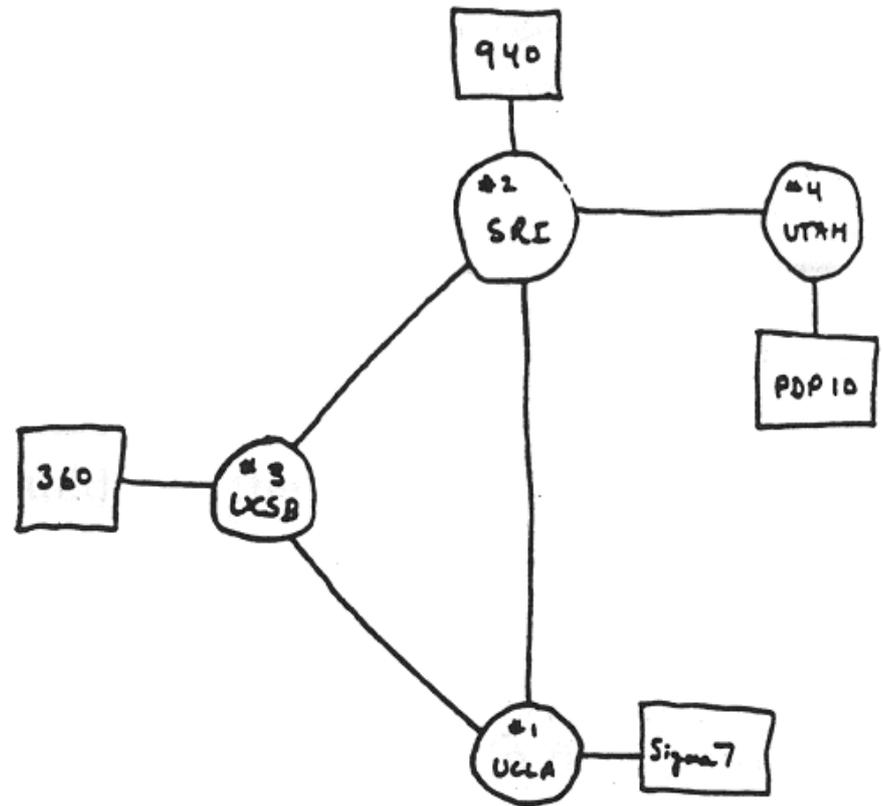
PANORAMA - ORTF
1968



https://www.youtube.com/watch?v=9qzUpN9CUoU&ab_channel=LeMonde

Au départ : Arpanet

Arpanet est l'ancêtre de l'Internet actuel. Il a été créé en 1967 par des universités américaines pour pouvoir communiquer entre elles via des ordinateurs grâce à un réseau de câbles enfouis.



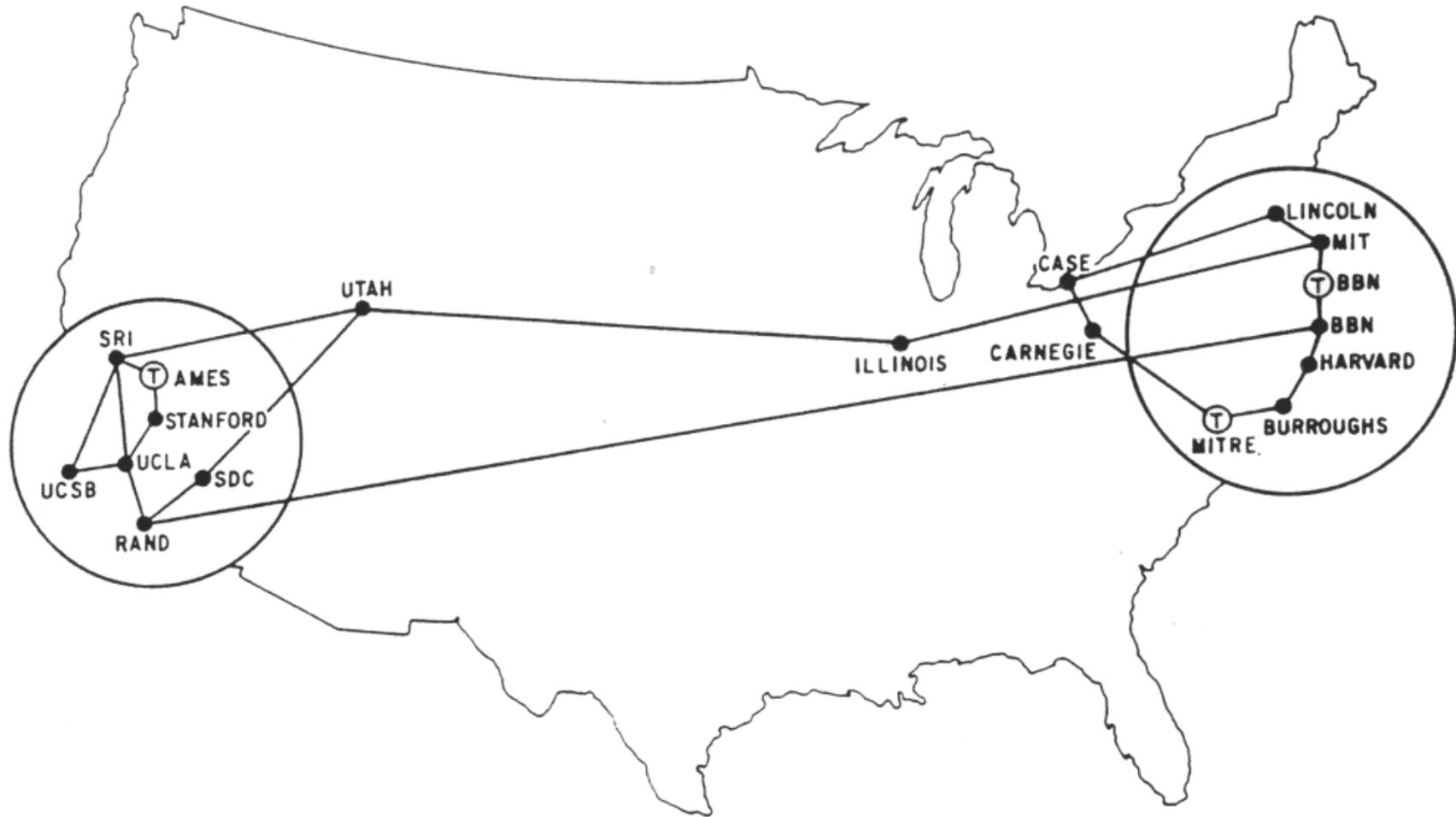
THE ARPA NETWORK

DEC 1969

4 NODES

FIGURE 6.2 Drawing of 4 Node Network
(Courtesy of Alex McKenzie)

Arpanet en 1971



MAP 4 September 1971

1989 : le WWW

(World Wide Web)

Création du World Wide Web — communément appelé WWW, W3, ou le web — est un système de pages web publiques interconnectées à travers l'Internet. Le web et l'internet ne sont pas la même chose : le web est l'une des nombreuses applications bâties au-dessus de l'internet.

Tim Berners-Lee a proposé l'architecture de ce qui a été ensuite connu sous le nom de World Wide Web. Il a créé les premiers serveur web, navigateur web, et page web sur son ordinateur au laboratoire de recherches scientifiques CERN en 1990.



Logo historique du World Wide Web par Robert Cailliau

1991 : le HTML

(Hypertext Markup Language)

C'est un langage de balisage permettant d'écrire de l'hypertexte, d'où son nom. Tim Berners-Lee présente le langage HTML en 1991. Il utilise le réseau civil d'Arpanet, auquel le CERN est connecté. Il reprend l'idée d'hypertexte, inventée en 1960, et simplifie un protocole existant, le SGML.

Il est rapidement adopté par la communauté scientifique, grâce au travail d'évangéliste du belge Robert Cailliau.

Tim Berners Lee

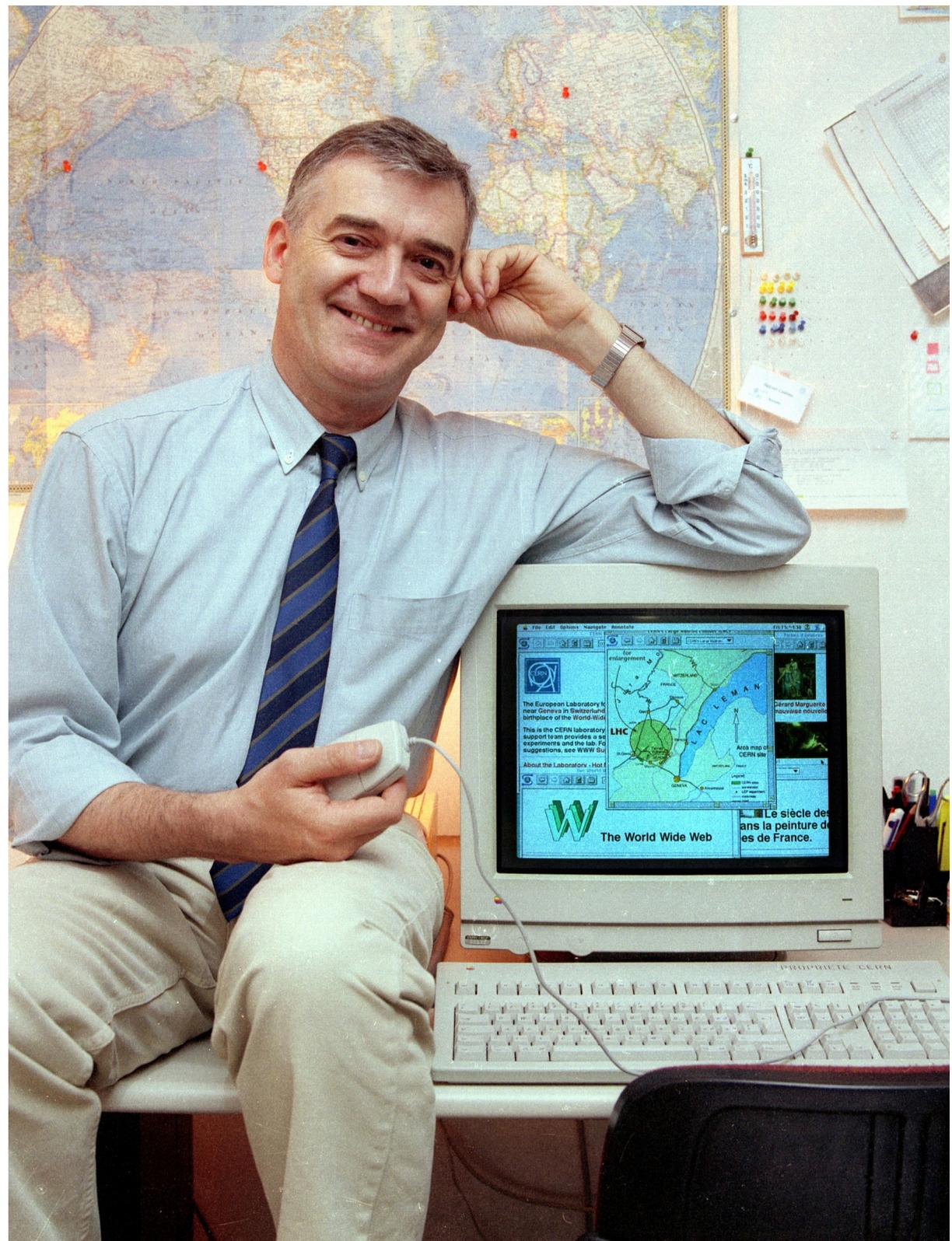
https://www.youtube.com/watch?v=PbCG_BeL5dY&ab_channel=FranceCulture



Robert Cailliau



https://www.youtube.com/watch?v=SBMvx4OUfVw&ab_channel=TV5MONDEInfo



1993 : premier navigateur

Tim Berners-Lee met au point le www en 1991, qui facilite la consultation des pages. Le premier navigateur, « Mosaic » et la gestion des noms de domaines se met en place en 1993.

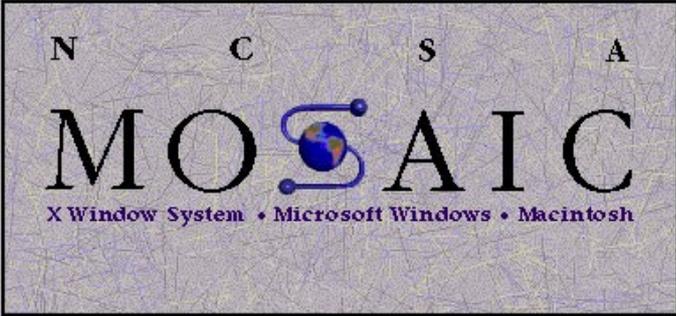
C'est parti pour une exploitation du web par le grand public et des investisseurs privés.

NCSA Mosaic for MS Windows

File Edit Options Navigate Hotlist Annotate Help

Document Title: NCSA Mosaic Home Page

Document URL: http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/NCSAMosaicHome.html



Welcome to NCSA Mosaic, an Internet information browser and [World Wide Web](#) client. NCSA Mosaic was developed at the [National Center for Supercomputing Applications](#) at the [University of Illinois](#) in --> Urbana-Champaign. NCSA Mosaic software is [copyrighted](#) by The Board of Trustees of the University of Illinois (UI), and ownership remains with the UI.

Jan `97

The Software Development Group at NCSA has worked on NCSA Mosaic for nearly four years and we've learned a lot in the process. We are honored that we were able to help bring this technology to the masses and appreciated all the support and feedback we have received in return. However, the time has come for us to concentrate our limited resources in other areas of interest and development on Mosaic is complete.

All information about the Mosaic project is available from the homepages.

NCSA Mosaic Platforms:

- [NCSA Mosaic for the X Window System](#)
- [NCSA Mosaic for the Apple Macintosh](#)
- [NCSA Mosaic for Microsoft Windows](#)

World Wide Web Resources The following resources are available to help introduce you to cyberspace and keep track of its growth:

- A [glossary](#) of World Wide Web terms and acronyms
- An [INDEX](#) to Mosaic related documents
- [NCSA Mosaic Access Page](#) for persons with disabilities
- Mosaic and WWW related [Tutorials](#)
- [Internet Resources Meta-Index](#) at NCSA
- [Supported Operating Systems](#) for Internet Exploration

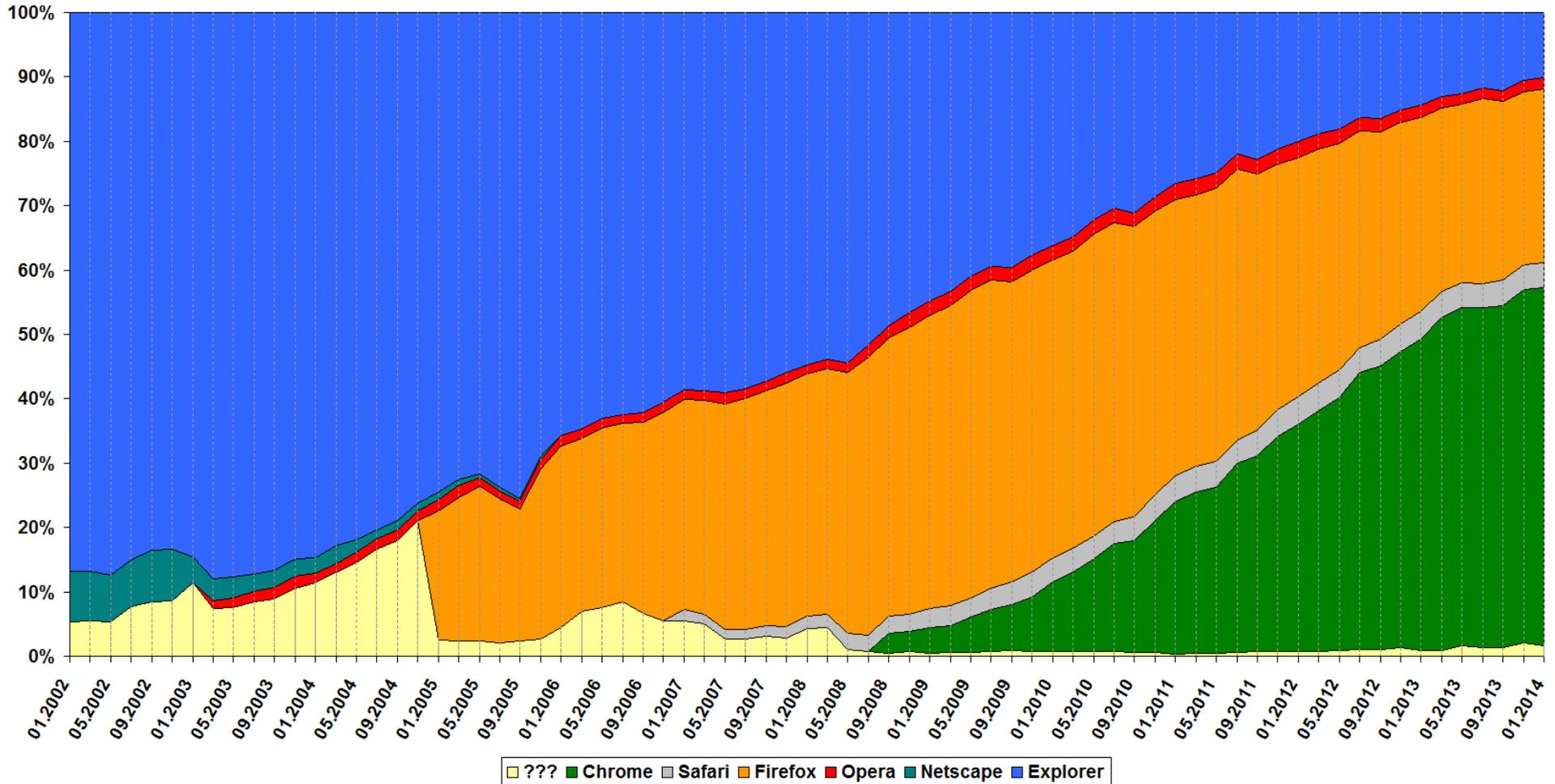
NUM

1994-2001 : la guerre des navigateurs

Netscape Navigator est lancé fin 1993. Il supplante Mosaic en quelques mois. Internet Explorer se lance dans la bataille et avec sa version 4 et une politique agressive. Microsoft décide d'incorporer son navigateur Internet Explorer au système d'exploitation Windows. Une combinaison qui sera fatale à la jeune entreprise. Windows finira par gagner sur son concurrent Netscape en 1997.

Durant ce temps, le web pour le grand public évolue de manière exponentielle.

La guerre des navigateurs



Le code meurtri par la guerre

Durant la guerre des navigateurs, chacun de ceux-ci propose à l'internaute des fonctionnalités que son concurrent n'a pas. Ces fonctionnalités passent par des codes que les développeurs peuvent placer pour rendre les pages plus « cool » : lecture de son, vidéo, couleur des polices, etc.

Par exemple, Windows fait du chantage pour imposer Internet Explorer. Des pages HTML sont créées pour êtres « moches » sous Netscape.

Le code meurtri par la guerre

Ces codes sont rarement adoptés par les navigateurs concurrents. Il en résulte une dérégulation de la norme HTML. Vers 2000, le langage HTML n'a plus de « standard ». Chacun crée ses propres balises.

Pour remettre de l'ordre, l'inventeur du HTML, Tim Berners-Lee crée le W3C.

Le W3C

W3C signifie WWW Consortium. Il s'agit d'un organe qui tente de standardiser les codes employé sur le web pour une meilleure compatibilité et interopérabilité entre ceux-ci. Tim Berners-Lee en est le fondateur en 1994.

Le W3C n'impose pas de règle mais fait des recommandations.



Mozilla

Historiquement, le nom Mozilla a été utilisé au sein de l'entreprise Netscape Communications comme nom de code pour le navigateur web Netscape Navigator. Ce nom proposé par Jamie Zawinski est en fait la contraction du nom Mosaic killer (littéralement tueur de Mosaic) ; killer étant orthographié killa pour correspondre à la prononciation du mot dans le langage courant, le nom s'est donc formé de la façon suivante : Moz (pour Mosaic) et illa (fin de killa) ce qui a donné Moz+illa. Le nom fait également référence au monstre Godzilla. (Source: Wikipédia)

La patience est une vertu

Le W3C a mis du temps à s'imposer, notamment parce que entre 2001 et 2007, il n'y a pas eu de réel concurrent à Microsoft Internet Explorer et que ce logiciel non conforme aux recommandations n'avait pas de pression pour faire le travail de conformation:

La norme, c'était lui.

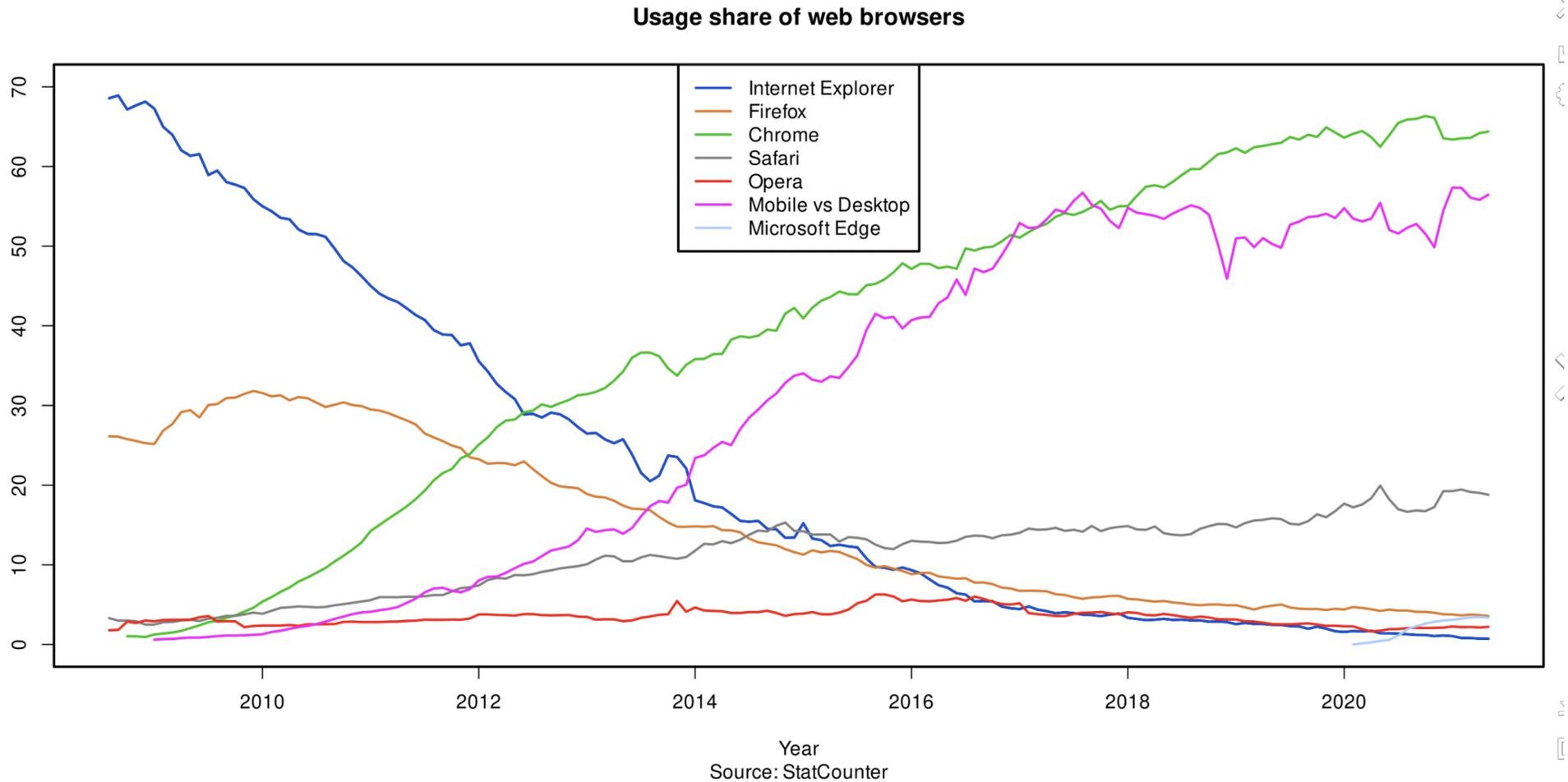


L'avenir du web : consolidation et interopérabilité

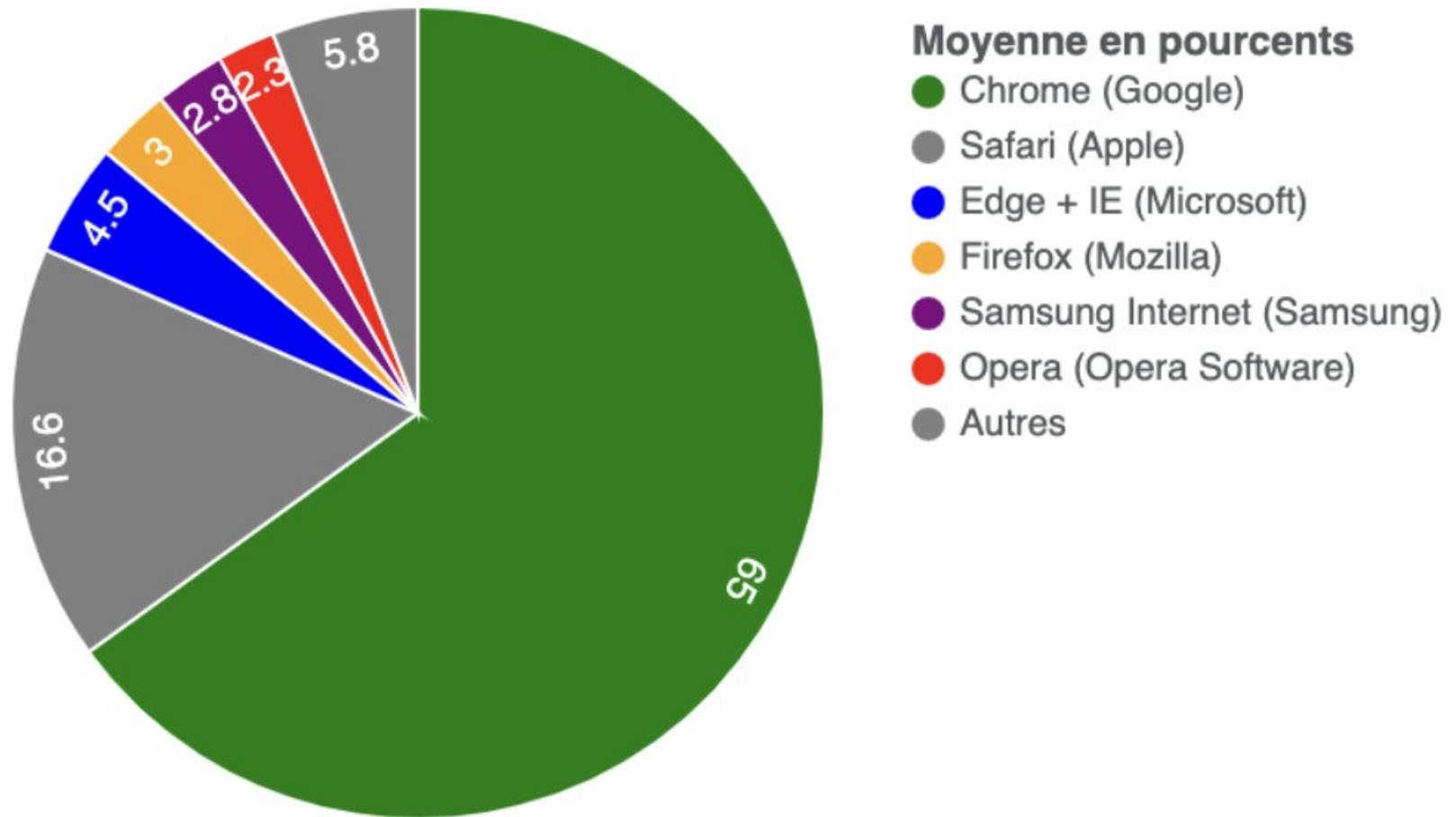
C'est une période de consolidation : le net a trouvé sa place dans les usages. Le HTML5 est déjà là et installe WEB interactif sans plug-in. La fin de Flash rend le WEB plus ouvert.

Chrome et Safari se partagent le marché des navigateurs. L'un vient de Google, l'autre d'Apple. Ce sont deux acteurs des GAFAM. Un autre Firefox est open-source, produit de la fondation Mozilla.

L'évolution des navigateurs



Utilisation des navigateurs (2022)



Web is democracy

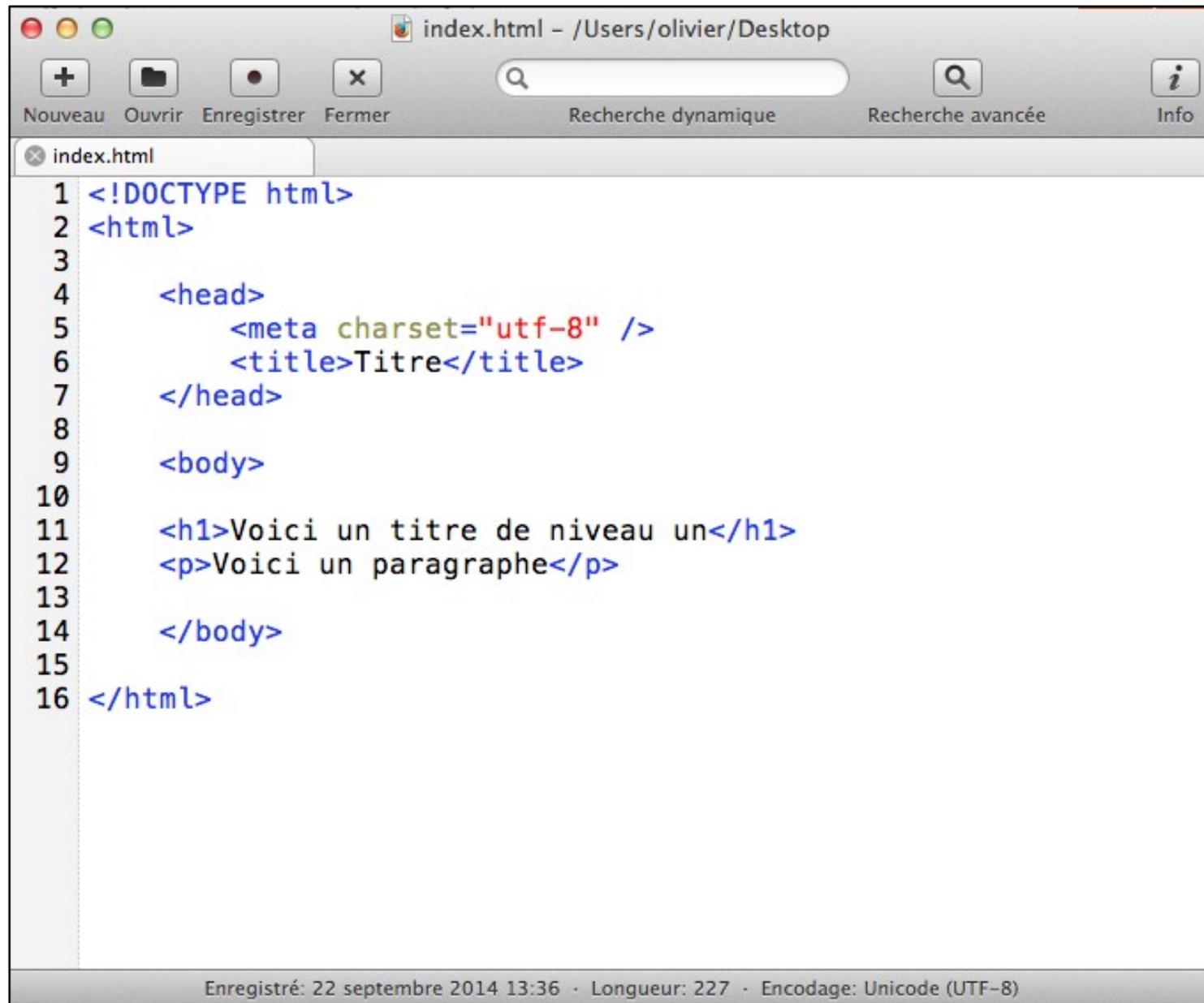
Les navigateurs DOIVENT s'accommoder de la disparité et des erreurs du code pour afficher aux mieux les contenus.

Le cas extrême

Une page html, contenant du texte seul, sera affichée par votre navigateur. Pas la moindre balise html et pourtant un contenu lisible par tous.

-> faites le test ;)

Alors pourquoi bien coder ?



The image shows a screenshot of a text editor window titled "index.html - /Users/olivier/Desktop". The window contains a valid HTML document structure. The code is as follows:

```
1 <!DOCTYPE html>
2 <html>
3
4   <head>
5     <meta charset="utf-8" />
6     <title>Titre</title>
7   </head>
8
9   <body>
10
11   <h1>Voici un titre de niveau un</h1>
12   <p>Voici un paragraphe</p>
13
14   </body>
15
16 </html>
```

At the bottom of the window, the status bar displays: "Enregistré: 22 septembre 2014 13:36 · Longueur: 227 · Encodage: Unicode (UTF-8)".

Alors pourquoi bien coder ?

- Une bonne structure assure une stabilité du rendu visuel
- Une bonne structure est plus facile à modifier
- Une bonne structure est mieux indexée par les moteurs de recherche comme Google (référencement)
- Une bonne structure est un gage de pérennité de la page, elle vieillit moins vite.

Structure sémantique

Une page web décrit sémantiquement un contenu (le `<body>`). Les balises employées sont faites pour identifier les éléments de la page: titres `<h1>`, paragraphes `<p>`, listes ``, etc...

Une autre partie du code (dans le `<head>`) permet au navigateurs d'identifier quel encodage et version du HTML est employée, ce qui évite des erreurs d'affichage.

Comportement visuel

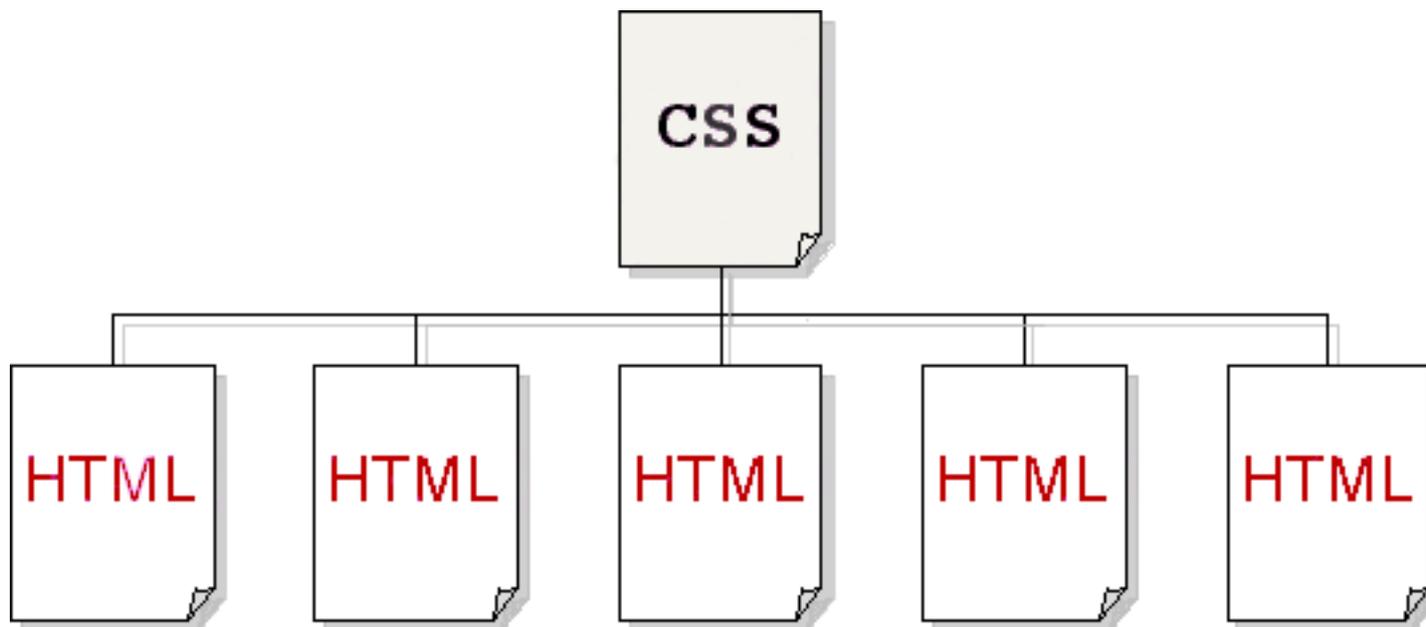
Une structure sémantique bien faite permet aux robots d'indexation, par exemple, de correctement comprendre la structure du document.

Mais si un document bien structuré est lisible, il est souvent assez rustre visuellement.

Plutôt que de placer du code supplémentaire dans la page pour le rendre agréable, on utilisera un autre code, appelé CSS.

Avantages des css

Outre un code plus clair parce séparé en deux fichiers plutôt que mélangé, l'usage des css permet de concentrer la mise en page sur un seul fichier.



C'est parti !

HTML

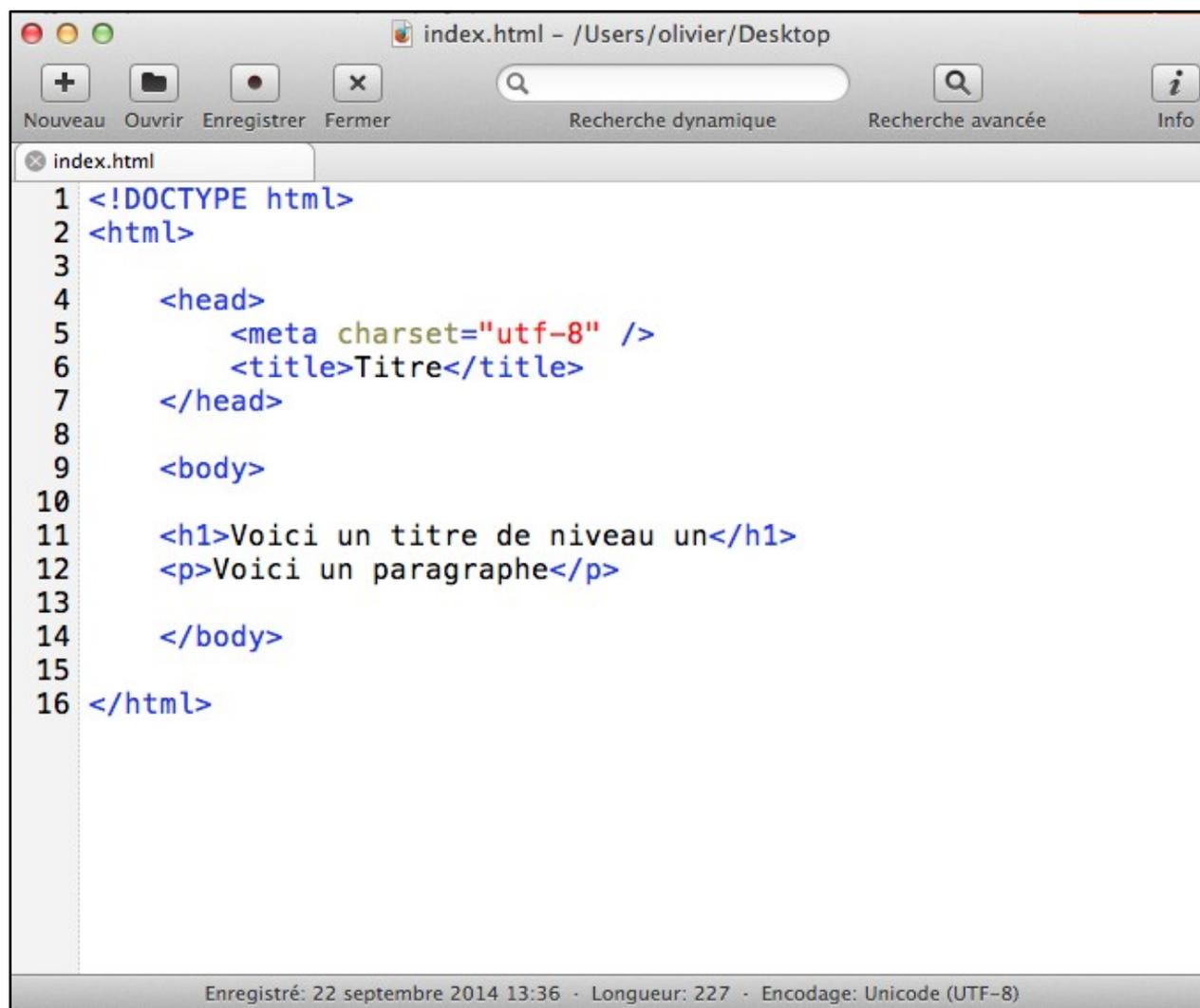
HyperText Markup Language

Le HTML, langage de description

Le html n'est pas un code d'instruction, donc pas un « langage de programmation ».

Il livre un contenu, et un balisage qui donne des indications sémantiques sur ce contenu. Ceci est un titre, un paragraphe, une image, un lien... De plus, il donne des indications sur la page et les normes qu'elle utilise.

Le HTML, langage de description



The image shows a screenshot of a text editor window titled "index.html - /Users/olivier/Desktop". The window contains the following HTML code:

```
1 <!DOCTYPE html>
2 <html>
3
4   <head>
5     <meta charset="utf-8" />
6     <title>Titre</title>
7   </head>
8
9   <body>
10
11   <h1>Voici un titre de niveau un</h1>
12   <p>Voici un paragraphe</p>
13
14   </body>
15
16 </html>
```

At the bottom of the window, the status bar displays: "Enregistré: 22 septembre 2014 13:36 - Longueur: 227 - Encodage: Unicode (UTF-8)".

Une balise

Simple

`<balise>Contenu</balise>`

Ex : `<h1>Mon titre</h1>`

Avec attribut

`<balise attribut="valeur">Contenu</balise>`

Ex : `Clic`

Ex : ``

Balises html, head et body

document hypertextuel

en-tête
contenu

```
<html>  
  
<head>  
  <title>le titre de la page</title>  
</head>  
  
<body>  
  <p>voici le premier paragraphe.</p>  
</body>  
  
</html>
```

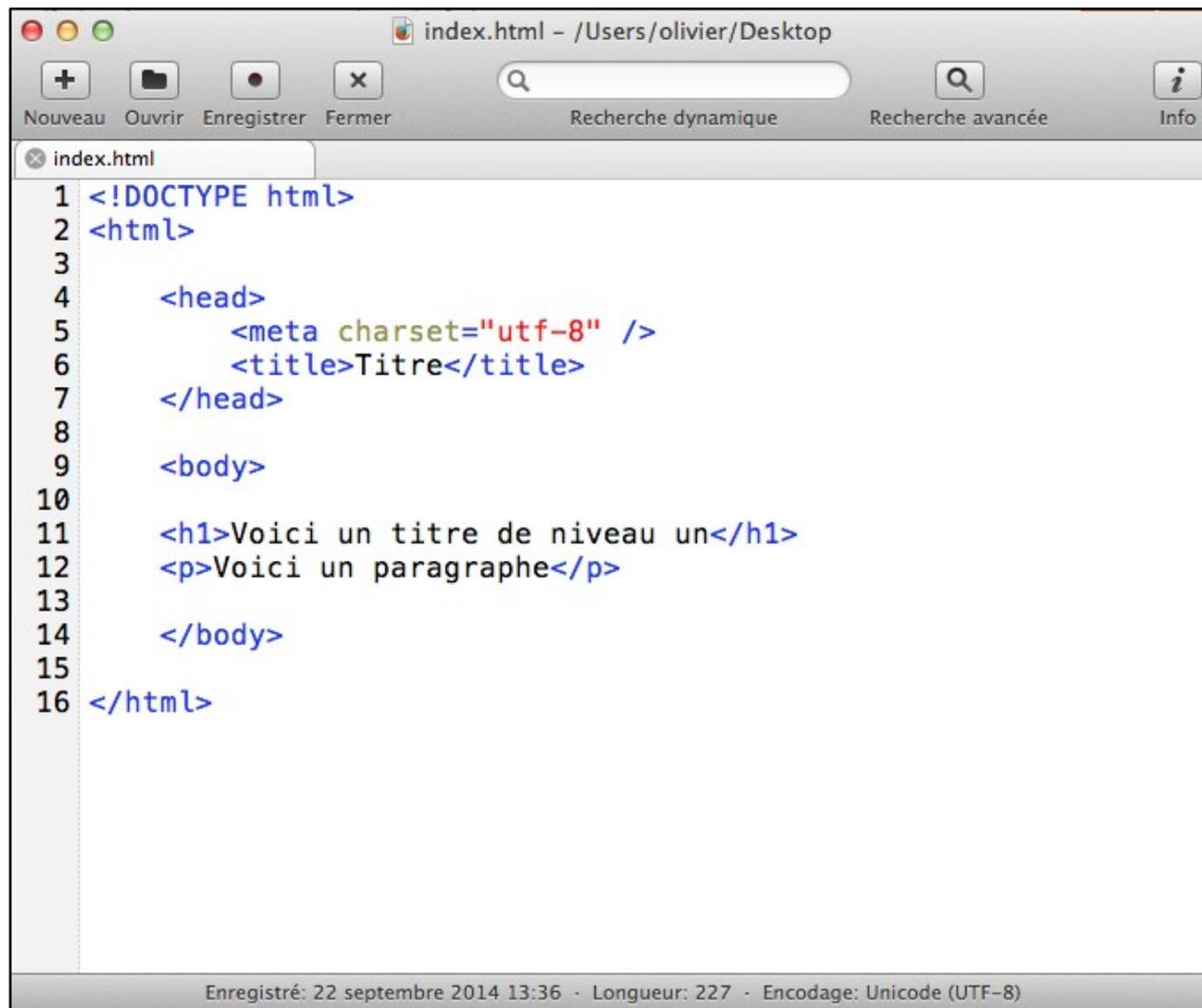
Balise <html> et doctype

Au départ, le document html débute avec le doctype. Celui-ci décrit de quel type de document il s'agit. On utilise actuellement la version simplifiée qui sous-entend html5 :

```
<!doctype html>
```

Ensuite, la première balise est la balise <html>. Elle s'ouvre au tout début et se ferme à la fin. Viennent ensuite la balise <head> et <body>.

Balise <html> et doctype



```
1 <!DOCTYPE html>
2 <html>
3
4   <head>
5     <meta charset="utf-8" />
6     <title>Titre</title>
7   </head>
8
9   <body>
10
11   <h1>Voici un titre de niveau un</h1>
12   <p>Voici un paragraphe</p>
13
14   </body>
15
16 </html>
```

Enregistré: 22 septembre 2014 13:36 · Longueur: 227 · Encodage: Unicode (UTF-8)

Balises `<head>` et `<body>`

La balise `<head>` va contenir toutes les infos que le navigateur (firefox, chrome, safari,...) doit stocker dans sa tête avant d'afficher le contenu de `<body>`.

La balise `<body>` contient le contenu (textes, images, liens,...) qui sera affiché par le navigateur.

L'arbre

Les balises s'imbriquent pour former l'arbre de la page. On peut citer aussi l'exemple des poupées russes.

Ex:

```
<body>
```

```
  <p>
```

```
    <strong>Texte imbriqué important dans le paragraphe
```

```
  </strong>
```

```
  </p>
```

```
</body>
```

Parents et enfants

Lorsque l'on parle des relations entre éléments à l'intérieur de l'arbre, on parle de parents et d'enfants.

Logiquement, les éléments enfants sont ceux contenus dans un élément parent.

Cette notion sera importante lors de la mise en forme visuelle.

Principales balises

<head> : définit l'entête de la page

<body> : le corps de la page, la partie visible

<h1> à <h6> : différents niveaux de titres

<p> : un paragraphe

<a> : lien

**** : image

**** : emphase (italique)

**** : important (pas gras)

** **: une liste “non-ordonnée”, ordonnée, élément de liste

**
**: un retour à la ligne

**<div> et ** : deux balises “non sémantiques”

Comportement visuel par défaut

Disons-le une première fois, car on le répétera :

Toutes les balises, hormis div et span (qui sont neutres sémantiquement comme vu), ont un comportement visuel par défaut !

Par exemple, les [liens sont bleus et soulignés.](#)

A propos des liens

Le html est basé sur le concept d'hypertexte. Une page web **importe** des contenus qui lui sont externes (des fichiers image, son et vidéo) et **renvoie** vers d'autres pages. Tous ces objets sont accessibles grâce à des liens.

Un lien est un texte structuré, qui désigne au navigateur l'endroit où se trouve l'objet à atteindre.

On distingue les liens **relatifs** ou **absolus**.

Liens relatifs

Un lien relatif désigne l'emplacement où se trouve le fichier lié par rapport au fichier actuel.

A la question « où se trouve ton bic », le lien relatif répond « sur la table à côté de moi ».

On obtient :

```
<a href= "table/bic.jpg">Mon bic</a>
```

Avantage des liens relatifs

```
<a href= "table/bric.jpg">Mon bic</a>
```

Les liens relatifs sont généralement courts, et suffisants dans la plupart des cas. Si vous changez d'emplacement votre site sans en changer la structure, les liens resteront corrects. C'est comme déménager le contenu de votre appartement ailleurs.

Par contre, si vous changez de lieu sans embarquer la table, la réponse « sur la table à côté de moi » devient fausse.

Enfin, les liens relatifs ne peuvent joindre que les objets présents sur votre serveur.

Liens absolus

Un lien absolu désigne l'emplacement où se trouve le fichier lié en lui donnant une adresse web complète.

A la question « où se trouve ton bic », le lien absolu répond « sur la terre, en Europe, en Belgique, à Liège, 41 Boulevard de la Constitution, 2^e étage, au 204, sur la table ».

On obtient :

```
<a href= "http://www.terre.net/europe/belgique/liege/41bdvconstitution/  
2/204/table/bic.jpg">Mon bic</a>
```

Avantage des liens absolus

Les liens absolus sont valables quelle que soit l'emplacement de la page qui l'appelle. Il peut donc importer un contenu se trouvant sur un autre serveur que le vôtre.

C'est le principe des vidéos youtube, que vous pouvez mettre dans votre site alors qu'elles ne quittent pas les serveurs de youtube.com

Par contre, si la structure du site où se trouve le document change, le lien absolu est « cassé ».

Enfin, les liens absolus sont parfois très longs.

Block et inline :

Deux types de balise

Les éléments de type “**inline**” se comportent comme du texte : ils occupent en largeur et hauteur l'espace nécessaire à leur affichage, n'ont pas de marges, ne peuvent pas être positionné.

a, img, em et strong sont des balises inline.

Exemple : Voici un texte `avec emphase`.

Bonnes pratiques HTML

- Pensez sémantique
- Ce qui s'ouvre se ferme
- Codez les balises en minuscules
- Mettez les attributs entre guillemets
- Évitez le sur codage
- Commentez votre code : `<!-- comme ceci -->`

Les images

Afin de diminuer leur taille en octets, les images sont compressées, on dit qu'elles sont alors optimisées pour le Web. Cela concerne leur résolution (pour le Web on utilise généralement 72 dpi ce qui correspond à la résolution de l'écran), leurs dimensions (exprimée en pixels) et leur format d'encodage (jpeg, gif, png).

CSS

Cascading Style Sheets

Les CSS entrent dans la danse

Les CSS décrivent le comportement visuel des éléments de la page. Ces comportements peuvent être différents et même en conflit avec la sémantique des éléments, bien qu'il y ait généralement une relation logique entre les deux.

Les CSS ont été inventées pour mettre de l'ordre dans le HTML, et séparer structure du code (sémantique) et mise en page (visuelle).

Le W3C recommande la séparation des codes. C'est une bonne pratique même si ça n'est pas toujours possible.

A quoi servent les CSS

Les CSS s'occupent :

- de la couleur, taille, graisse, famille de typographie des textes
- de la couleur et images de fond des différents éléments de la page
- de bordures, lignes, espaces de séparation entre les éléments de la page
- du placement des différentes informations dans la page (ce qu'on appelle le positionnement)

Ou déclarer un comportement CSS ?

Il existe plusieurs possibilités MAIS la solution recommandée (et donc la seule que nous utiliserons) consiste à écrire le CSS dans un autre document enregistré en .css (exemple: styles.css). Il faudra alors relier le document .html au document .css grâce à un lien à placer dans le <head> du document .html

Ou déclarer un comportement CSS ?

Il existe plusieurs possibilités MAIS la solution recommandée (et donc la seule que nous utiliserons) consiste à écrire le CSS dans un autre document enregistré en .css (exemple: styles.css). Il faudra alors relier le document .html au document .css grâce à un lien à placer dans le <head> du document .html

```
<head>  
  <meta charset="utf-8" />  
  <link rel="stylesheet" href="styles.css" />  
  <title>Mon super site</title>  
</head>
```

Déclarer un comportement CSS : syntaxe

Element { attribut: valeur; }

Par exemple :

```
body {  
    font-family: arial, helvetica, sans-serif;  
    font-size:11px;  
    color: #ccc;  
}
```

Déclarer un comportement CSS : 3 façons d'atteindre les éléments

On peut sélectionner les éléments à styler de trois façons :

1) sélectionner par balise

```
p {  
    color:#fc0;  
    font-size:14px;  
    line-height:140%;  
}
```

Déclarer un comportement CSS :

3 façons d'atteindre les éléments

2) sélectionner par id

Html : `<p id="principal">un texte</p>`

Css :

```
#principal {  
border:1px solid red;  
}
```

Déclarer un comportement CSS :

3 façons d'atteindre les éléments

3) sélectionner par classe

Html : `<p class="niveau1">un texte</p>`

Css :

```
.niveau1 {  
border:1px dashed yellow;  
letter-spacing:0.1em;  
}
```

Héritage CSS

La notion d'héritage est centrale dans le codage css

Certains comportements sont passés des parents vers les enfants.

La définition de l'attribut font-family employée pour la balise body, par exemple va affecter tous les éléments de la page.

Par contre, un margin n'affectera que l'élément pour lequel il est défini.

Sélection hiérarchique

Autre notion importante : la sélection hiérarchique

```
#principal p {  
    color:red;  
}
```

Ne va affecter que les paragraphes situés dans un élément possédant l'id "principal" et aucun autre paragraphe dans la page.

Priorité

Et encore une notion importante !

a { color:green; } supplante le bleu des liens

p a { color:yellow; } supplante la ligne précédente

#principal a { color:red; } supplante la ligne précédente

Si deux règles ont la même importance, la dernière est prise en compte.

Comportement visuel par défaut

Disons-le une deuxième fois :

Toutes les balises, hormis `div` et `span` (qui sont neutres sémantiquement comme `vu`), ont un comportement visuel par défaut !

Par exemple, les `<h1>` jusque `<h6>` ont des tailles par défaut, sont en gras et possèdent une marge en haut et en bas.

Positionnement CSS

<https://openclassrooms.com/fr/courses/1603881-apprenez-a-creer-votre-site-web-avec-html5-et-css3/1606402-decouvrez-dautres-techniques-de-mise-en-page>

Positionnement CSS

Le positionnement permet de modifier le comportement dans l'espace de la page des éléments.

Rappel : seuls les éléments de type "block" peuvent être positionnés.

Il y a 4 types de positionnement.

4 positionnements et une bidouille

On peut positionner un élément dans le flux, en relatif, en absolu, en fixe. Ajoutons le positionnement en float et on a tous les outils disponibles.

A vous de coder les comportements visuel de tout ça.

Par défaut : le flux

Un élément positionné en “static” se place sous son élément frère - ou sœur - qui le précède dans le code et pousse vers le bas l'élément frère - ou sœur - suivant.

```
p {  
  position: static;  
}
```

Un peu bizarre : le relatif

Un élément positionné en “relative” reste dans le flux, mais peut bouger de cette position sans que cela n'affecte l'endroit qu'il occupe, ni la position des éléments frère – ou soeur – qui le suivent.

```
h1 {  
  position: relative;  
  left: -10px;  
}
```

Le king dangereux : l'absolu

Un élément positionné en “absolute“ quitte le flux et se positionne relativement à son ancêtre positionné.

Les éléments suivants l'ignorent, ce qui peut provoquer des chevauchements foireux si on ne pense pas bien le positionnement des éléments.

Un élément en absolu adapte sa taille par rapport à son contenu.

```
p {  
position: absolute;  
left: 0;  
top: 50px;  
}
```

Le frère mal aimé : le fixe

Un élément positionné en “fixed” quitte le flux. Il se positionne relativement à la fenêtre du navigateur.

```
aside {  
  position: fixed;  
  left: 0;  
  top: 50px;  
}
```

La bidouille : le float

Un élément recevant un comportement en float reste lié à sa position dans le flux. Les éléments inline qui le suivent vont respecter son positionnement, mais pas les éléments de type block, qui vont l'ignorer. On peut forcer les éléments suivants à se placer sous lui en leur donnant la propriété “clear” en left, right ou both.

```
img {  
float: right;  
width: 200px;  
}
```

Bonne pratique :

2 colonnes avec le inline-block

- 1) Un élément recevant un comportement en « inline-block » reste lié à sa position dans le flux. Les éléments inline ou block qui le suivent vont respecter son positionnement.**
- 2) Vous devez impérativement travailler au minimum sur 2 block.**
- 3) Vous devez aussi réduire la largeur de chaque block afin qu'ils puissent être côte à côte.**

2 colonnes avec le inline-block

```
aside {  
display : inline-block ;  
width: 100px;  
vertical-align : top ;  
}
```

```
section {  
display : inline-block ;  
width: 200px;  
vertical-align : top ;  
}
```

L'avenir : le flex-box

Il y a plusieurs façons de mettre en page un site. Au fil du temps, plusieurs techniques ont existé :

Au début, les webmasters utilisaient des tableaux HTML pour faire la mise en page.

Puis, CSS est apparu et on a commencé à faire une mise en page à l'aide de la propriété float.

Cette technique avait des inconvénients. Une autre, plus pratique, a consisté à créer des éléments de type inline-block sur la page.

Aujourd'hui, une bien meilleure technique encore existe : Flexbox ! Elle permet toutes les folies (ou presque ;)), et c'est celle que je vous recommande d'utiliser si vous en avez la possibilité, lorsque vous créez un nouveau site. Flexbox est désormais reconnu par tous les navigateurs récents!

<https://openclassrooms.com/fr/courses/1603881-apprenez-a-creer-votre-site-web-avec-html5-et-css3/3298561-faites-votre-mise-en-page-avec-flexbox>