Les pages & application web

en 2025

Le format HTML
La mise en forme avec CSS3 et
L’interaction avec Javascript

Organisation du cours / TD

- **Session de 30 minutes pour maitriser un concept puis exercise de prise en main**
- **Objectif: Etre en mesure de réaliser une mise en forme au format html.**
- **Projet directeur: Créer son cv ou ses savoir faire en html**
- **Projet en TP: Creation d'un quizz qui evalue les reponses**

Pre-requis

- **Disposer d’un PC avec un editeur de code (notepad++, visual studio, …) et un navigateur à jour (Chrome de preference)**
- **Connaître le fonctionnement d’un PC ce qu’est un fichier texte, un logiciel.**
- **Aucun pre-requis sur le html css ou javascript**
- **Les ressources utiles :**
  - W3Schools: https://www.w3schools.com/: https://www.w3schools.com/
  - Mozilla Developer Network: https://developer.mozilla.org/en-US/: https://developer.mozilla.org/en-US/
  - Codecademy: https://www.codecademy.com/: https://www.codecademy.com/
  - tailwindcss : https://tailwindui.com/documentation
  - DaisyUI : https://daisyui.com/components/

Sur 4 jours

Jour 1

HTML5 (CM: 2h)

Introduction au HTML et le protocole HTTP

Les balises/ Attributs / Elements / Tableaux /Formulaires HTML

CSS3 (CM: 2h)

Introduction au CSS

Les sélecteurs / propriétés / positionnement / bordure / police / Arrière plan CSS

HTML/CSS (TP: 3h)

Mise en page d'un CV ou d'un service proposé en html

Mise en page d'un questionnaire pour évaluer si vous voulez donner des informations personnel (email, photo,...) à la personne qui vous visite.

La combinatoire des reponses = le nom d'une page web à afficher.

Jour 2

JavaScript (CM: 4h)

Introduction au JavaScript

Les variables / opérateurs / conditions / boucles / fonctions / événements / animations

JS (TP: 3h)

Modeliser un questionnaire en format JSON
Afficher chaque question une par une et calculer un score en fonction des réponses créer une redirection vers une page contenant les bonnes reponses dans son nom.html. Dereferencé la page des moteurs de recheche

Jour 3

Les outils du dev web (CM:3h)

Introduction à Git, gestion de version en équipe
Creation d'un compte github public

Les Framework css (Bootstrap, Tailwindcss,..)
Les Framework js (vue.js, react, angular,...)

Tailwindcss et daisyUI (CM:1h TP:3h)

Introduction à node.sj
Introduction tailwindcss/daisyUI avec le client

Les composants / interactions

Etude de cas pratique d'interaction avec les composants existant

Jour 4

Projet (TP:4h) Realiser un projet git avec l'environnement node.js et Tailwindcss/DaisyUI en ligne de commande.

Une page de présentation de son CV ou sur un autre sujet. Un menu déroulant permettant de rentrer en contact avec le créateur de la page.
Pour prendre contact faire dérouler un questionnaire qui recécupére les réponses une par une. Si la combinaison des réponses sont bonnes cela donne l'accès à une page html avec les coordonnées précises de la personne (email numéro de téléphone nom prénom)

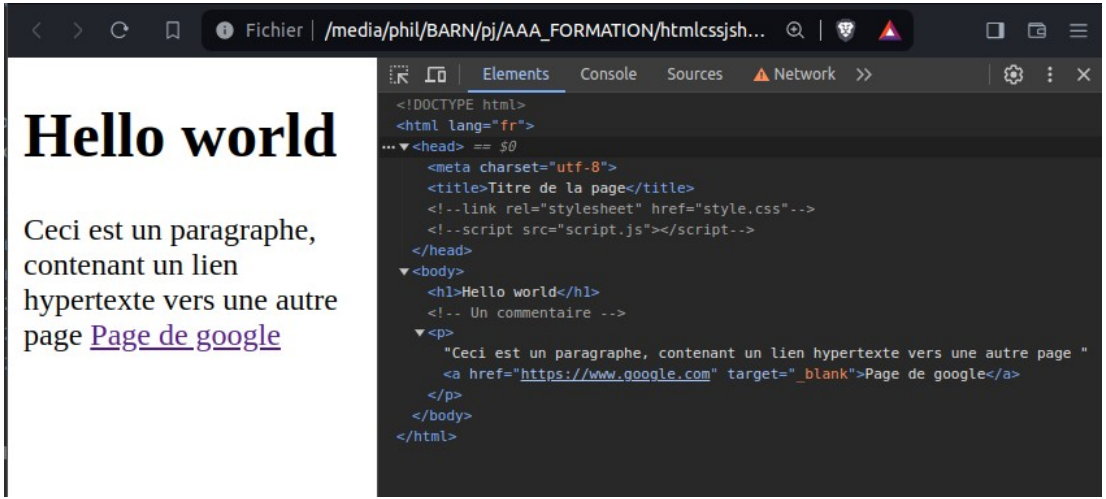
Evaluation:Le projet github sera évalué

1. HTML 5 (2h)

Introduction

HTML HyperText Markup Language

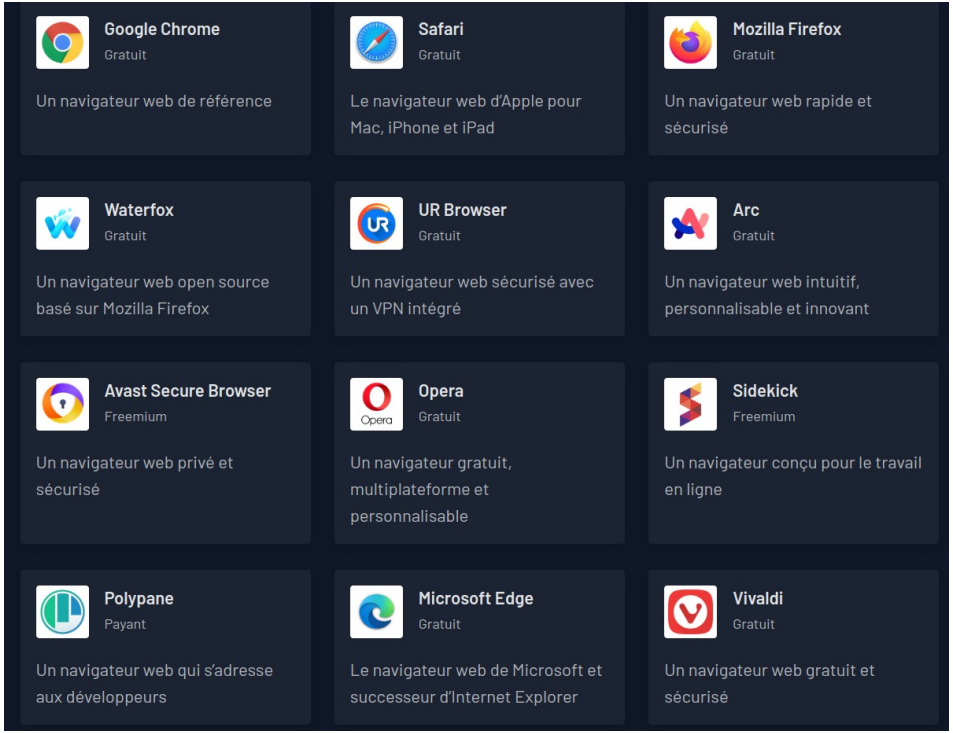
- **HTML n’est pas :**
  - Un langage de programmation.
  - Un langage de mise en forme de document
- **HTML est :**
  - Une organisation de contenu texte dans une page web
  - Une indexation d’un ensemble de pages web
  - Une déclinaison du XML qui est une représentation de données plus complexe



Introduction

HTML HyperText Markup Language

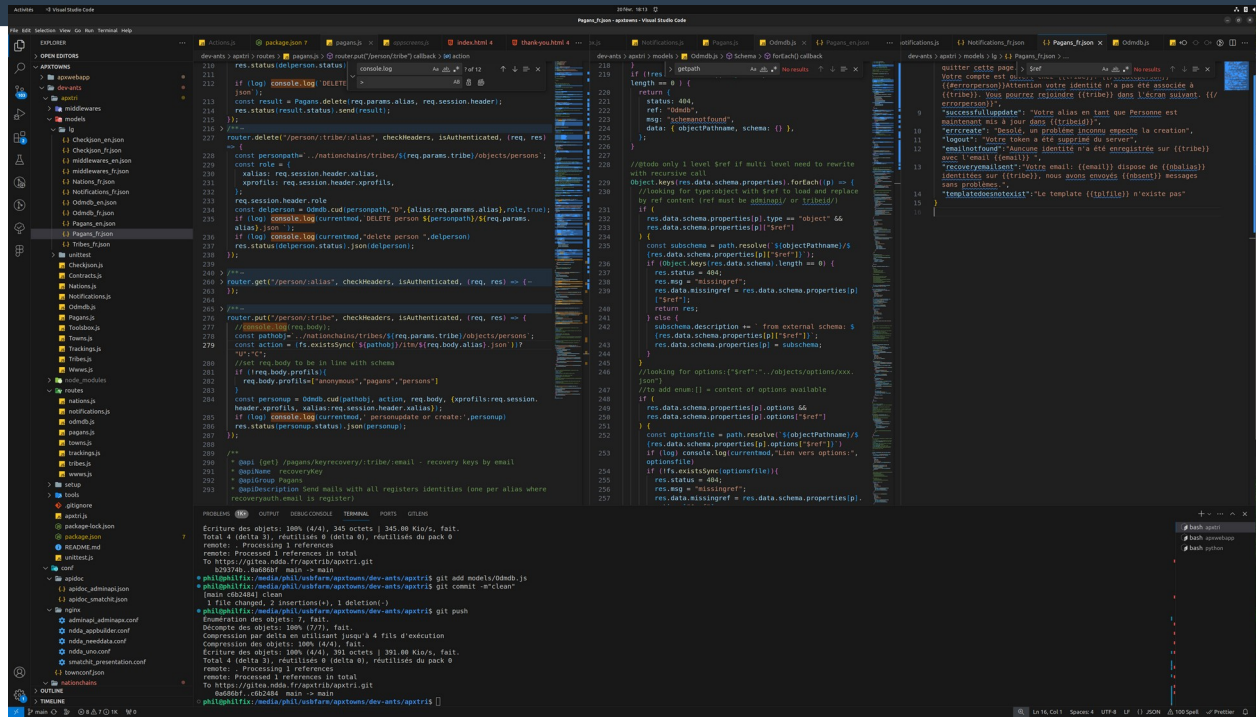
- **HTML est un fichier texte.**
  - Il a un encodage
  - Il a une langue
  - Il est composé de balises caractérisées par des attributs qui permettent à des logiciels appelés navigateur d'interpréter son contenu et de le mettre en forme.



Introduction

HTML HyperText Markup Language

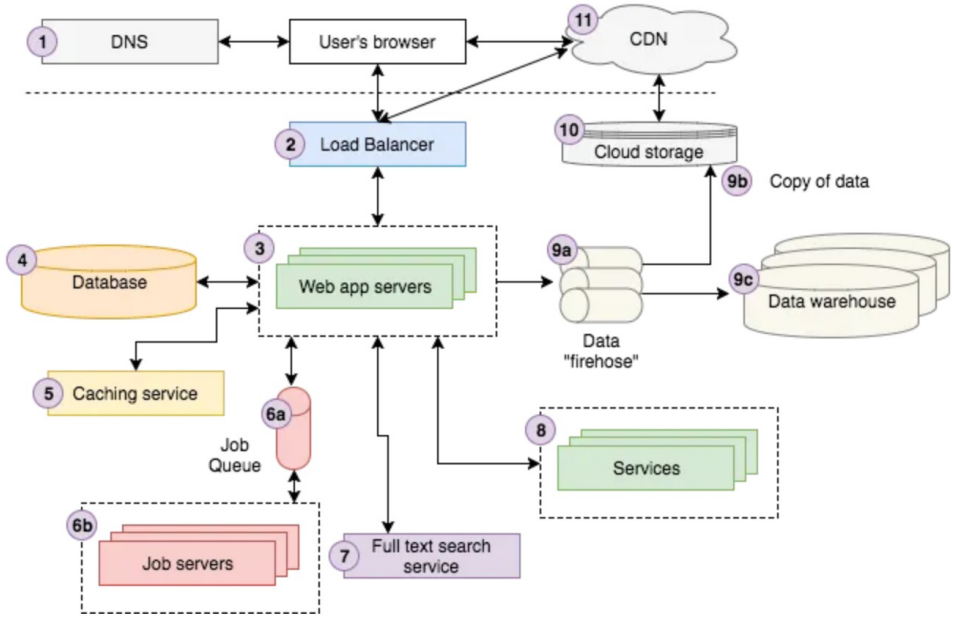
- **Un editeur de texte/code**
  - Affiche le texte sans modification
  - Permet d'exécuter du code
  - Permet de previsualiser le rendu html
- **Notepad++ / visual studio / bracket / …**



Un site ou une app web

Définition & objectifs

- **Définition d'une architecture web**
  - L'architecture web est l'organisation des composants et des interactions qui constituent une application web.
  - Elle définit la structure, le comportement et les interactions entre les différents éléments d'une application web.
- **Objectifs d'une architecture web**
  - Fournir une base solide pour le développement et le déploiement d'applications web
  - Garantir la cohérence, la fiabilité et la performance des applications web
  - Faciliter l'évolutivité et l'extensibilité des applications web

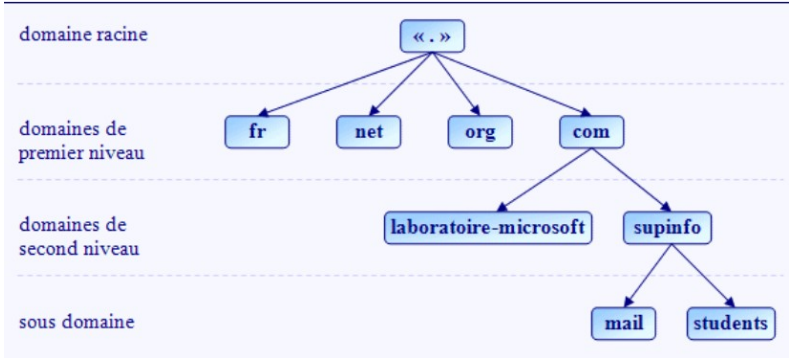


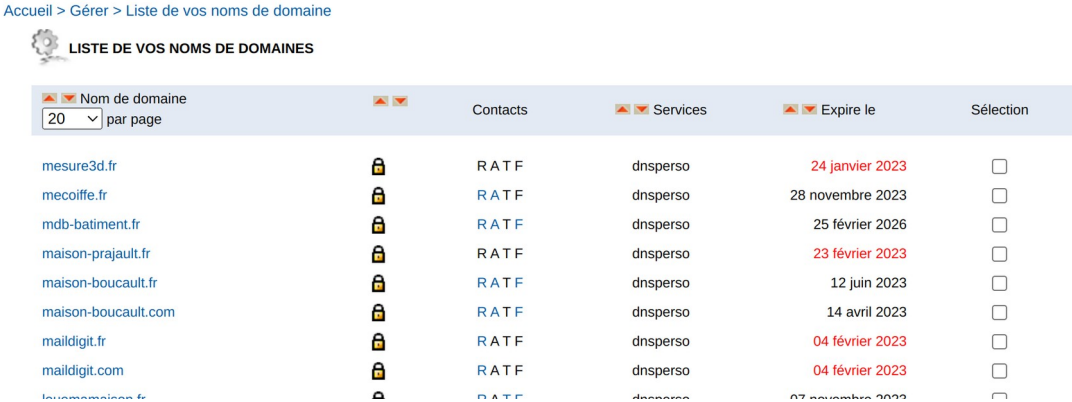
Modern web application architecture overview

Le réseau

Les noms de domaines et les Domain Name Server

- **Sur internet IPv4 ou v6 sont difficiles à retenir et surtout peuvent être dynamiques.**
  - Un annuaire hierarchique permet de traduire un nom de domaine en adresse IPv4 ou IPv6
  - Il existe 7 annuaires DNS physiques qui se synchronisent
- **Pour charger une page web 4 serveurs DNS sont impliqués**
  - Recurseur DNS (www.google.com va être découpé en sous requete vers les 3 autres serveurs)
  - Serveur de noms racine (.fr .com ...)
  - Serveur de noms TLD (top level domain)
  - Serveur de noms de reference (sous-domaine)
- **Des registars gérent les domaines racine et les TLD.**
  - On passe par ovh, ionos, 1&1, ... ont des accès à ces registars et revendent des TLD
  - Ils disposent d'interface de configuration type

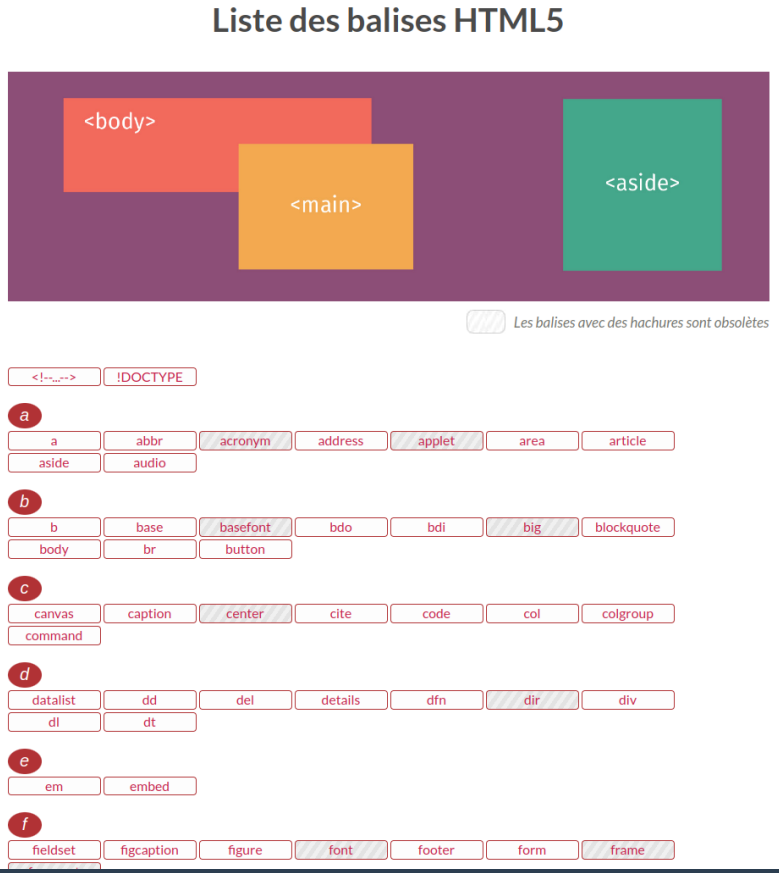




Les balises

HTML HyperText Markup Language

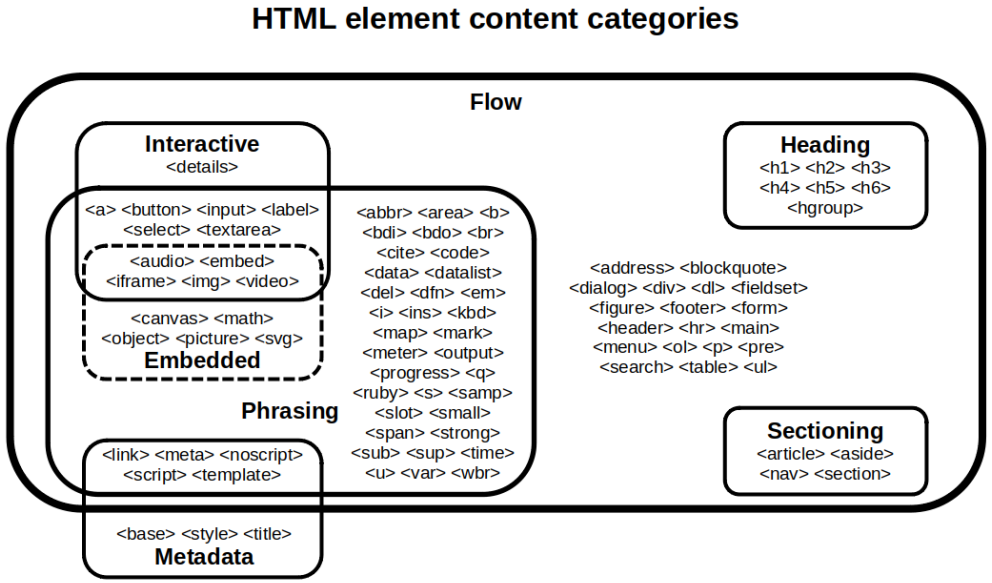
- **Les balises permettent d'isoler des blocs de textes**
- **Ces balises permettent aux navigateurs d'y associer des comportements particuliers**
- **https://jaetheme.com/balises-html5/**



Les balises

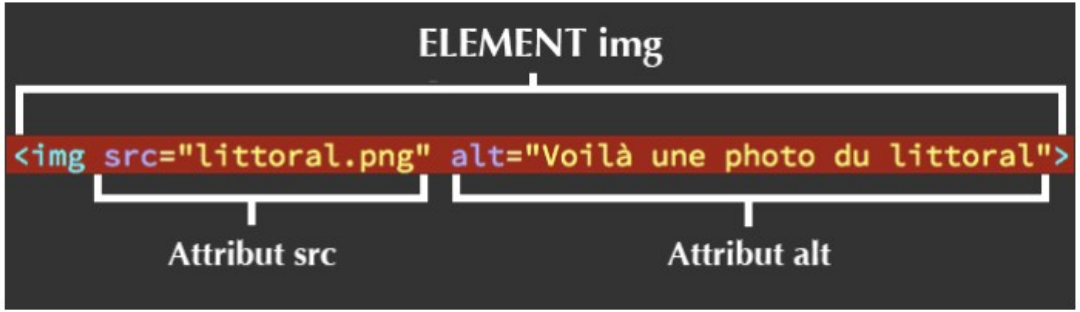
HTML HyperText Markup Language

- **Règle générale : une balise s'ouvre et se ferme pour encapsuler un contenu**
  - <h1>titre <a ></h1>
  - <h1> titre <a></h1>
- **La racine de base**
  - <html>
- **Les metadata**
  - <base> <head> <link> <meta> <style> <title>
- **La racine de section**
  - <body>
- **Contenu**
  - <article><aside><footer> <header> <hx> <main> <nav> <section>
- **Contenu textuel en bloc**
  - <hr>   <menu> <p> <pre>
- **Contenu texte en ligne**
  - <a> <b> <i> <s> <small> <time>
 <data> <code> ...
- **Contenu images et medias**
  - <img> <audio> <video> <area> <map> <track>
- **Contenu embarqué**
  - <embed> <objects> <portal> <iframe>
- **Contenu svg et mathML**
  - <svg><matlm>
- **Scripts**
  - <canvas><noscript><script>



Les attributs

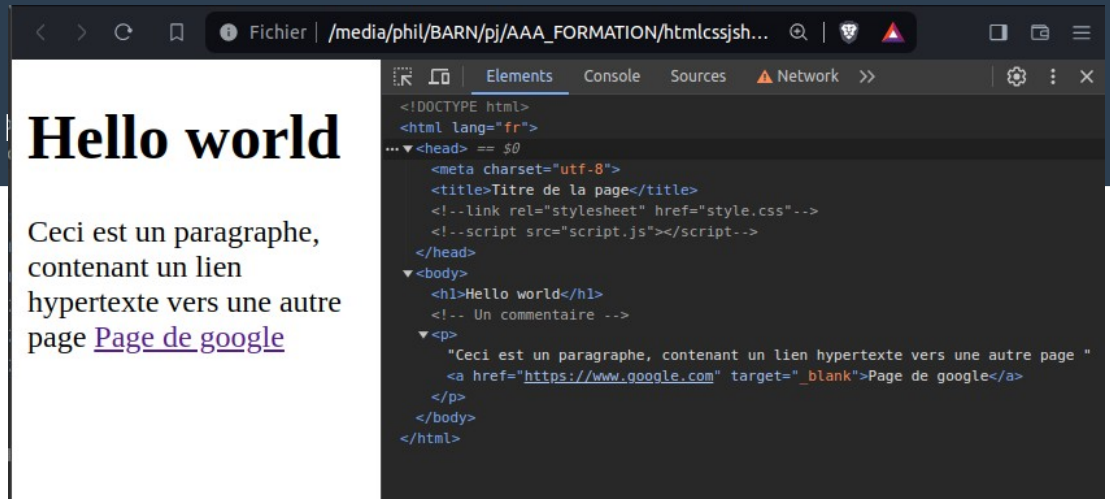
HTML HyperText Markup Language



- **Pour qualifier des balises, une balise a une liste d'attributs réservés :**
- **<tag attribute="valeur"></tag>**
- **Les plus utilisés :**
  - id : identifiant unique sur la page
  - class : pour typer un élément via une class css
  - src : indique la source de l'élément
  - data-xx : personnalisable utile dans des programme
  - alt, placeholder : utile pour remplacer un élément inaccessible ou en modèle
  - width, height, size, rows, color, mise en forme de balise

Les Elements

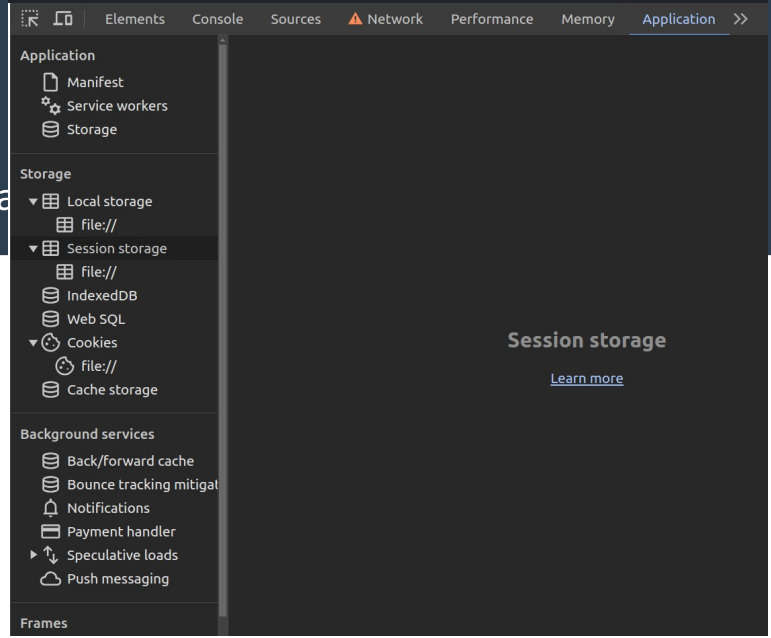
HTML



- **Une page = ensemble d'éléments :**
- **Créer un fichier local pagebase.html avec le contenu ci-dessus**
- **Enregistrez le fichier pagebase.html**
  - Affichez le dans votre explorateur de fichier
  - ouvrez le dans votre editeur de texte
  - puis ouvrez le dans votre navigateur
- **Modifier Hello world par hello {votre prenom}**
  - Class : pour typer un element via une class css
- **Passer en mode dev votre navigateur F12 ou click droit et inspecter**

Les navigateurs

HTML HyperText Markup Langua



- **Le mode developpeur est l'outil de debugage**
  - Éléments : visualise le code html
  - Console : permet de voir les erreurs et les traces console.log('hello') dans un programme javascript
  - Sources : visualise tous les fichiers en ligne
  - Network : liste et trace les requetes http
  - Application : stockage de données en local

Les bases

HTML HyperText Markup Language

- **Les titres <h1> <h6> <h1> titre</h1>**
- **Les paragraphes blabla **
- **Les blocs Textetexte en rouge **
- **Les retours à la ligne : texte sur ligne1
 texte sur ligne2 **
- **Les barres horizontale <hr>**
- **Mise en forme de base : <b> gras, <s> barré <i> italic**
- **Les espaces ne sont pas affichés sauf   (existe double ou quadruple espace)**

- **Exercice :**
  - Rédiger une présentation de qui vous êtes professionnellement (ou voulez être) avec un titre, et un ou 2 paragraphes avec mise en forme.

Les tableaux

- **Les tag clés : table, caption, tbody, tr (ow) , th (head) td (cellule)**

<table>

```
<caption>Titre</caption>
<tr><th>Titre col1</th><th>Titre col2</th></tr>
<tr><td>Cellule col1 ligne 1</td>
    <td> Cellule col2 ligne1</td>
</tr>
```

</table>

- **Exercice :**
  - Créer un tableau qui liste vos compétences avec 2 colonnes. La première avec le nom de la compétence et dans la deuxième votre niveau (débutant, maîtrise, expert)

Les formulaires

HTML HyperText Markup Language

- Les tag clés : form, input, textarea, select/options, button

```
<form action="/ma-page-de-traitement" method="post">
 <ul>
  <li>
    <label for="name">Nom :</label>
    <input type="text" id="name" name="user_name" />
  </li>
  <li>
    <label for="mail">E-mail :</label>
    <input type="email" id="mail" name="user_mail" />
  </li>
  <li>
    <label for="msg">Message :</label>
    <textarea id="msg" name="user_message"></textarea>
  </li>
 </ul>
 <div class="button">
   <button type="submit">Envoyer le message</button>
 </div>
</form>
```

- **Exercice :**
  - Créer un formulaire de contact, nom , email, message et un select avec 2 options (personnel ou professionnel)
  - Modifier le select pour le rendre ‘multiselectionnable’

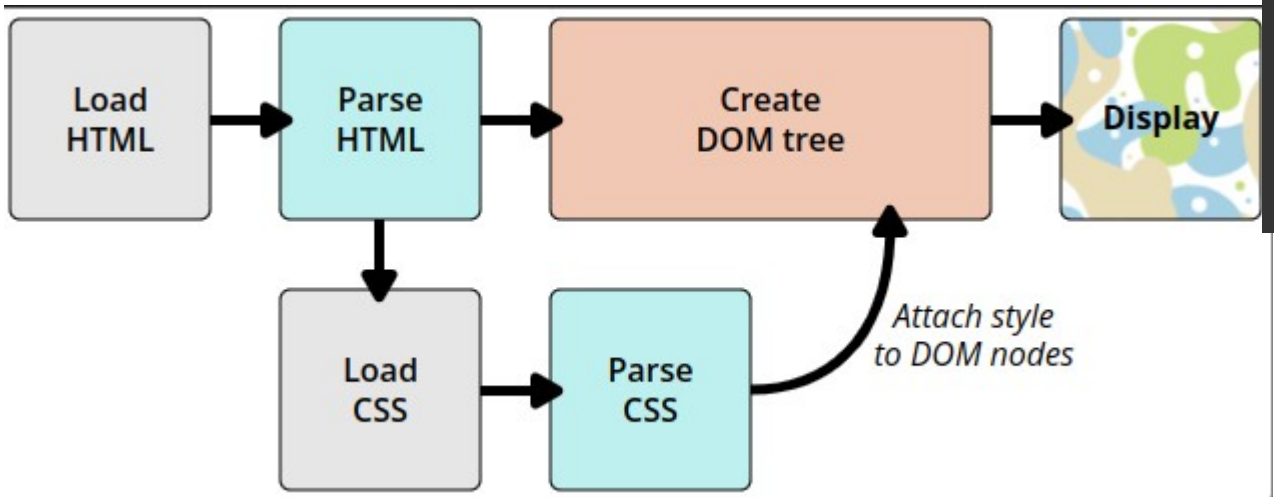
2. CSS 3 (2h)

Le CSS

Cascading Style Sheets

Mise en forme d'élément HTML (ou XML)

- 2 possibilités pour ajouter du style
- <style></style> dans le <head> ou un lien <link rel= 'stylesheet' href='style.css'>




```
<p>
  Let's use:
  <span>Cascading</span>
  <span>Style</span>
  <span>Sheets</span>
</p>
```

```
P
 ├─ "Let's use:"
 ├─ SPAN
 |   └─ "Cascading"
 ├─ SPAN
 |   └─ "Style"
 └─ SPAN
     └─ "Sheets"
```

```
span {
  border: 1px solid black;
  background-color: lime;
}
```

Let's use: Cascading Style Sheets

Le css

Style sheet (interne ou externe)

```
Selecteur {
directive : valeur;
}
```

ou Inline

```
<h1 style="color:blue;background-color:yellow;border:1px solid green ;">
Texte mis en forme</h1>
```

- **Exercice :**
  - Modifier vos titres et vos paragraphes (utiliser span pour modifier des morceaux de texte)

Le css / les selecteurs

HTML HyperText Markup Language

Selecteur {directive : valeur;}

#identifiant =>

.reponse =>

p =>

a[title] =>

input[name='email'] =><input name='email'>

p , h1 => <h1></h1> et

a:hover => <a> au passage du curseur

p::fisrt-line => blabla

lala selectionne la 1ere ligne

article > p => <article> </article> uniquement le 1er p à l'interieur de article.

section p => tous les p de la section

Sélectionne un tag et applique les directives à l'ensemble des éléments enfants.

Voir la table de reference des selecteurs

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Selectors

- **Exercice :**
  - Ajouter des identifiants et des class pour appliquer des styles identiques sur plusieurs sélecteurs et évaluer l'impact des sélecteurs

Le css / les directives

HTML HyperText Markup Language

Selecteur {directive : valeur;}

- **Mise en forme de texte**
  https://developer.mozilla.org/fr/docs/Learn/CSS/Styling_text
- **Directive display / position / float le positionnement**
  https://developer.mozilla.org/fr/docs/Learn/CSS/CSS_layout

Préparation d'une charte graphique :

- Code couleur https://paletton.com
- Font https://fonts.google.com

- **Exercice :**
  - Préparer une mise en forme de page web avec des lignes et des colonnes

Le css / les directives

HTML HyperText Markup Language

Selecteur {directive : valeur;}

- **Texte css: text-align, line-height, letter-spacing, color, margin, padding, text-shadow**
- **Animations (si changement alors délai)**
  - **#elt {transition : margin-left 5s;}**
- **Box-sizing (taille de bloc <> height width**
  - *** { box-sizing : border-box;}**
- **Positionnement**
  - **De base : utiliser weight, height, max-width avec margin et padding**
  - **Complexe : Utiliser principalement display :flex (bloc) ou display:inline (forcer la ligne)**
- **Utile avec js :**
  - **{ display : none ; visibility:hidden;}**
- **Responsive design :**
  - **html : <meta name="viewport" content="width=device-width, initial-scale=1">**
  - **css : @media all and (min-width: 760px) and (max-width: 959px) {}**
- **Fonts**
  - **<html><head> : <link href="http://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet" type="text/css" />**
  - **CSS : html { font-size:10px ; font-family : "Open Sans", sans-serif ; }**

- **Exercice :**
  - Installer plusieurs fonts sur un site
  - Choisir une police pour les <h> et une autre pour le <p> <a>


Javascript

Les bases du js : variables

Atelier : Les variables basejs.js

- **Une variable permet de stocker une valeur**
- **Declaration**

  let mavariable='texte' ; variable dont la portée est celle du bloc courant (lecture/ecriture)

  const mavariable='texte' ; constante dont la portée est celle du bloc (lecture uniquement)

  var mavariable='texte' ; variable globale

  Avant ES6 var

  Après ES6 let const (variable scopées moins de risque d'erreurs)
- Const objet = {} n'est pas modifiable mais ses propriétés le sont ex : objet.key=val

Variable	Explication	Exemple
Chaîne de caractères	Une suite de caractères connue sous le nom de chaîne. Pour indiquer que la valeur est une chaîne, il faut la placer entre guillemets.	`let myVariable = 'Bob';`
Nombre	Un nombre. Les nombres ne sont pas entre guillemets.	`let myVariable = 10;`
Booléen	Une valeur qui signifie vrai ou faux. `true` / `false` sont des mots-clés spéciaux en JS, ils n'ont pas besoin de guillemets.	`let myVariable = true;`
Tableau	Une structure qui permet de stocker plusieurs valeurs dans une seule variable.	`let myVariable = [1,'Bob','Étienne',10];` Référez-vous à chaque élément du tableau ainsi : `myVariable[0]`, `myVariable[1]`, etc.
Objet	À la base de toute chose. Tout est un objet en JavaScript et peut être stocké dans une variable. Gardez cela en mémoire tout au long de ce cours.	`let myVariable = document.querySelector('h1');` tous les exemples au dessus sont aussi des objets.

Les bases du js : Objets standards

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Global_Objects

Objets globaux standards (par catégorie)

Propriétés - valeurs

Les propriétés globales renvoient une valeur simple, elles ne possèdent aucune propriété méthode :

- `Infinity`
- `NaN`
- `undefined`
- le littéral `null`
- `globalThis`

Propriétés - fonctions

Les fonctions globales, appelées globalement (et non par rapport à un objet), renvoient directement leur résultat à l'objet appelant.

- `eval()`
- **`uneval()`** ⚠
- `isFinite()`
- `isNaN()`
- `parseFloat()`
- `parseInt()`
- `decodeURI()`
- `decodeURIComponent()`
- `encodeURI()`
- `encodeURIComponent()`
- `escape()` 🗑
- `unescape()` 🗑

Les bases du js : Objets standards

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Global_Objects

Objets fondamentaux

Ces objets sont les objets fondamentaux de JavaScript. Parmi ces objets, on retrouve les objets génériques, les fonctions et les erreurs.

- `Object`
- `Function`
- `Boolean`
- `Symbol`
- `Error`
- `EvalError`
- `InternalError` ⚠
- `RangeError`
- `ReferenceError`
- `StopIteration`
- `SyntaxError`
- `TypeError`
- `URIError`

Données structurées

Ces objets permettent de représenter et de manipuler des tampons de données (*buffers*) et des données utilisant la notation JSON (**J**ava**S**cript **O**bject **N**otation).

- `ArrayBuffer`
- `SharedArrayBuffer` 🧪
- `Atomics` 🧪
- `DataView`
- `JSON`

Objets de contrôle d'abstraction

- `Promise`
- `Generator`
- `GeneratorFunction`
- `AsyncFunction` 🧪

Nombres et dates

Ces objets permettent de manipuler les nombres, dates et calculs mathématiques.

- `Number`
- `BigInt`
- `Math`
- `Date`

Manipulation de textes

Ces objets permettent de manipuler des chaînes de caractères.

- `String`
- `RegExp`

Voir le site https://regex101.com

Et d'autres :
Collection indexées
Collection avec clés
Introspection
Internationalisation
webassembly

Les bases du js : Expressions & opérateurs

Atelier : baseis.js

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Operators

Expressions primaires
Expression « vers la gauche »
Incrementation et décrémentation
Operateurs unitaires
Opérateurs arithmétiques
Opérateurs relationnels
Opérateurs d’égalité
Opérateurs de décalage binaires
Opérateurs binaires booléens
Opérateurs logiques
Opérateur conditionnel ternaire
Opérateur d’affectation

Les bases du js : Instructions et déclarations

Atelier : Les variables basejs.js

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Statements

Contrôle de flux (bloc, break, if..then, try...catch)
Déclaration de variable (vu au début)
Fonctions et classes
Itération
Autres (export, import)

Les bases du js : Functions , Class, Error

Atelier : basejs.js

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Functions

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Classes

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Errors

Exercice js

- **Exercice :**
  - Créer 2 input pour saisir des entiers
  - Créer 3 boutons + - * au click appliquer l'opération et afficher le resultat dans un paragraphe.

- **Exercice :**
  - Créer une variable questionnaire =

    [{

    question : 'question1',

    reponses:[{id:1,reponse : 'Vrai'}, {id:2,reponse :'faux'}]

    }]
  - Afficher une par une la question avec les boutons
  - Au click alimenter une variable reponses=[] avec le code de la reponse clickée puis passer à la question suivante.