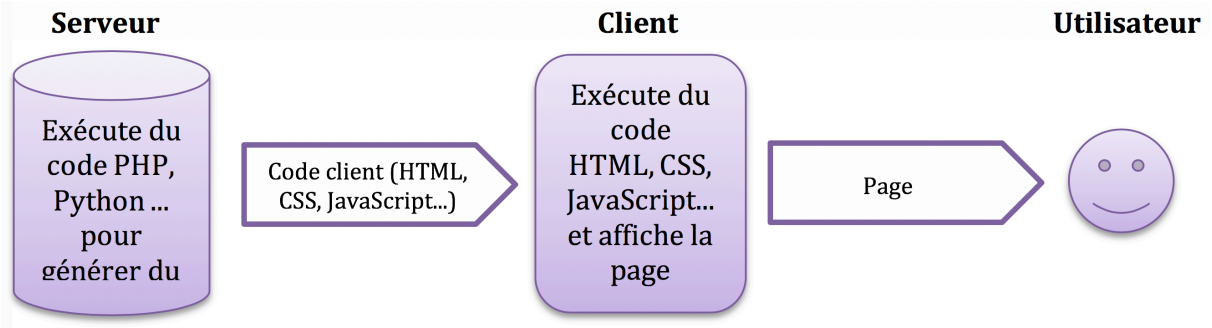
Soutien informatique

Langages Web côté client (1)

Les sites Web sont stockés sur des ordinateurs appelés **serveurs** et sont consultés par d'autres ordinateurs appelés **clients**. Lorsque vous vous connectez à un site, par exemple lorsque vous tapez www.google.fr dans votre navigateur Web, cela revient à demander une page Web stockée sur un ordinateur/serveur de Google. Une fois la demande transmise, le serveur renvoie le code de la page au client, c'est à dire à votre navigateur Web, qui interprète ce code et affiche la page correspondante. Pour visualiser le code d'une page, vous devez faire un clic droit dessus et sélectionnez « Afficher le code source » si vous êtes sur Firefox, ou la ligne équivalente si vous êtes sur un autre navigateur. Faites-le sur n'importe quelle page Web.

Il existe deux types de langages Web. Les langages du premier type (PHP, Python...) sont exécutés par le serveur, et permettent de créer des codes lisibles par le client. Les langages du second type (HTML, CSS, JavaScript...) sont lisibles et exécutables par le client. Lorsque vous faites un clic droit sur une page dans votre navigateur pour afficher son code source, vous ne verrez donc jamais de codes serveurs. En effet, ces codes auront été exécutés par le serveur avant que la page ne soit envoyée au client. En revanche, vous verrez des codes clients : ceux-ci sont envoyés par le serveur au client pour que ce dernier les exécute.



Dans ces séances de soutien, nous allons nous intéresser aux codes clients suivants :

- **HTML :** permet de définir le fond (contenu textuel, images, vidéos...) et la structure (titres, paragraphes...) de vos pages.
- **CSS :** permet de définir la forme de vos pages (quelle sera la police des paragraphes, la taille des images, la couleur des titres...).
- **JavaScript :** permet de rendre vos pages interactives (affichage de tel texte au survol de tel élément, changement de telle couleur de fond après avoir cliqué sur tel bouton…).

Exercice 1 : HTML

Une page Web au **format HTML** (.htm ou .html) contient des textes et des médias (images, vidéos...) ainsi que des **balises** permettant de les structurer. Par exemple, un titre de niveau 1 est déclaré à l'aide d'une balise `h1` :

```
<h1>Soutien informatique</h1>
```

Balise marquant le début du titre — Titre — Balise marquant la fin du titre

De même, un paragraphe est délimité par la balise `p` : `<p>Texte du paragraphe.</p>`.

Ouvrez le fichier cv.html dans une votre navigateur (de préférence Firefox ou Chrome). Faites un clic droit sur le fichier pour afficher la source et observez les balises. A quoi servent les balises `ul` et `li` ? A quoi servent les balises `h2` et `h3` ? A quoi sert la balise `strong` ?

Pour être correct, tout document HTML doit posséder une structure de base, dont les balises sont obligatoires :

```
<!DOCTYPE html>  Permet de spécifier la version HTML utilisé (ici HTML 5)
<html lang="fr">  Balise marquant le début du fichier HTML

  <head>  Balise marquant le début de l'en-tête de la page
    <meta charset="UTF-8" />  Balise spécifiant l'encodage utilisé lors de la
                                                    création du fichier (ici UTF-8)
    <title>  Titre affiché dans l'onglet  </title>
  </head>  Balise marquant la fin de l'en-tête de la page

  <body>  Balise marquant le début du contenu de la page
    Fond (contenu et balises structurantes)
  </body>  Balise marquant la fin du contenu de la page

</html>  Balise marquant la fin du fichier HTML
```

Lorsque vous créez un document HTML, commencez toujours par ajouter ces balises, avant d'ajouter votre contenu dans le `body`.

La norme décrivant les balises et la façon de les utiliser est définie par un organisme appelé le **W3C**. Les navigateurs utilisent ensuite cette norme pour décoder les fichiers HTML et les afficher correctement. Il arrive que certains navigateurs arrivent à afficher correctement une page mal codée. Cependant, si vous voulez avoir l'assurance que votre code fonctionnera correctement sur n'importe quel navigateur, il vous faut suivre la norme. Le W3C met à disposition un site permettant de vérifier la validité d'un code : https://validator.w3.org/#validate_by_input. Copiez-collez le code du fichier cv.html et vérifiez qu'il n'y a pas d'erreur.

Nous allons maintenant éditer le fichier source pour modifier la page. Ouvrez cv.html avec un éditeur de texte (NotePad++ ou ATOM par exemple). Remplacez une des balises `ul` par une balise `ol` sans modifier les `li` qui se trouvent à l'intérieur. Ouvrez cv.html dans votre navigateur Web (ou actualisez la page si vous ne l'aviez pas fermé). Quelle est la différence entre `ol` et `ul` ?

A vous de jouer : mettez à jour le CV en saisissant vos propres informations. Attention à bien structurer votre code, car si vous ne le faites pas, vous ne pourrez pas effectuer la mise en forme dans l'exercice suivant.

La plupart des balises s'utilisent ainsi :

```
<balise>Texte sur lequel la balise s'applique.</balise>.
```

Cependant, il existe des balises qui n'ont pas besoin d'être « fermées », i.e. il n'y a pas besoin de spécifier un texte sur lequel elles s'appliquent. Dans cv.html, c'est le cas des balises `<br />` et `<meta charset="UTF-8" />`. Comme vous pouvez l'observer, quand une balise est auto-fermante, un `/` est ajouté juste avant le `>`. Les balises peuvent aussi contenir des **attributs**. Par exemple, la balise `meta` de cv.html possède un attribut `charset` dont la valeur est `UTF-8`.

La balise `img` permet d'insérer une image dans une page HTML. C'est aussi une balise auto-fermante :

```
<img src="maphoto.jpg" alt="Ma photo" />
```

L'attribut `src` permet de spécifier le fichier de l'image, l'attribut `alt` permet de spécifier le texte qui sera affiché si l'image n'est pas disponible (par exemple si la connexion Internet n'est pas assez rapide ou si l'image ne se trouve pas sur le serveur). Dans cv.html, ajoutez une photo de vous juste avant le titre (balise `h1`). Le fichier de l'image doit que trouver dans le même répertoire que le fichier HTML.

Vérifier que votre code ne contient pas d'erreur en le copiant sur le service de validation du W3C (https://validator.w3.org/#validate_by_input). S'il en contient, corrigez-les avant de passer à l'exercice suivant.

Exercice 2 : CSS

Un style peut être directement inclut dans une page HTML, mais cette solution est vivement déconseillée. En effet, il est toujours très important de séparer le fond et la structure de la forme. Cela permet par exemple d'appliquer un même style à toutes les pages HTML d'un même site sans dupliquer le code sur chaque page. Cela permet aussi de faciliter la modification du code en cas de changement de charte graphique.

Pour séparer la forme du fond, on utilise des **feuilles de style CSS** qui se présentent sous la forme de fichiers portant l'extension .css. Ces fichiers sont chargés dans une page HTML à l'aide de la balise ci-dessous, qui doit être placée dans l'en-tête du document.

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

L'attribut `href` permet de spécifier le nom du fichier CSS.

Récupérez le fichier monfichier.css, placez-le dans le même répertoire que cv.html et chargez le grâce à la balise ci-dessus. Vous devriez observer un changement de la forme de l'image, des sous-titres et des paragraphes, ainsi qu'un changement de police sur tous les textes et une bordure grise sur le `body`. Si ce n'est pas le cas, c'est que vous avez fait une erreur, à vous de la trouver !

Ouvrez le fichier monfichier.css dans un éditeur de texte et observez le code.

Une règle CSS se définit ainsi :

```
selecteur{  Le sélecteur sélectionne des boites sur lesquelles les règles s'appliquent
  prop1: val1 ;  prop : nom d'une propriété.
  prop2: val2 ;  val : valeur que prend cette propriété pour les balises définies
  prop3: val3 ;                                                  par le sélecteur.
  ...
}
```

Dans le fichier monfichier.css, vous pouvez observer deux types de sélecteurs : `*` et `nomBalise`. Dans le premier cas, la règle s'applique à toutes les **boites** du document, c'est-à-dire à tous les contenus délimités par des balises. Dans le second cas, elle s'applique uniquement à toutes les boites délimitées par des balises de type `nomBalise`. Faites en sorte de comprendre l'effet que chaque propriété a dans le fichier monfichier.css (vous pouvez par exemple les modifier ou les supprimer momentanément pour observer les différences).

Dans votre navigateur Web, observez la bordure grise ajoutée à la balise `body`. Vous pouvez ainsi voir que la largeur du `body` s'adapte à la largeur de la fenêtre du navigateur. Lorsque cette largeur est importante, la lecture des longs paragraphes peut être difficile, on se perd lors des retours à la ligne. Il faut donc ajouter une largeur au `body` pour empêcher ce genre de problème (propriété `width`). Modifiez la feuille de style pour obtenir une largeur de `700px` et observez le résultat dans votre navigateur.

Le `body` apparait alors sur la gauche de la page. Une astuce pour le centrer consiste à définir des marges automatiques à droite et à gauche du `body`. Dans votre feuille de style, ajoutez les propriétés suivantes au `body` :

```
margin-right: auto;
margin-left: auto;
```

Observez le résultat : le `body` est désormais centré dans la page. Cette technique peut être utilisée pour centrer n'importe quelle boîte.

La propriété `float` appliquée à un élément permet de le faire flotter à gauche ou à droite. Dans votre feuille de style, ajoutez à votre image la propriété `float` de la façon suivante :

```
float: right;
```

Observez le résultat. L'image s'affiche désormais en haut à droite. Ajoutez aussi une marge en haut et à droite de l'image pour que cette dernière ne soit pas collée à la bordure du `body`.

Il existe de nombreuses autres propriétés pouvant être modifiées. Les principales sont répertoriées sur cette page : https://openclassrooms.com/fr/courses/1603881-apprenez-a-creer-votre-site-web-avec-html5-et-css3/1608902-memento-des-proprietes-css.

Jusqu'à présent, nous avons vu deux types de sélecteurs : `*` et `nomBalise`. Il peut être utile parfois de n'appliquer un style qu'à une boîte particulière, c'est à dire une occurrence particulière d'une certaine balise. Si l'on utilise le nom de la balise comme vu précédemment pour définir un tel style, alors celui-ci sera appliqué à toutes les autres balises portant le même nom. Pour ne l'appliquer qu'à une seule boite, il faut commencer par donner un identifiant à la boite en question dans le fichier HTML :

```
<nomBalise id="monIdentifiant">...</nomBalise>
```

Il faut ensuite utiliser le sélecteur suivant dans la feuille de style :

```
#monIdentifiant{
  ...
}
```

En utilisant cette technique, modifiez le paragraphe situé en dessous du titre principal (celui qui contient votre nom, votre prénom, votre adresse...) de façon à ce qu'il apparaisse en gras et avec une autre couleur (trouvez les bonnes propriétés dans la page Web mentionnée ci-dessus). Observez le résultat.

Il existe d'autres types de sélecteurs. Vous pourrez les trouver à l'adresse https://fr.wikibooks.org/wiki/Le_langage_CSS/Les_s%C3%A9lecteurs mais nous ne les testerons pas dans le cadre de cette séance.

A vous de jouer : continuez à mettre en forme votre CV en ajoutant/modifiant des règles dans votre feuille de style. Pour vous aider dans cette tâche, la plupart des navigateurs proposent des outils d'aide au développement. L'accès à ces outils se fait différemment selon les navigateurs :

- Firefox : Outils -> Développement Web -> Inspecteur
- Chrome : Afficher -> Options pour les développeurs -> Outils de développement -> Éléments
- Safari : Développement -> Afficher l'inspecteur Web

Ouvrez ces outils et survolez les différentes balises de votre page. Vous pouvez ainsi voir où les boîtes de chaque balise apparaissent dans la page et les règles CSS qui s'y appliquent.

Comme pour le HTML, la norme décrivant la façon de créer une feuille de style correcte est définie par le W3C. Vérifier que votre code CSS ne contient pas d'erreur en le copiant sur le service de validation du W3C (https://jigsaw.w3.org/css-validator/#validate_by_input). S'il en contient, corrigez-les avant de passer à l'exercice suivant.

Exercice 3 : JavaScript

Le **JavaScript** est un langage de programmation permettant d'ajouter du dynamisme à vos pages HTML. Il permet de modifier des éléments de fond, de structure ou de forme au déclanchement de certains **événements** comme le clic ou le survol de la souris sur une boite. Le code se place généralement dans l'en-tête de votre document à l'aide d'une balise de type `script` :

```
<script type="text/javascript">
   Votre code doit être placé ici.
</script>
```

La fonction JavaScript `alert` permet d'ouvrir un *popup* contenant la chaine de caractères passée en paramètre. Dans l'entête de cv.html, ajoutez un script dont le code est `alert("Bonjour")`. Rechargez votre page et vérifiez que le *popup* apparait et qu'il affiche la chaine de caractères `"Bonjour"`.

Comme pour les feuilles de style, un code JavaScript peut être placé dans un fichier spécifique et importé dans votre fichier HTML grâce à la balise suivante :

```
<script type="text/javascript" src="monscript.js"></script>
```

Supprimez le code que vous venez de créer de votre fichier HTML, créez un fichier monscript.js et ajoutez-y le même code. Importez-le ensuite dans votre fichier HTML et vérifiez que tout fonctionne encore correctement. Remarque : il ne faut pas écrire la balise script dans le fichier monscript.js. C'est la ligne d'import ci-dessus qui joue ce rôle.

Supprimez le code contenu dans le fichier monscript.js avant d'aller plus loin.

Pour ajouter un code JavaScript déclenché par certain événement sur une boite, il faut d'abord commencer par ajouter l'attribut correspondant sur la balise de la boite. Par exemple, le code suivant permet d'appeler une fonction JavaScript nommée `test()` lorsque l'utilisateur clique sur le titre :

```
<h1 onclick="test()">Data Scientist</h1>
```

Dans le fichier .js, il faut ensuite définir la fonction. Essayez avec la fonction suivante :

```
function test(){
  alert("Bonjour");
}
```

Si tout fonctionne correctement, lorsque vous cliquez sur « Data Scientist », un *popup* contenant la chaine de caractères « Bonjour » s'ouvre. Si c'est le cas, supprimer l'appel de la fonction `alert` de votre fonction `test`.

Le JavaScript permet de modifier le style d'un élément en le récupérant grâce à son identifiant. Commencez par attribuer un identifiant à votre titre comme nous l'avons vu dans l'exercice précédent :

```
<h1 id="idtitre" onclick="test()">Data Scientist</h1>
```

Maintenant, dans votre fonction `test`, utilisez la ligne suivante pour récupérer la boite `h1` sous forme de variable ici nommée `t` :

```
var t = document.getElementById("idtitre");
```

Donc maintenant, lorsque l'utilisateur clique sur le titre, on entre dans la fonction `test` qui récupère la boite du titre sous forme de variable. Il ne reste plus qu'à modifier le style contenu dans la variable `t`. Pour cela, utilisez la ligne suivante :

```
t.style.color="rgb(255, 0, 0)";
```

Dans ce cas, lorsque l'utilisateur clique sur le titre, sa couleur est changée en rgb(255, 0, 0), c'est du rouge. Testez.

Le JavaScript contient toutes les commandes des langages classiques : `if`, `else`, `while`, `for`… D'après vous, que fera le code suivant si on le place dans la fonction `test` à la place de `t.style.color="rgb(255, 0, 0)";` ?

```
if(t.style.color=="rgb(255, 0, 0)"){
  t.style.color="rgb(0, 0, 0)";
}
else {
  t.style.color="rgb(255, 0, 0)";
}
```

Essayez !

En HTML, il existe d'autres évènements que `onclick` permettant d'appeler une fonction JavaScript : https://developer.mozilla.org/fr/docs/Web/Events. Quels sont ceux qui correspondent au survol d'une boite par la souris ?

A vous de jouer : modifiez les codes des pages cv.html et monscript.js de façon à ce que les listes des sous-sections ne soient affichées qu'au survol par la souris de leur titre `h2` (puisque la sous-section « Projets et Réalisations » ne contient pas de liste mais un ensemble de `h3` et de `p`, on verra comment la gérer plus tard, concentrez-vous pour l'instant sur les autres sous-sections) :

1. Dans votre feuille de style, il vous faut d'abord cacher les listes `ul` grâce à leur propriété `display`. Si sa valeur est `none`, alors la liste est cachée, si elle est `block`, alors la liste est affichée.
2. Une fois vos listes cachées, modifiez le fichier cv.html de façon à ajouter des identifiants à vos listes et déclencher un événement qui appelle une fonction `afficher(idListe)` quand la souris survole un `h2` et `cacher(idListe)` quand la souris ne le survole plus.
3. Enfin, définissez ces fonctions dans le fichier monscript.js : elles doivent modifier la propriété `display` du style des listes `ul`.

Pour vous aider, il est possible d'afficher les erreurs de code grâce à la console des outils de développement proposés par les navigateurs :

- Firefox : Outils -> Développement Web -> Console web
- Chrome : Afficher -> Options pour les développeurs -> Console JavaScript
- Safari : Développement -> Afficher la console JavaScript

Donc quand vous êtes bloqué sur du code JavaScript, commencez par ouvrir la console et essayez de comprendre les erreurs !

Il ne reste plus qu'à faire la même chose pour la sous-section « Projets et Réalisations ». Le problème est qu'elle ne contient pas une seule liste comme les précédentes mais un ensemble de sous-sous-titres (`h3`) et de paragraphes (`p`). Il y a deux solutions possibles : (1) on fait des fonctions spécifiques qui les affichent et qui les cachent tous un par un, (2) on les regroupe dans une boite commune et on utilise le code déjà fait pour les listes sur cette boite. C'est cette solution que nous allons mettre en œuvre :

1. Dans un premier temps, regroupez les éléments de la sous-section « Projets et Réalisations » à l'intérieur d'une boite `<div>...</div>` (cette boite est très pratique pour ce genre d'opération, rappelez-vous en !).
2. Ensuite, donnez un identifiant à ce div.
3. Dans la feuille de style, cachez ce div.
4. Enfin, sur le titre, ajoutez les événements appelant les fonctions `afficher` et `cacher`.

Testez !