# Remise à niveau HTML/CSS

## Table des matières

# I. Utilisation d'un système de versionning : Git

Pour faciliter les échanges de fichiers du formateur, on va utiliser le logiciel git.

Cela permet de « versionner » les fichiers sur un serveur, afin que tous les développeurs travaillent sur les mêmes sources en même temps, avec chacun sa propre version sur son poste.

Le code de ce cours est disponible à l'url : https://github.com/ice-devel/formation-202006.git

Pour récupérer un projet :

- installer git
- git clone https://github.com/ice-devel/formation-202006.git

Pour récupérer les nouvelles modifications :

- git pull

# II. HTML

## 1. Présentation HTML/CSS

Le HTML/CSS sont deux langages de description. Ils servent à indiquer aux navigateurs quels éléments afficher dans une page ainsi que leur disposition.

C'est la base du web, les spécifications HTML/CSS sont basés sur un standard : tous les navigateurs les comprennent.

On code le html grâce à des éléments sous forme de balises ouvrantes/fermantes :

Contenu de la balise div

## 2. Utiliser la console du navigateur

Dans chaque navigateur, il y a un outil de debug : on peut y accéder en cliquant droit sur la page, et en sélectionnant « Inspecter l'élément », ou encore en appuyant sur la touche F12. Cela permet d'analyser le code HTML/CSS/JS en temps réel, de faire des modifications et visualiser ce qu'elles donnent, pour ensuite reporter les bonnes modifications dans le code source.

## 3. Balises/tags de base

- Html/head/body (title/charset)
- Paragraphes
- Images
- Blocs
- Titres
- Liens (url absolue / relative - ancres - attribut target)
- Listes à puces / listes numérotées

## 4. Les attributs des balises

- Attributs obligatoires

Certains attributs sont obligatoires. Par exemple, pour la balise image, il faut nécessairement renseigner la source via la balise « src », pour les liens, il faut indiquer l'URL via la balise « href », etc.

- Attributs facultatifs

Les « id », « class », « style » sont donc des attributs facultatifs, on les renseigne si besoin.

## 5. Position block/inline

Les éléments de type bloc prennent par défaut toute la largeur, et sont disposés les uns en dessous des autres.

Les éléments inline prennent la largeur de leur contenu et sont affichés les uns à côté des autres.

## 6. Exercice de mise en page

Créer un fichier HTML et intégrer le design fourni (exercice.jpg). Respectez bien les balises html, head, body et le doctype.

## 7. Tableau HTML

Les tableaux HTML permettent d'afficher des données sous forme de lignes et de cellules, un peu comme une feuille Excel.

## 8. Formulaire HTML

- Les différents types de champs
- Les attributs
    - name : obligatoire pour pouvoir récupérer la valeur côté serveur
    - value : obligatoire pour les boutons radios (et fortement conseillé pour les listes déroulantes)
    - min, max : number
    - minlength, maxlength : champs text
    - accept : champ file
- Les validations côté client
    - Required : champ obligatoire
    - Pattern : format de valeur attendu : expression régulière

Validation html5 : se déclenche au moment de la soumission du formulaire

Never trusts user input : Le formulaire peut être modifié par l'utilisateur, car il est côté client. Il faudra donc faire ces validations également côté serveur.

- Labels / Placeholders
- Envoi des données au serveur / attribut name / attribut action

Clé/valeur

- Method GET/POST

Avantages/inconvénients

    - GET : valeurs envoyées dans l'url (limiter en nb de caractères et visible par les personnes derrière le périphérique)
    - POST : valeurs envoyées mais en arrière-plan, pas de limite

On utilise GET pour partager des URLS, les mettre en favoris, etc.

## 9. Autres balises (html5, em/strong)

Header, Nav, Footer, Section, Article, Aside, Main : uniquement pour la sémantique

<audio> <video> : lecteur multimédia (dont le style est défini par le navigateur)

<iframe> : pour inclure une page web dans votre page web

# III. CSS

Le CSS permet de mettre en forme, de positionner des éléments HTML.

## 1. Les propriétés courantes

- Font-size : taille de la polie
- Color : couleur de la police
- Font-style : italic ou non
- Font-weight : gras ou non
- Border : bordure (taille, type, couleur)
- Margin : marge extérieure
- Padding : marge intérieure
- Background-color : couleur de fond

## 2. Style inline

On peut également appliquer du style directement dans le HTML, grâce à l'attribut style. Tous les html peuvent avoir cet attribut. On peut donc écrire le css directement dans cet attribut, c'est le style « inline ».

## 3. Sélecteur par tag / id (unique) / class

Chaque élément HTML peut avoir un ID (grâce à l'attribut id), une ou plusieurs classes (grâce à l'attribut class). Le tag est simplement le nom de la balise.

Grâce à ces 3 éléments, on peut sélectionner précisément un ou plusieurs éléments dans la page HTML pour leur appliquer du style.

- # (diese) : pour sélectionner par ID
- . (point) : pour sélectionner par classe
- rien : pour sélectionner par tag/balise. On écrit simplement le nom de la balise

## 4. Priorité entre sélecteurs / règle « !important »

Régle !important > Style Inline > Sélecteur ID > Sélecteur Classe > Sélecteur Tag

Les priorités ne font qu'écraser les propriétés déjà définies.

## 5. Combinaisons de sélecteurs / symboles spécifiques

## 6. Positionnement

- Static : dans le flux (les uns à côté des autres ou en dessous des autres)
- Absolute : hors du flux (positionnement avec coordonnées)
- Relative : dans le flux (mais positionnable avec coordonnées)
- Fixed : toujours au même endroit visuellement, même si scroll dans la page

## 7. Positionnement avancé

- Flexbox : permet de créer des « grilles » afin de positionner des blocs proprement et facilement. Les uns à côté des autres / en dessous des autres / dans l'ordre / dans l'ordre inverse, de répartir la largeur de façon équitable entre les différents éléments, etc.

## 8. Responsive design / Medias queries

Dans le but d'appliquer un style différent aux mêmes éléments en fonction de la largeur du périphique/navigateur, on peut utiliser les médias queries. Grâce à cela, on peut avoir un seul code HTML dont l'affichage est propre sur toutes tailles d'écran.

## 9. Frameworks HTML/CSS/JS

- Bootstrap
- MaterializeCSS
- Foundation

## 10. Minifier son code CSS

Pour optimiser son code CSS, on peut le minifier : cela permet un chargement plus rapide sur le réseau lorsqu'un navigateur télécharge le fichier.

Pour le minifier, il faut enlever les espaces inutiles, les retours à la ligne, etc. : on peut le faire à la main à chaque fois qu'on modifie ses fichiers inutiles, mais évidemment il vaut mieux passer par un outil qui le fait pour nous. On peut citer des « task runner » comme gulp, webpack, qui permettent d'automatiser ce traitement : il va générer automatiquement les fichiers minifiés qu'on lui aura dit de suivre.

Il suffit ensuite d'inclure dans notre code HTML les fichiers minifiés plutôt que les originaux (qu'il faut garder quand même pour les futures modifications, les fichiers minifiés étant illisibles).

Dans l'autre sens, si on n'a pas accès aux fichiers originaux, on peut copier-coller le code minifié dans son IDE favori et lui demander de reformater en fonction de notre style syntaxique préféré. Par exemple dans PHPStorm, le raccourci est : ctrl + alt + L.

Par convention on appelle minifié de la même façon que les originaux, mais en rajoutant « .min »juste avant l'extension du fichier.