IS3/IS2A4 - Internet

CM1 - Introduction aux technologies d'internet

Thibault Liétard
Polytech' Lille

2025

Page du cours

http://thibault.lietard.fr/internet.html

Introduction

Au commencement...

Il n'y avait rien.. et puis...

- Alan Turing
- Les premiers ordinateurs (60")
- Les premiers terminaux distants (70")
- Les ordinateurs personnels (80")

Échange de données dans les années 80 ?

- Par support (disquettes, bandes,...)
- En réseau local (Ethernet)

Arrivent les années 90

13 mars 1989
Sir Tim Berners-Lee invente le Web

À quoi s’intéresse t-on ?

Internet

- Principe de base
- Architecture
- Utilisation

WWW

- Navigation
- Langages
- Liens avec d’autres technologies

Internet

Qu’est ce que l’internet ?

(proposition de) Définition
Internet est un réseau de réseaux informatiques hétérogènes et autonomes, basé sur la communication par paquet et l’accès distant.

Un peu d'histoire

Naissance d'internet
Internet voit le jour dans les laboratoires de la Defense Advanced Research Projects Agency (DARPA), organisme de recherche pour la défense américaine, au début des années 60. C'est d'abord une recherche théorique qui se concrétise en 1969 avec ARPANET.

Fin des 60"
La première connexion internet se fait entre l'université de Californie à Los Angeles (UCLA) et l'université de Standford.

Les 70"
ARPANET se développe aux USA, Cyclades se développe en France. Naissance de TCP/IP, des e-mails, des communications radio et satellites.

Les 80"
Dans les années 1980, l'ARPA et le CERN finissent par interconnecter leurs réseaux en acceptant les deux types de communications, FTP et HTTP, ce qui facilite la communication entre scientifiques. ARPANET devient également accessible au public.

Les 90"
Invention du WWW, on y reviendra =)

Qu'est-ce qu'un protocole ?

Définition
Un protocole est un « langage » de communication utilisé par deux ordinateurs pour communiquer entre eux.

Exemples

- Réseaux physiques (Ethernet,...)
- Réseaux virtuels (modem, téléphone,...)
- Systèmes de communication par paquets (IP, GSM,...)

Les couches réseaux

Les protocoles (wikipédia)

7	**Application**	par exemple : HTTP, HTTPS, Gopher, SMTP, SNMP, FTP, Telnet, NFS, NNTP
6	**Présentation**	par exemple : ASCII, Unicode, MIME, XDR, ASN.1, SMB, AFP
5	**Session**	ex. ISO 8327 / CCITT X.225, RPC, Netbios, ASP
4	**Transport**	par exemple : TCP, UDP, SCTP, SPX, ATP
3	**Réseau**	par exemple : IP (IPv4 ou IPv6), ICMP, IGMP, X.25, CLNP, ARP, RARP, OSPF, RIP, IPX, DDP
2	**Liaison**	par exemple : Ethernet, Token Ring, PPP, HDLC, Frame relay, RNIS (ISDN), ATM, Wi-Fi, Bluetooth, ZigBee, irDA (Infrared Data Association)
1	**Physique**	par exemple : techniques de codage du signal (électronique, radio, laser, etc.) pour la transmission des informations sur les réseaux physiques (réseaux filaires, optiques, radioélectriques ...)

À qui communiquer ?

Unicast

- Un émetteur
- Un récepteur

Multicast

- Un émetteur
- Plusieurs récepteurs

Anycast

- Un émetteur
- Un récepteur parmi N

Des protocoles fiables... ou pas

Protocoles non fiables

- Réémission et réordonnancement à la charge de l'utilisateur
- Meilleure utilisation d'un réseau fiable et disponible
- Exemples : IP, UDP,...

Protocoles fiables

- Réémission et réordonnancement automatiques
- Surcoût systématique au niveau du réseau
- Exemple : TCP

UDP et TCP

UDP (User Datagram Protocol)

- protocole de transfert de données asynchrone
- a l'avantage d'être peu verbeux mais ne donne aucune garantie sur l'ordre d'arrivée ou le bon acheminement du paquet

TCP (Transmission Control Protocol)

- protocole synchrone de transfert de données qui nécessite la validation d'une connexion et l'envoi d'accusés de réception lorsque les données sont acheminées
- verbeux

Deux ordinateurs discutent.

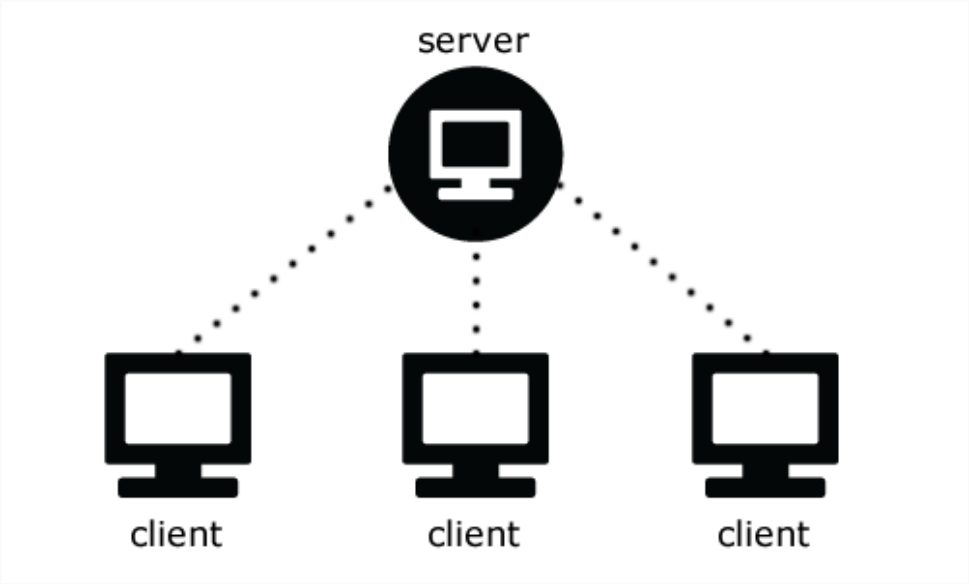
Le premier dit "Je sais oui "

Le deuxième dit "Tu savais que les paquets UDP n’arrivent pas forcément dans l’ordre ?"

Architecture

client-serveur

Définition
Le paradigme client–serveur désigne un mode de transaction (souvent à travers un réseau) entre plusieurs programmes ou processus : l'un, qualifié de client, envoie des requêtes ; l'autre, qualifié de serveur, attend les requêtes des clients et y répond.



Quels clients ?

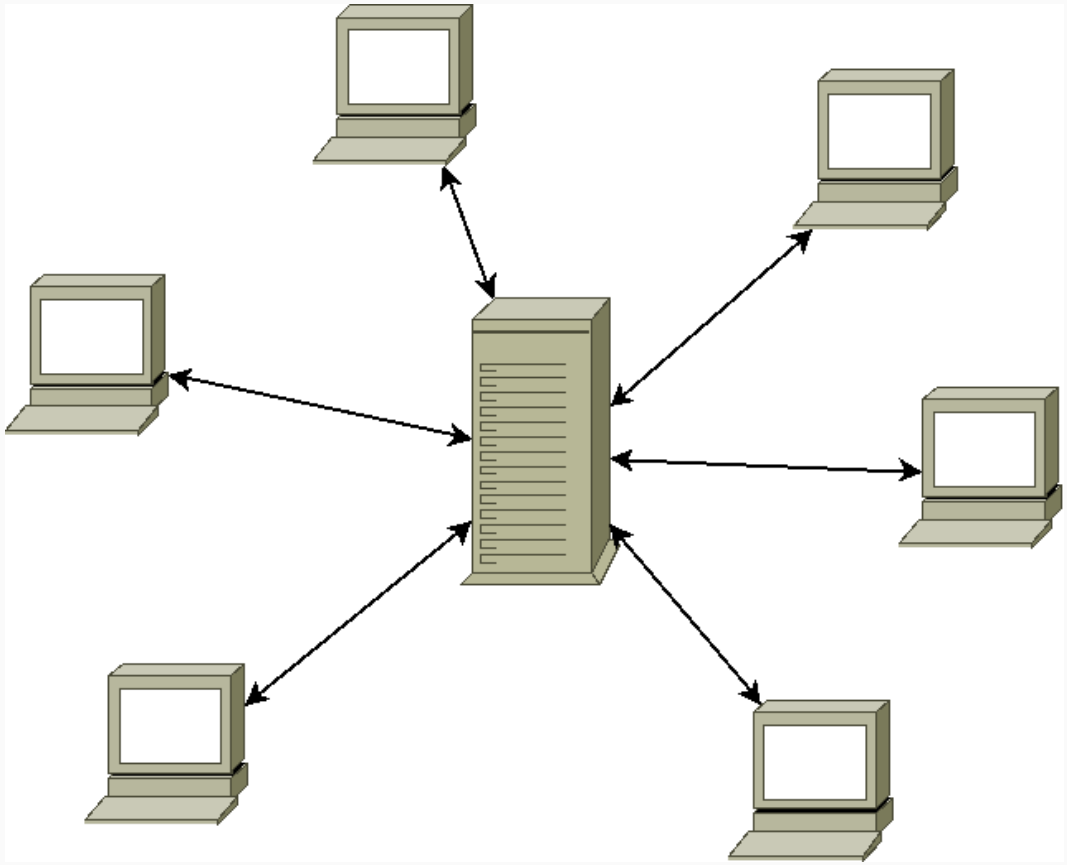
Client léger
Un client léger est une application où le traitement des requêtes du client est entièrement effectué par le serveur, le client se contente de recevoir et mettre en forme pour afficher les réponses calculées et envoyées par les serveur.

Client lourd
Un client lourd est une application (applications de bureau, applications mobile) où les traitements sont principalement effectués sur la machine locale dite cliente. Le serveur se contentant principalement de répondre aux demandes de données du client.

Client riche
Un client riche est une application où le traitement des requêtes du client est effectué majoritairement par le serveur, le client recevant les réponses « semi-finies » et les finalisant. C'est un client léger plus évolué permettant de mettre en œuvre des fonctionnalités comparables à celles d'un client lourd.

Pour aboutir à...

Architectures centralisées
Un seul système auquel se connectent plusieurs postes.



Pourquoi la centralisation ?

Avantages

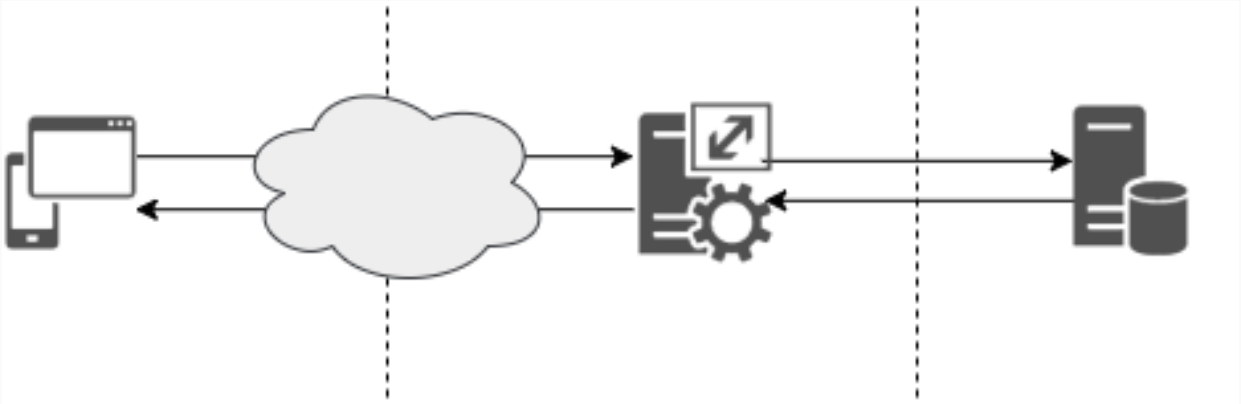
- Sécurité plus facile à gérer
- Coûts d'administration faible par utilisateur
- Maintenance (installation, MàJ,..) aisée

Inconvénients

- Interface utilisateur difficile à gérer
- Faible autonomie de l'utilisateur
- Systèmes propriétaires
- Difficile de migrer vers un autre Système
- Protection des données ?

Architecture trois-tiers

Une architecture 3-tiers est une architecture dans lequel un middleware se charge de la communication entre le client et le serveur.



Avantages et inconvénients

Avantages

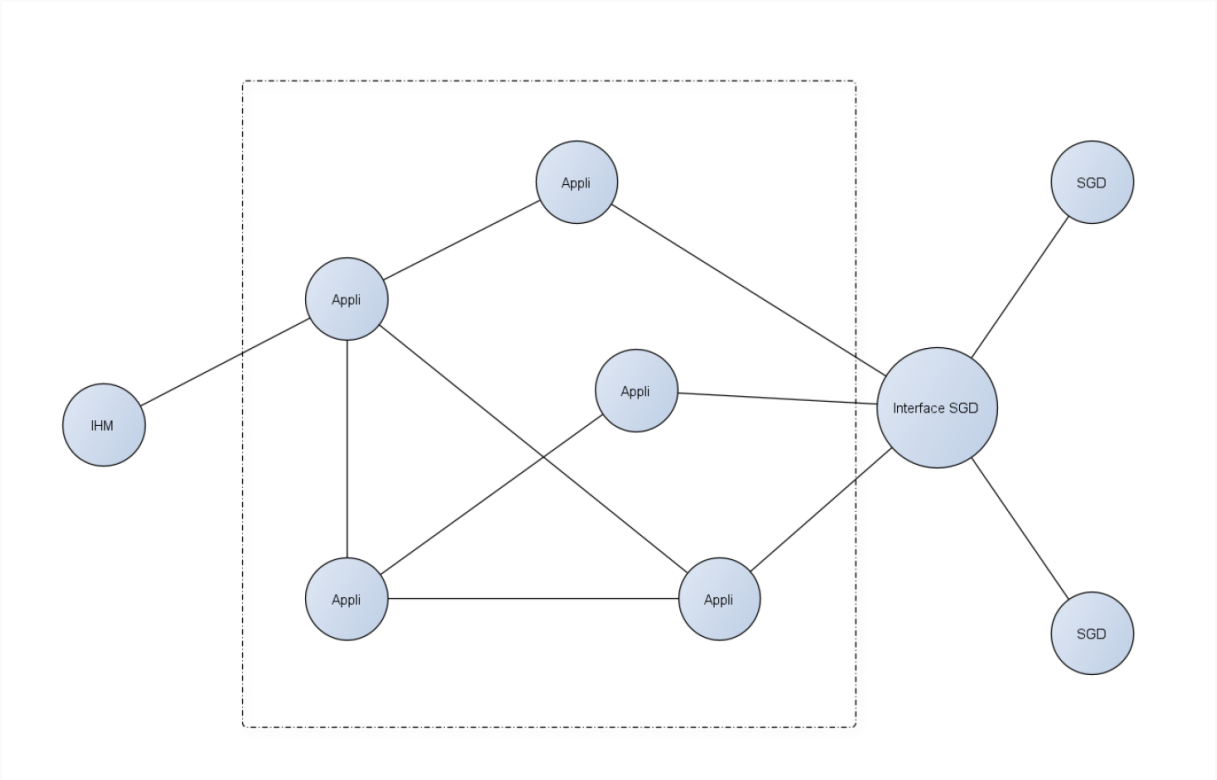
- Réduction du trafic
- les applications serveur sont indépendantes des IHM
- Total transparence à la localisation
- Passage à l'échelle beaucoup plus facile
- Transparence à l'hétérogénéité
- Très bonne sécurité, car l'utilisateur n'accède jamais directement au serveur

Inconvénients

- Mise en place plus lourde qu'un Client-Serveur classique.

Architecture multi-tiers

Une architecture multi-tiers est une extension de l'architecture trois tiers à un ensemble de middleware fournissant chacun des services distincts et indépendants des autres applications. Souvent, le serveur final est simplement un serveur de données.



Et si on décentralisait ?

Notion de système réparti
Un système réparti est un ensemble de machines autonomes et distinctes connectés par un réseau et équipé d'un logiciel dédié à la coordination des activités du système et au partage de ses ressources.

Pour le dire autrement
Un système réparti se compose de :

- plusieurs systèmes calculatoires autonomes (sinon, non réparti)
- sans mémoire physique commune (sinon c'est un système parallèle)
- qui communiquent par l'intermédiaire d'un réseau

Pourquoi répartir ?

- **Accès distant**: un même service peut être utilisé par plusieurs acteurs, situés à des endroits différents
- **Redondance**: des systèmes redondants permettent de pallier une faute matérielle, ou de choisir le service équivalent avec le temps de réponse le plus court
- **Performance**: la mise en commun de plusieurs unités de calcul permet d'effectuer des calculs parallélisables en des temps plus courts
- **Confidentialité**: les données brutes ne sont pas disponibles partout au même moment, seules certaines vues sont exportées

Exemples d'architectures distribuées

- WWW
- Base de données distribuées
- Mail
- Smartphone
- Guichets d'une gare
- ...

Que penser des systèmes décentralisés ?

Avantages

- Répartition des traitements
- Interface utilisateur plus ergonomique
- Très grande autonomie de l'utilisateur
- Meilleur performance globale
- Données plus accessibles en cas de défaillance réseaux

Inconvénients

- Coût d'administration plus élevé par utilisateur
- Faible maîtrise des effets d'échelle
- Déploiement des applications plus compliquée
- Gestion des configurations clients plus problématique

le World Wide Web

Quoi qu'est-ce ?

Wikipédia
Le World Wide Web (littéralement la « toile (d'araignée) mondiale », abrégé www ou le Web), est un système hypertexte public fonctionnant sur Internet.

Attention
On l'a déjà dit, mais le web est une *application* d'internet. Les deux ne peuvent pas être confondus. Internet est une technologie, le web est un paradigme implémenté grâce à cette technologie.

Hypertexte ?

Définition
Un hypertexte est un document ou un ensemble de documents contenant des unités d'information liées entre elles par des hyperliens.

Hyperlien ?
Un hyperlien ou lien hypertexte, est une référence dans un système hypertexte permettant de passer automatiquement d'un document consulté à un autre document. Il a été inventé par Ted Nelson en 1965.

Principe du web

Mathématiquement
Le World Wide Web, en tant qu'ensemble de ressources hypertextes, est modélisable en graphe orienté possédant des cycles avec les ressources pour sommets et les hyperliens pour arcs. Comme le graphe est orienté, certaines ressources peuvent constituer des puits : il n'existe aucun chemin vers le reste du web. À l'inverse, certaines ressources peuvent constituer des sources : il n'existe aucun chemin depuis le reste du web.

Qu’est-ce qu’un site ?

Wikipédia
Un site web est un ensemble de **pages web** et de ressources reliées par des hyperliens, défini et accessible par une **adresse web**. Un site est développé à l’aide de **langages de programmation web**, puis hébergé sur un **serveur web** accessible via le réseau mondial **Internet**

Le premier site web (version du 13 novembre 1990)
`http://info.cern.ch/hypertext/WWW/TheProject.html`

Comment parcourir la toile ?

Le navigateur
Il s'agit purement et simplement d'un client HTTP (à la base). Il permet de résoudre une adresse et d'afficher la réponse du serveur à une requête GET HTTP.

Pour l'anecdote...
Le terme « navigateur » est inspiré de Netscape Navigator.

Le premier navigateur

WorldWideWeb, en ligne de commande

https://line-mode.cern.ch/

Moteur de recherche

Wikipédia
Un moteur de recherche est une application web permettant à un utilisateur d'effectuer une recherche en ligne (ou recherche internet), c'est-à-dire de trouver des ressources à partir d'une requête composée de termes.

Pour le dire simplement
Il s'agit basiquement d'un site web qui permet d'indexer le contenu d'autres sites web à partir d'éléments clefs.

Le protocole HTTP

Définiton (wikipédia)
L'**Hypertext Transfer Protocol** (protocole de transfert hypertexte) est un protocole de communication client-serveur développé pour le World Wide Web. HTTPS (avec S pour *secure*) est la variante sécurisée par le chiffrement et l'authentification.

Aspect réseau
HTTP nécessite une connexion fiable, on utilise donc TCP pour la couche transport. En général, un serveur HTTP utilise le port 80.

Fonctionnement d'un serveur HTTP

Principe
Un serveur HTTP comprend et répond à un ensemble de requêtes appelées ici *méthodes*

Méthodes courantes

- GET :Pour demander une ressource. Une requête GET est sans effet sur la ressource
- HEAD : ne demande que des informations sur la ressource, sans demander la ressource elle-même
- POST : transmet des données pour traitement par une ressource (le plus souvent depuis un formulaire HTML). Le résultat peut être la création de nouvelles ressources ou la modification de ressources existantes
- PUT : permet de remplacer ou d'ajouter une ressource sur le serveur
- DELETE : permet de supprimer une ressource du serveur.

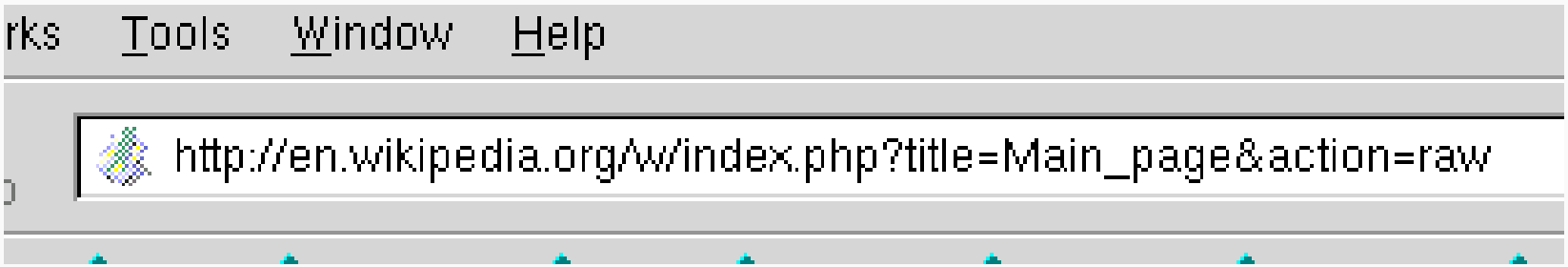
Réponse d'un serveur HTTP

Cas général
En principe, un serveur HTTP répond avec la ressource demandée, ou avec un code spécifiant que tout s'est bien passé. Cependant, il peut parfois répondre avec un code d'erreur.

Codes d'erreur courants

- 200 : succès de la requête ;
- 301 et 302 : redirection, respectivement permanente et temporaire ;
- 401 : utilisateur non authentifié ;
- 403 : accès refusé ;
- 404 : ressource non trouvée ;
- 500, 502 et 503 : erreurs serveur ;
- 504 : le serveur n'a pas répondu.
- 418 : "je suis une théière"

Notion d'URL


rks Tools Window Help
http://en.wikipedia.org/w/index.php?title=Main_page&action=raw

Les langages du web

Création de contenu
HTML : HTML signifie « HyperText Markup Language » qu'on peut traduire par « langage de balises pour l'hypertexte ». Il est utilisé afin de créer et de représenter le contenu d'une page web et sa structure.

Mise en forme
Les CSS (Cascading Style Sheets en anglais, ou « feuilles de style en cascade ») sont le code utilisé pour mettre en forme une page web.

Le dynamisme

Page web dynamique
Une page web dynamique est une page web générée à la demande, son contenu peut donc varier en fonction d'informations (heure, nom de l'utilisateur, formulaire rempli par l'utilisateur, etc.) qui ne sont connues qu'au moment de sa consultation. À l'inverse, le contenu d'une page web statique est a priori identique à chaque consultation.

Les langages

- Javascript
- PHP (PHP : Hypertext Protocol)

Rappels d’HTML

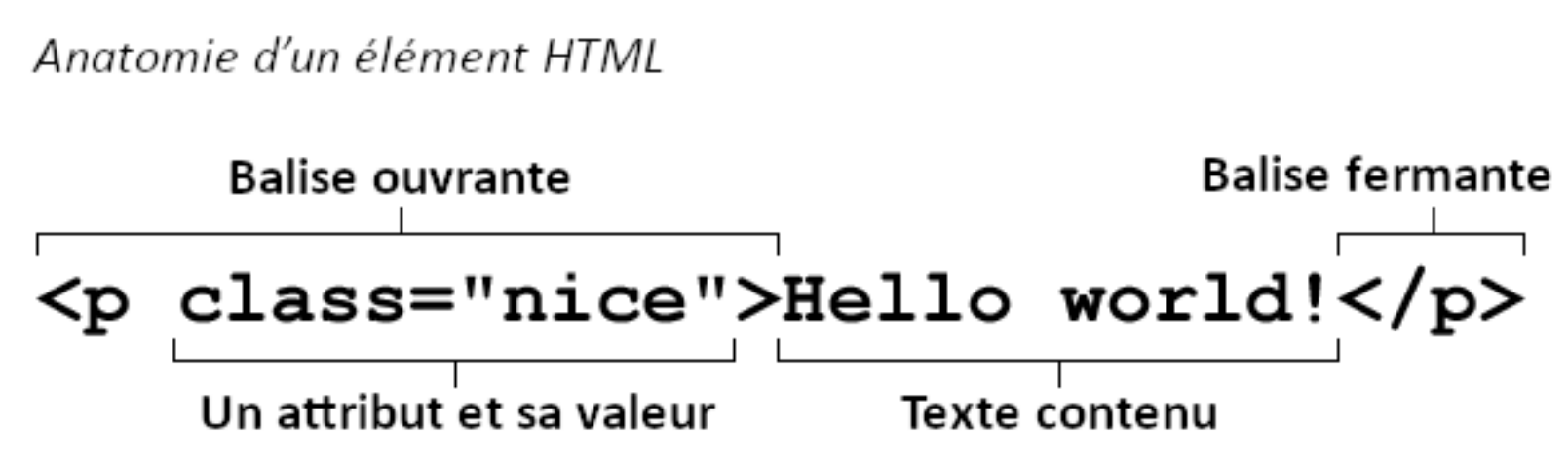
Sources

- https://developer.mozilla.org
- https://developer.mozilla.org/fr/docs/Learn/Getting_started_with_the_web/HTML_basics
- https://developer.mozilla.org/fr/docs/Learn/HTML

Définition

Wikipédia
Le HyperText Markup Language, généralement abrégé HTML ou, dans sa dernière version, HTML5, est le langage de balisage conçu pour représenter les pages web. HTML est inspiré du Standard Generalized Markup Language (SGML). Il s'agit d'un format ouvert.

Fonctionnement



Fonctionnement 2

Type de fichier
Un fichier HTML est un fichier texte, généralement enregistré avec une extension '.html' ; on trouve parfois l'extension '.htm', qui découle simplement des limitations à 3 caractères des anciennes version de windows.

Exécution
Un fichier HTML est mis à disposition par un serveur web (distant ou local), et le rendu est réalisé par un navigateur Web, comme nous avons pu le voir dans les cours précédents.

Rappel des balises classiques

Balises de texte

- : paragraphes
- : liste non ordonnée
- : liste ordonnée
- : item de liste (imbriqué dans <ul> ou <ol> donc)

Balises de titres

- <h1></h1> : Titre de niveau 1
- <h2></h2> : Titre de niveau 2
- <h3></h3> : Titre de niveau 3
- <h4></h4> : ...

Structure d'un bloc HTML

Exemple

```
1 <h1>Mon titre 1</h1>
2 <p>Exemple de paragraphe</p>
3 <ul>
4     <li>item 1</li>
5     <li>item 2</li>
6 </ul>
```

Liens

$< a >< /a >$
Avec des attributs :

- href : donne la cible du lien
- title : permet de donner d'autres informations sur le lien au survol

Exemple
le lien

Anatomie d'un document HTML

Page complète

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ma page test</title>
  </head>
  <body>
    <p>Voici ma page web</p>
  </body>
</html>
```

Éléments d'une page HTML

Structure de la page

1. **Doctype** : le type de document. Héritage des anciennes version de HTML. <!DOCTYPE html> est la plus courte spécification qui fonctionne.
2. **html** : cet élément est le contenant de tout le code de la page et est parfois connu comme l'élément racine.
3. **head** : Cet élément a le rôle de conteneur pour toute chose que vous souhaitez inclure dans la page HTML qui ne soit pas du contenu à afficher : mots clés, description de page pour les résultats de recherche, style CSS, déclarations de jeu de caractères...
4. **body** : Il contient tout le contenu que vous souhaitez afficher aux internautes lorsqu'ils visitent votre page

Éléments d'une page HTML 2

Que trouver à l'intérieur ?

- **head**:
  1. <meta charset=¨ utf-8"> : cet élément définit que le jeu de caractères à utiliser pour votre document, en l'occurrence :UTF-8
  2. <title></title> : définit le titre de la page, celui qui s'affiche dans l'onglet du navigateur dans lequel la page est chargée et qui est utilisé pour décrire la page lorsque vous la marquez ou l'ajoutez aux favoris.
- **body** : toutes les balises de contenu que nous avons vu précédemment : texte, titre, paragraphes, liens, etc...

Les balises orphelines

Définitions
Certaines balises n'ont pas de balises fermantes : elles sont appelées balises orphelines.

Exemples

-
 : permet de passer une ligne (saut de ligne forcé)
- <hr/> : permet de tirer un trait horizontal

Quoi qu'est-ce ?
HTML5 (HyperText Markup Language 5) est la dernière révision majeure du HTML (format de données conçu pour représenter les pages web). Cette version a été finalisée le 28 octobre 2014.

Le travail a été repris par le W3C en mars 2007 après avoir été lancé par le WHATWG. Les deux organisations travaillent en parallèle sur le même document afin de maintenir une version unique de la technologie. Le W3C clôt les ajouts de fonctionnalités le 22 mai 2011, annonçant une finalisation de la spécification en 2014, et encourage les développeurs Web à utiliser HTML 5 dès ce moment. Fin 2016, la version 5.1 est officiellement publiée.

Que fournit HTML5 ?

Plein de choses !
Mais en ce qui nous concerne, on s'intéresse surtout aux nouvelles balises. Elles permettent de définir de nouveaux blocs qui ont tous un sens différent.

Attention
Ces blocs n'auront pas forcément un affichage différent sur une page sans css car les affichages "de baseßont les mêmes. Ce qui compte, c'est d'avoir la possibilité de différencier ces affichages en fonction du **sens**.

Balises HTML5

- **main** : Définit le contenu principal de la page, il doit être unique dans la page.
- **section** : Section générique regroupant un même sujet, une même fonctionnalité, de préférence avec un en-tête, ou bien section d'application web.
- **nav** : Section possédant des liens de navigation principaux (au sein du document ou vers d'autres pages).
- **article** : Section de contenu indépendante, pouvant être extraite individuellement du document ou syndiquée (flux RSS ou équivalent), sans pénaliser sa compréhension.

Balises HTML5 2

- **aside** : Section dont le contenu est un complément par rapport à ce qui l'entoure, qui n'est pas forcément en lien direct avec le contenu mais qui peut apporter des informations supplémentaires.
- **mark** : Définit un texte marqué. Surligneur de texte.
- **header** : Section d'introduction d'un article, d'une autre section ou du document entier (en-tête de page).
- **footer** : Section de conclusion d'une section ou d'un article, voire du document entier (pied de page).

HTML5 : un exemple

```html
<html lang="fr">
    <head>
    </head>
    <body>
        <header>
            <nav>
                <ul>
                    <li><a href="#contenu-principal">contenu principal</a></li>
                </ul>
            </nav>
        </header>
        <main id="contenu-principal">
            <!-- Contenu principal de la page -->
        </main>
    </body>
</html>
```

Attributs d'identification d'éléments

id
Cet attribut définit un identifiant, unique au sein de tout le document, ,pour un élément. Il doit permettre d'identifier un élément lorsqu'on crée un lien vers lui et/ou lorsque le manipule avec des scripts ou avec CSS.

class
Une classe, ou une liste de classes séparées par des espaces, permet de catégoriser l'élément. Les classes permettent au CSS et à JavaScript de manipuler des éléments spécifiques grâce à des sélecteurs de classe.

Création d'autres division sémantiques

div
En tant que conteneur « pur », l'élément ne représente rien en soi. Il est plutôt utilisé pour regrouper le contenu afin qu'il puisse être facilement stylé à l'aide des attributs class ou id

span
identique à div mais en ligne plutôt qu'en bloc.

Notations

Attention
 et sont des **balises** qui servent à créer des éléments sémantiques, *id* et *class* sont des **attributs** qui servent à les identifier.

Exemple

- le bloc de ma division
- la ligne avec span

Spécificité

Attention 2
class et **id** s'utilisent avec quasiment n'importe quelle balise, pas seulement les balises HTML5.

Cela veut dire qu'on peut aussi bien les utilisers avec des balsies de structurations que des balises de texte, de liens, etc...

Images en HTML5

Exemple d'intégration d'image

```
<img src="images/dinosaur.jpg"
     alt="The head and torso of a dinosaur skeleton;
          it has a large head with long sharp teeth">
```

Éléments présents

- La balise <img> (ou <img/ > puisque c'est une balise orpheline) qui permet d'afficher une image
- l'attribut *src* qui donne le chemin d'accès à l'image (ou son url)
- l'attribut *alt* qui propose un texte alternatif à l'image décrivant celle-ci. Très utile de le préciser si l'image ne charge pas ou si un utilisateur déficient visuel utilise le site.
- on peut éventuellement ajouter un attribut *title* qui affichera le titre de l'image au survol avec la souris.

CSS avant l’heure

Attention
Les balises *img* sont des balises de type ïnline", c’est à dire qu’elle ne crée pas de retour à la ligne et ses possibilités graphiques sont limitées. On veillera à toujours la mettre dans une balise bloc adapté pour gérer le CSS par la suite.

Tableaux HTML : structure

Éléments

- <**table**> : balises ouvrantes et fermantes pour définir un tableau.
- <**tr**> : balises ouvrantes et fermantes pour définir une ligne d'un tableau (tr = table row)
- <**td**> : balises ouvrantes et fermantes pour définir une case dans une ligne d'un tableau (td = table data)

Exemple

Code

```
1 <table>
2     <tr>
3         <td>Jean</td><td>Marie</td><td>Salim</td>
4     </tr>
5     <tr>
6         <td>12</td><td>18</td><td>15</td>
7     </tr>
8 </table>
```

Rendu

Jean	Marie	Salim
12	18	15

Élément de bloc

<thead>
Permet de créer un bloc avec les éléments qui sont l'en-tête du tableau. Les lignes sont toujours marquées par la balise <tr> mais les éléments de la ligne utilisent la balise <th> (table head).

<tbody>
Permet de mettre toutes les autres lignes du tableau pour les différencier de l'en-tête. On peut aussi utiliser la balise <th> au sein de l'élément *tbody* pour différencier les première cases des lignes par exemple.

Case étendue

Attributs
On dispose de deux attributs pour étendre une case sur plusieurs lignes ou colonnes :

- **colspan** : pour étendre sur plusieurs colonnes
- **rowspan** : pour étendre sur plusieurs lignes.

Reprise de l'exemple précédent

Code

```
01 <table>
02     <thead>
03         <tr>
04             <th colspan=4>Notes du DS</th>
05         </tr>
06     </thead>
07     <tbody>
08         <tr>
09             <th>Nom</th><td>Jean</td><td>Marie</td><td>Salim</td>
10         </tr>
11         <tr>
12             <th>Note</th><td>12</td><td>18</td><td>15</td>
13         </tr>
14     </tbody>
15 </table>
```

Rendu

Notes du DS			
Nom	Jean	Marie	Salim
Note	12	18	15

Remarque
Ça n'est pas top esthétiquement, mais on fera mieux quand on aura fait du CSS =)

Attention

Ne pas utiliser de tableaux pour
organiser des pages Web, par exemple : une ligne pour contenir l'en-tête, une ligne pour les colonnes de contenu, une ligne pour le pied de page, etc.

Les mises en page sur la base de tableaux comportent généralement des structures de balisage plus complexes que des techniques de mise en page appropriées. Le code résultant sera plus difficile à écrire, à maintenir et à déboguer.

Javascript

Le Marketing
JavaScript est un langage de programmation qui ajoute de l'interactivité à un site web (par exemple : jeux, réponses quand on clique sur un bouton ou des données entrées dans des formulaires, composition dynamique, animations).

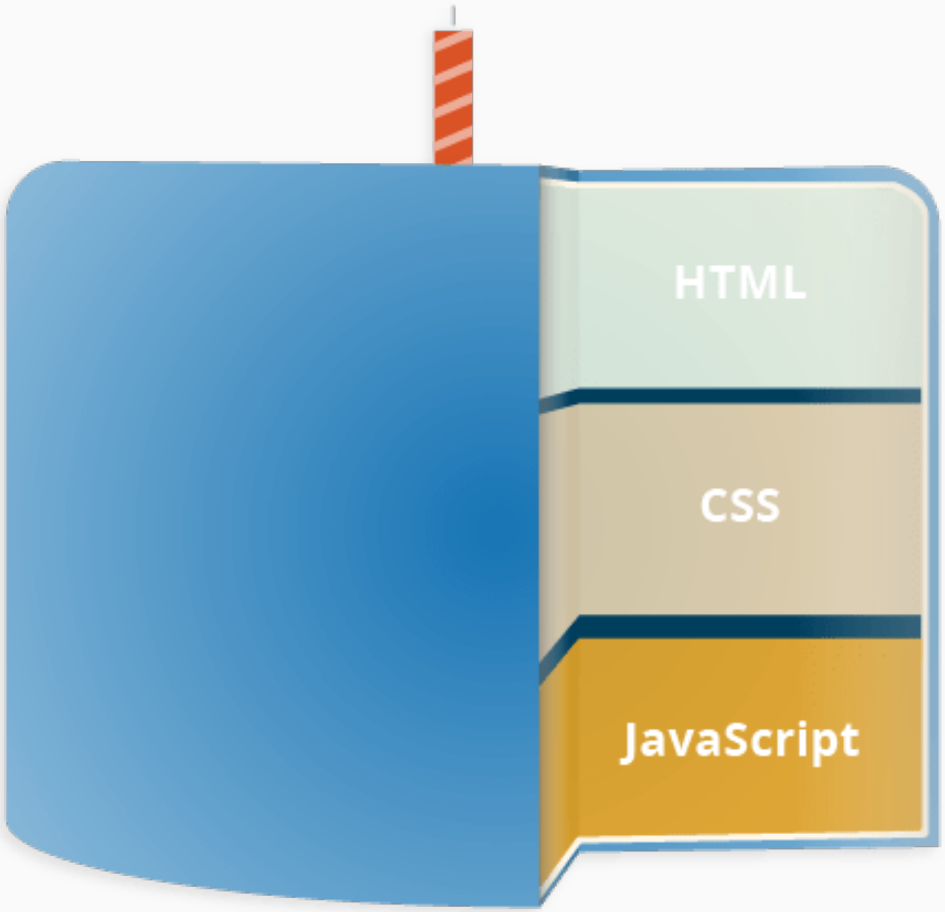
Plus formellement
JavaScript (souvent abrégé en « JS ») est un langage de script léger, orienté objet. Le code JS est interprété ou compilé à la volée. JS dispose d'un typage faible et dynamique qui permet de programmer suivant plusieurs paradigmes de programmation : fonctionnelle, impérative et orientée objet.

Interprété vs compilé

JavaScript est un langage **interprété** : le code est exécuté de haut en bas et le résultat du code exécuté est envoyé immédiatement. Vous n'avez pas à transformer le code en une autre forme avant que le navigateur ne l'exécute.

Les langages **compilés** quant à eux sont transformés en une autre forme avant de pouvoir être exécutés par l'ordinateur. Par exemple le C et le C++ sont compilés en langage assembleur qui est ensuite exécuté par l'ordinateur.

Après HTML et CSS... Javascript



Utilité de Javascript

Ça sert à quoi ?

- Stocker des valeurs dans des variables
- Effectuer des opérations, en particluier sur du texte
- Exécuter du code en réponse à des *événements*
- Utiliser des *API*

Notion d'API

Quoi qu'est-ce ?
Les API sont des blocs de code déjà existants qui peuvent être inclus dans le code Javascript en partie ou totalement. En général, on fait référence à une API pour désigner un ensemble plus ou moins homogène et cohérent de bloc distincts.

Ikea
Les API sont au code ce que les meubles en kits sont à la décoration — Si vous voulez une étagère, vous pouvez travailler vous-même sur le design, aller chercher le bon bois, couper tous les panneaux à la bonne taille et la bonne forme, de trouver les vis de la bonne taille, puis les assembler pour faire une étagère ; ou vous pouvez en acheter une en kit et la monter en 20 minutes. C'est beaucoup plus rapide, mais si vous ne faites pas un peu de DIY, vous avez la même étagère que tout le monde.

Deux types d'API

API de navigateur

- **DOM (Document Object Model)** : permet de manipuler du HTML et du CSS
- **l'API de géolocalisation** : récupère des informations géographiques.
- **Canvas** et **WebGL** : permettent de créer des animations 2D et 3D.
- ...

API tierces
Twitter, Google Maps, Facebook,... quasiment tous les services classiques proposent une API

Exécution

ordre
Le JavaScript est exécuté par le moteur JavaScript du navigateur, après que le HTML et le CSS ont été assemblés et combinés en une page web.

Sécurité
Chaque onglet du navigateur représente un environnement d'exécution différent. Cela signifie que, dans la plupart des cas, le code de chaque onglet est exécuté complètement séparément, et le code d'un onglet ne peut affecter directement le code d'un autre onglet ou d'un autre site.

Ajouter du Javascript

Dans la page

Les balises
Il existe des balises entre lesquelles vous pouvez placer du code javascript :

<script>

code javascript

</script>

Via un script

Charger un fichier .js
Il est possible d'écrire le code javascript dans un fichier d'extension .js et de charger ce fichier depuis a page HTML :

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

Quand charger Javascript ?

Problème
Un problème courant est que le code HTML d'une page se charge en suivant l'ordre d'apparition dans le code source. Si vous utilisez JavaScript pour manipuler des éléments sur la page (plus précisément, le DOM), votre code ne fonctionnera pas si le JavaScript est chargé et analysé avant le code HTML sur lequel vous voulez opérer.

Solution 1

Le script en bas de la page
Une solution à ce problème consiste à placer votre élément de script juste au bas du corps (par exemple, juste avant la balise), afin qu'il soit chargé après que tout le code HTML a été analysé.

Inconvénient
Le chargement et l'analyse du script sont complètement bloqués jusqu'à ce que le DOM HTML soit chargé. Sur des sites plus importants avec beaucoup de JavaScript, cela peut entraîner un problème de performances majeur

Solution 2

Tout mettre dans une fonction

```
document.addEventListener("DOMContentLoaded", function() {
...
});
```

Inconvénient

Même souci que précédemment

Solution 3

async* et *defer

Il s'agit d'attributs qui, au chargement d'un script, vont définir comment doit se faire le chargement de ce dernier par rapport au reste de la page

<script src="script.js" async></script>
<script src="script.js" defer></script>

Fonctionnement

async
Les scripts avec l'attribut *async* téléchargent le script sans bloquer le rendu de la page et l'exécuteront dès que le téléchargement du script sera terminé. Vous n'obtenez aucune garantie que les scripts s'exécutent dans un ordre spécifique, mais seulement qu'ils n'empêcheront pas le reste de la page de s'afficher. Préférable lorsque les scripts de la page s'exécutent indépendamment.

defer
L'attribut *defer* exécute les scripts dans l'ordre dans lequel ils apparaissent dans la page et les exécute dès que le script et le contenu sont téléchargés

Bases du langages

Déclarer une variable

Syntaxe simple
let a = o;

Points importants

- Termine par un ';'
- Pas de déclaration de type
- Peut stocker n'importe quelle valeur objet

Sur les chaînes de caractères

Syntaxe simple
let a = "test";

Mais...

- “2“ - “2“ => 0
- “2“ * “2“ => 4
- “2“ / “2“ => 1
- “2“ + “2“ => “22“... Pourquoi ?

Condition

Syntaxe

```
1- if (condition) {
2-     code à exécuter si la condition est true
3- } else {
4-     sinon exécuter cet autre code à la place
5- }
```

Points importants

- *else* facultatif
- Possible de l'écrire sur une seule ligne si une seule instruction par bloc
- La condition peut être n'importe quoi renvoyant *true* ou *false*

Opérateurs de comparaison

Operateur	Nom
===	Égalité stricte
!==	Non-égalité stricte
<	Inférieur à
>	Supérieur à
<=	Inférieur ou égal à
>=	Supérieur ou égal à

Opérateurs logiques et variables

AND, OR et NOT
Comme en C : **&&** pour AND, **||** pour OR, **!** pour NOT

Une variable comme unité logique
Toute valeur autre que *false*, *undefined*, *null*, 0, *NaN* ou “ “ renvoie *true* lorsqu’elle est testée dans une structure conditionnelle. On peut donc utiliser un if pour savoir si une variable existe (c’est à dire si elle n’est pas *undefined*).

Switch

Syntaxe

```
01- switch (expression) {
02-     case valeur1:
03-         ...
04-         break;
05-     case valeur2:
06-         ...
07-         break;
08-     ...
09-     default:
10-         ...
11- }
```

Boucles

Syntaxe du Pour

```
for (let i = 0; i < 100; i++) {
...
}
```

Syntaxe du Tant Que

```
while (conition) {
...
}
```

Syntaxe du Faire...Tant que

```
do {
...
} while (condition)
```

Interruption de boucle

D'une itération
On utilise le mot clé **continue** pour interrompre l'itération en cours et passer à la suivante.

De toute la boucle
Le mot clé **break** sert à arrêter une boucle avant la fin. Le navigateur passe alors au bloc de code suivant.

Déclarer une fonction

Syntaxe simple

```
01 - function maFonction(param1, param2, ...){
02 -     ...
03 -     return val;
04 - }
```

Points importants

- Pas de déclaration de type
- return facultatif

Array

Syntaxe
let tab = [val1, val2, val3,...]

Fonctions utiles

- tab.length
- tab.push(val)
- tab.indexOf(val)
- ...

Manipulation de page

Exemple

HTML

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Ma page</title>
    </head>
    <body>
        <main>
            <h1>Test pour voir</h1>
            <p>Ceci est un paragraphe</p>
        </main>
        <script src="main.js"></script>
    </body>
</html>
```

Javascript

```javascript
let myHeading = document.querySelector('h1');
myHeading.textContent = 'Bonjour, monde !';
```

Rendu

Sélectionner un élément

querySelector()
Il s'agit d'une méthode de l'interface Document retournant le premier **Element** dans le document correspondant au sélecteur - ou groupe de sélecteurs - spécifié(s), ou *null* si aucune correspondance n'est trouvée.

Syntaxe
element = document.querySelector(sélecteurs);

Problème ?

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Ma page</title>
    </head>
    <body>
        <main>
            <h1>Titre 1</h1>
            <p>Ceci est un paragraphe</p>

            <h1>Test à nouveau</h1>
            <p>Et si je mettais un deuxième paragraphe ?</p>
        </main>
        <script src="main.js"></script>
    </body>
</html>
```

Solution 1

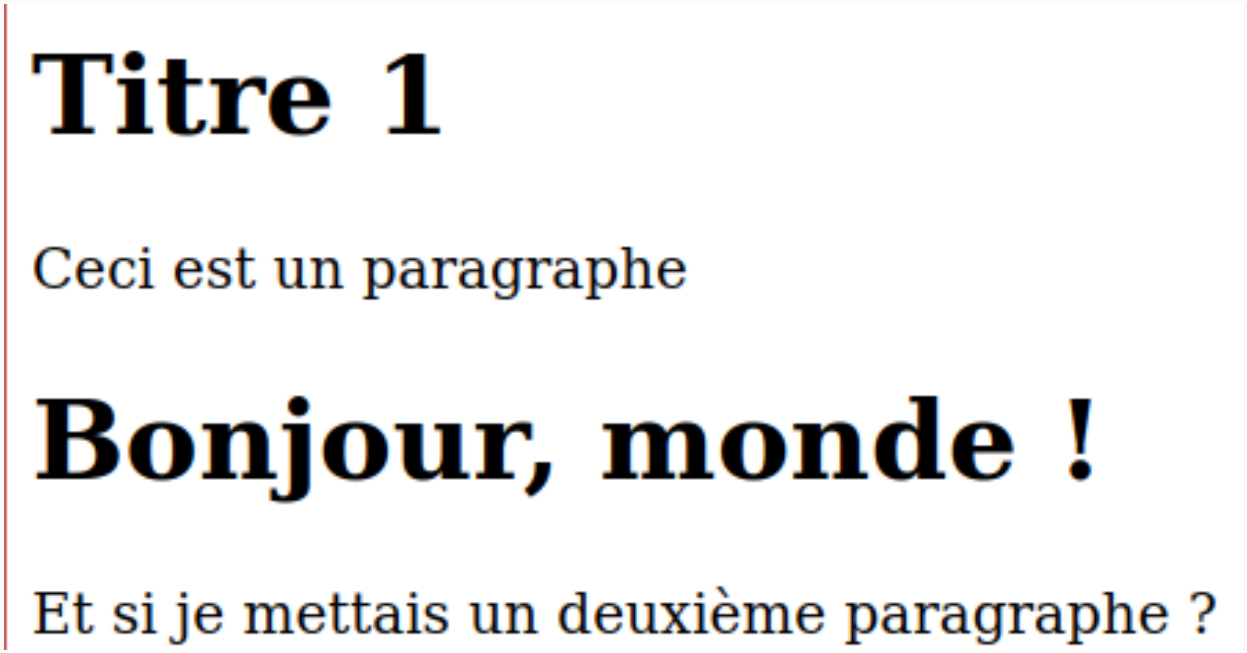
```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Ma page</title>
    </head>
    <body>
        <main>
            <h1>Titre 1</h1>
            <p>Ceci est un paragraphe</p>

            <h1 class="toto">Test à nouveau</h1>
            <p>Et si je mettais un deuxième paragraphe ?</p>
        </main>
        <script src="main.js"></script>
    </body>
</html>
```

Solution 2

Modifier le sélecteur

```
1  let myHeading = document.querySelector('.toto');
2  myHeading.textContent = 'Bonjour, monde !';
```

Rendu



Dynamisme et événements

Fonctionnement

Définition
Les **événements** sont des actions ou des occurrences qui se produisent dans le système que vous programmez et dont le système vous informe afin que vous puissiez y répondre

Principe
Chaque élément disponible a un **gestionnaire d'événement** : un bloc de code qui sera exécuté lorsque l'événement se déclenchera. Lorsqu'un tel bloc de code est défini pour être exécuté en réponse à un déclenchement d'événement, nous disons que nous **enregistrons** un gestionnaire d'événements (*EventListner* en anglais).

Différents événements

Exemple

- focus
- blur
- click
- keydown
- ...

Liste complète
`https://developer.mozilla.org/fr/docs/Web/Events`

Un exemple

HTML

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Ma page</title>
    </head>
    <body>
        <main>
            <h1>Exemple 1</h1>
            <button>Appuyez</button>
        </main>
        <script src="main.js"></script>
    </body>
</html>
```

Un exemple

Javascript

```
1  var btn = document.querySelector('button');
2
3  btn.onclick = function() {
4    alert("vous avez appuyez !");
5  }
```

Un exemple

Rendu

Un exemple

Action

Un autre exemple

HTML

```html
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>Ma page</title>
    </head>
    <body>
        <main>
            <h1>Exemple 2</h1>
            <button>Change color</button>

        </main>
        <script src="main.js"></script>
    </body>
</html>
```

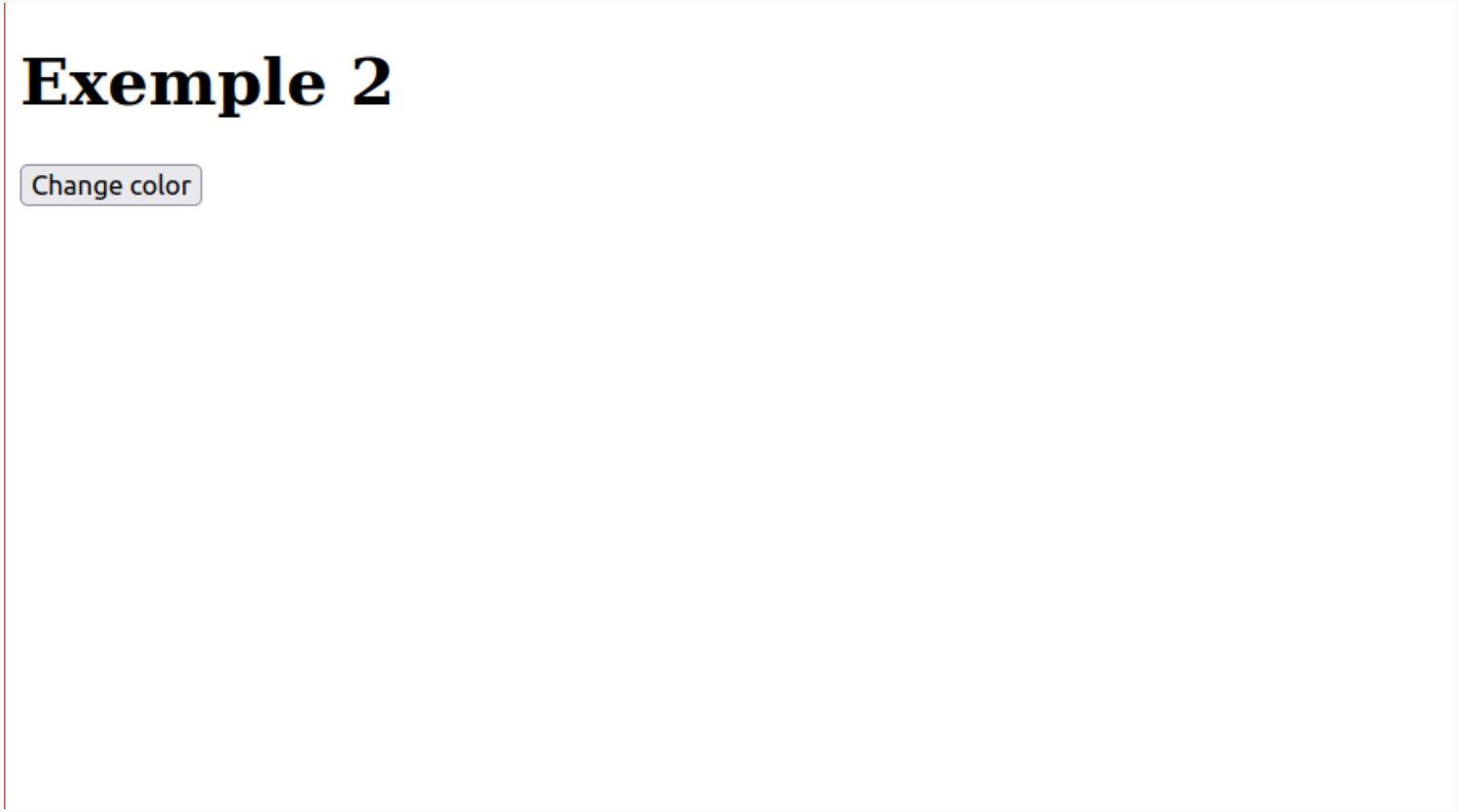
Un autre exemple

Javascript

```
1  var btn = document.querySelector('button');
2
3  function random(number) {
4    return Math.floor(Math.random()*(number+1));
5  }
6
7  btn.onclick = function() {
8    var rndCol = 'rgb(' + random(255) + ',' + random(255) + ',' + random(255) + ')';
9    document.body.style.backgroundColor = rndCol;
10 }
```

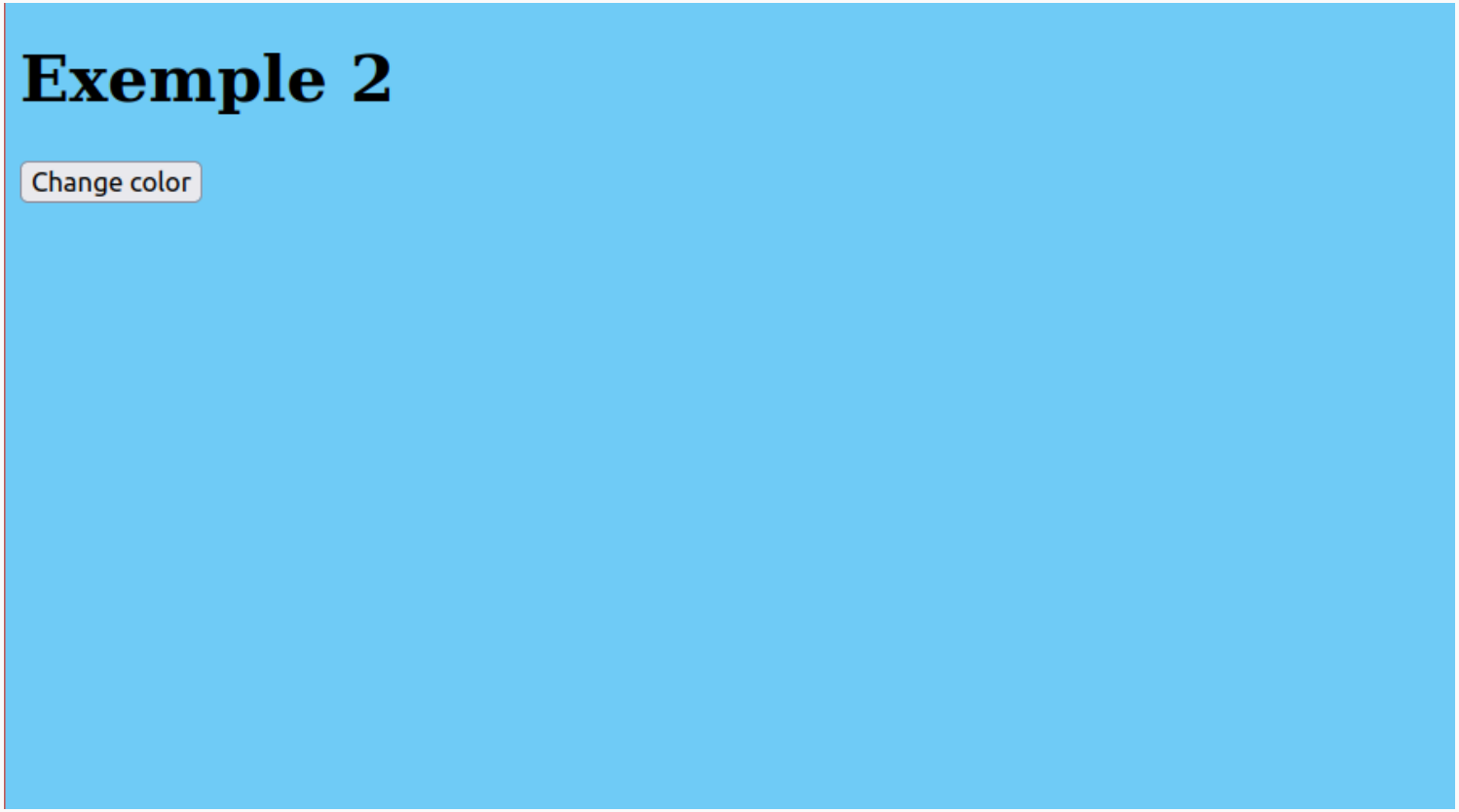
Un autre exemple

Rendu



Un autre exemple

Action



La fonction *alert()*

Fonctionnement
Permet d'afficher une fenêtre popup. Prend en paramètre la chaîne de caractères à afficher dans la boîte de message qui s'ouvrira.

Boîte de dialogue : *prompt()*
Une boîte de dialogue permet à l'utilisateur d'entrer des informations. La fonction *prompt* fonctionne exactement comme *alert()* mais fourni également une zone de saisie dont elle retourne la valeur sous la forme de String.

L'objet *style*

Dans le CSS

Il est possible de modifier directement les propriétés CSS d'un élément en accédant à son objet *style*. N'importe quelle propriété existant de façon standard en CSS est ainsi modifiable. C'est ce qui est fait dans la ligne suivante :

```
document.body.style.backgroundColor = rgb(0,0,0);
```

L'élément *button*

Définition
L'élément <*button*> représente un bouton cliquable, utilisé pour soumettre des formulaires ou n'importe où dans un document pour une fonctionnalité de bouton accessible et standard.

La variable *onclick*

Gestionnaire d'événement en cas de clic
La plupart des éléments HTML contenus dans le document possèdent en javascript une variable önclick"permettant de définir quelle fonction sera exécutée en cas de clic.

Pour le reste...
Il existe d'autres types de variables pour d'autres types d'événements :

- onfocus / onblur
- onmouseover / onmouseout
- ondbclick
- ...

Ajout de gestionnaires

addEventListener

C'est une méthode de n'importe quel objet du document pouvant posséder un gestionnaire d'événement. Dans la fonction *addEventListener()* , nous spécifions deux paramètres - le nom de l'événement pour lequel nous voulons enregistrer ce gestionnaire, et le code qui comprend la fonction du gestionnaire que nous voulons exécuter en réponse.

Variation de l'exemple 2

```
var btn = document.querySelector('button');

function bgChange() {
  var rndCol = 'rgb(' + random(255) + ',' + random(255) + ',' + random(255) + ')';
  document.body.style.backgroundColor = rndCol;
}

btn.addEventListener('click', bgChange);
```

Supprimer un gestionnaire

removeEventListener
Même propriétés que la précédente. Permet simplement de supprimer un gestionnaire d'événement pour les cas où cela serait nécessaire.

Pour en savoir plus

Sur les événements :
`https://developer.mozilla.org/fr/docs/Learn/JavaScript/Building_blocks/Events`

Sur Javascript :
`https://developer.mozilla.org/fr/docs/Learn/JavaScript`

This is it ☺