**Les bases HTML5 ET CSS3**

ctrl / (pas sur le pavé numérique)pour un commentaire sur sublime text

<p style="color: red">Hello my love !</p> balise de pair, + css.

pour les attributs:

- sur une balise pair

<p attribut ="valeur">Hello le monde</p>

sur une balise orpheline ou simple.

<br attribut = "valeur">

ex : <p style="color:blue">Coucou le monde</p>

- **<p></p>** la balise p est pour un paragraphe.

- **<br/>** la balise br pour est pour un saut de ligne.

- **<span></span>** la balise span pour un petit texte ex les formulaires ou footer.

- De **<h1>** à **<h6>** balises de titres de différents niveaux de priorités (taille de la police).

- **<em> </em>** pour mettre en évidence les mots peu importants. l'explorateur le met en italique.

- **<strong> </strong>** pour mettre en évidence les mots importants

- **<mark> </mark>** marque le texte, le surligne pour attirer l'attention

- **<i> </i>** pour mettre en italique la police (italic).

- **<b> </b>** pour mettre en gras (bold).

- **<u> </u>** pour souligné.

- **Liste non ordonnées** : unordered list

- **<ul> </ul>** a utiliser avec **<li></li>** list item (avec puces)

- **Liste ordonnées** : ordred list

- **<ol> </ol>** a utiliser avec **<li></li>**(avec numérotation)

- **Listes de données** : data list

- **<dl> </dl>** a utiliser avec **<dt></dt>** data title, **<dd></dd>** data data ()

- **Les listes en css** :

pour enlever les puces:

list-style-type:none;

- **<a></a>** ancre pour donner la direction d'un autre fichier ou bien une direction dans la page ouverte.

Avec **<a>** il y a l'attribut (href=" ") ou l'on doit renseigner l'adresse complète du site, s'il est factice mettre #.

- **le a en CSS** :

Enlever le soulignement :

text-decoration: none;

- Dans un texte encadrer un mot du tag **<Strong>**, le mot devient en gras. (Mot ou autre partie du texte)

le tag **<Span>** peut être utilisé à la place de **<strong>**, mais il doit être complémenter avec un attribut(plus précis)

- le tag **<div>** est utiliser pour compartimenter la page web Il est aussi possible de lui donner une taille.

- le tag **<form>** est pour les formulaires il est comme le tag des listes, il doit être complémenter avec le tag **<label>** et le tag **<imput>** (balise seul) et leurs attributs, il peut y avoir aussi des tags comme textarea, select et option

- la balise **<em>** est pour mettre en italiques les mots de moyenne importance.

La balise **<strong>** le met en gras et la balise **<mark>** surligne le mot important, mais attention !! Ces trois balises concernent

le moteur de recherche et peuvent être modifier d'apparence en CSS

les alternatives sont **<i>** pour l'italique et **<b>** pour (bold) gras et **<u>** pour souligné-->

- insérer une image avec le tag **<img/>** (il est unique et autofermé) il est complémenté par src (source) qui donne

L’emplacement de l'image. L’attribut alt est pour remplacer celle-ci au cas où l'image n'apparait pas. L'attribut title

Lui décrit le logo

- le tag <figure> permet de mettre une légende a l'image, lui donné de l'importance en complémentarité du tag <figcaption>

-------------------------------------------------------

**css**

- Le selector ex:( p ) ensuite les crochet{} puis enfin les valeurs (font-size...) et leurs attributs ex:(30px)

Il y a les classes (.nomDeLaClasse), les id (#nomDeId) à utilisation unique et le sélecteur universel(\*) pour dire que tous les sélecteur son concerné.

Récapitulatif des sélecteurs :

**La sélection par la balise**

p {}

**La multi-sélection**

h1, em { }

**La sélection par la classe**

.class { }

**La sélection par l'ID**

#id { }

**Le sélecteur universel** (toutes les balises)

\* { }

**La balise B contenue dans A**

A B { }

**La balise a qui a un attribut title**

a[title] { }

**La balise a qui a un attribut title qui a un valeur "Accueil"**

a[title="Accueil"] { }

**La balise a qui a un attribut title dont le mot "Accueil" est en tout ou partie la valeur**

a[title\*="Accueil"] { }

**Tous les parargraphes suivant directement le h1.**

h1~p {}

**les titres de niveau 2 dans les sections**

section > h2 { }

**les titres et les textes important sont concerné**

h1, em {}

**Sélectionne la première balise<p>située après un titre <h3>**

h3+ p {}

Toute la liste : <https://www.w3.org/Style/css3-selectors-updates/WD-css3-selectors-20010126.fr.html#selectors>

- **les couleurs** :

Elles peuvent êtres exprimés en nom (red, blue... 16 couleurs officiel), en rgb (0, 12, 255) Red Green Blue, en rgba, a qui

Signifie alpha (le degré de transparence d'une couleur) qui va de 0 à 1, et enfin en hexadécimale (ne pas oublier la valeur #) certains

site propose ces valeurs (<http://www.code-couleur.com/)(www.colorpicker.com>).

Il est possible de jouer avec la transparence avec.

* opacity : est une propriété qui gère la transparence d'un élément, d'un texte, d'une image. Comme le rgba les valeurs sont entre 0 et 1.

- **La police** :

On peut exprimer la police en mot (small, medium, large..), il y a le pixel (px), et la valeur em ex:(1em = taille normal de la police selon l'élément)il est conseillé de mettre la taille de la police en em car elle s'ajuste selon la taille de l'écran.

En css il y a trois police intéger: le serif, le sans-serif et le cursive, il y a les polices propre à notre

PC et enfin les polices rajoutées au code via des sites ou l'entreprise qui nous employe. Utilisation des guillemets obligatoire

Lorsque la police est en 2 mots.

- Jouer avec la police :

**font-size** : règle la taille de la police. (px =>taille absolue, em => taille relative ou bien en %)par defaut la police est à 16px.

**font-family** : choix de la police. Si la police comporte des espaces, il faut mettre des guillemets.

**font-variant**: small-caps; petites majuscule.

<https://openclassrooms.com/fr/courses/1603881-apprenez-a-creer-votre-site-web-avec-html5-et-css3/1605329-formatez-du-texte>.

**font-style** : italic, oblique, normal. font-style ne prend que ces trois valeurs.

**font-weight** : normal , bold. C'est l'épaisseur de la police.

**text-decoration** : underline=> souligné; line-through => barré; overline=> ligne au dessus; blink=> clignotant; none=> normal(par defaut)

**text-transform**: uppercase, capitalize,lowercase, initial..

**text-align** : left, right, center, justify.

**text-shadow** :(en px) le décalage horizontal de l'ombre, le décalage vertical de l'ombre, l’adoucissement du dégradé (opacité),(optionnel, la grosseur), la couleur de l'ombre (peut-être donné en hexadécimal, rgb ou rgba). Idem pour box-shadow.

**letter-spacing** : espace entre les lettres.

- **Mettre une police personnalisé** :

- exemple avec **font-face**:

**@font-face**{

**font-family**: 'nom de la police';

src:url('L'adresse de la police par rapport au fichier CSS et son nom complet');

format('otf'); pour le format ttf, il faut écrire truetype entre ().

}puis mettre la police dans le sélecteur voulu.

- exemple avec google-font :

Aller sur <https://fonts.google.com/> sélectionné la police voulue avec le petit +, une petite fenêtre s'ouvre. Une fois cliqué dessus, il n'y a plus qu'à copier la première adresse dans le fichier html et la seconde dans le CSS.

- **Les flottants** :

Float : ne prend que 2 valeurs left et right.

Voir le lien OpenClassRooms dans le texte des polices pour son utilisation.

clear : pour stoppé un élément flottant. les valeurs sont : left, right et both.

- **insertion d'une image**:

Pour l'insertion d'une image, il est conseillé de mettre celle-ci dans le même fichier que celui du css.

Avec background (la propriété la plus utilisée)ou background-image.

il est possible de de mélanger les images de fonds en cumulant les url par des virgules.

background :avec background on peut cumuler toutes ces propriétés ci-dessous.

background-repeat : par defaut c'est repeat, on peux mettre no-repeat.

background-position : top, center, bottom, left, right.. ou bien en px.

background-attachement :

fixed: l'image de fond reste fixe .

scroll: l'image de fond défile avec le texte (par défaut).

exemple: selecteur{

background: url('images/steps-icon-1.png') no-repeat top center;

=> (dossier/nom de l'image.extention, cela peut être un adresse internet )no-repeat ou bien repeat(valeur par defaut):selon la taille de

l'image et du texte, l'image peut se dupliqué sur la longueur du texte (**repeat-x** ou **repeat-y**); top

et center pour placé l'image en haut et au milieu du paragraphe.

}

- **La propriété border** :

il y a trois valeurs à rentrées: sa taille (px), sa couleur, et son type (comme ses 9 types):

none: pas de bordure (par défaut) ;

solid: un trait simple ;

dotted: pointillés ;

dashed: tirets ;

double: bordure double ;

groove: en relief ;

ridge: autre effet relief ;

inset: effet 3D global enfoncé ;

outset: effet 3D global surélevé.

il est possible de choisir l'angle de coté, border-top, border-left ...etc.

Avec la propriété border-radius (px, %, em)il est possible d'arrondir les angles.ON peut choisir les angles à arrondir(haut gauche, haut droite,

bas droite, bas gauche). pour une courbe elliptique mettre un / entre 2 valeur. ex:10px/30px.

- **Box-shadow:**

rajoute une ombre à la bordure.

Voici les valeurs à renter:

le décalage horizontal de l'ombre x;

le décalage vertical de l'ombre y;

l'adoucissement du dégradé ou flou de l'ombre;

la longeur de l'ombre.(optionnel)

la couleur de l'ombre.

Si l'on veut le dégadé à l'interieur de la bordure on rajoute à la fin des valeur le mot inset, par defaut c'est la valeur outset.

- **Le passage de la souris** :

le selecteur :pseudo-format{ propriete } ne pas oublier les :.

les valeurs sont

*: hover*(quand la souris est sur le lien),

*: active* (permet d'appliquer un style particulier au moment du clic),

*: focus* (appplique un style lorsque l'élément est sélectionné)

*: visited* (quand le lien a déjà été vu).

---------------

**HTML**

**Structuré son code HTML** : avec les blocks et inlines.

- *block* : une balise de type block sur votre page web crée automatiquement un retour à la ligne avant et après. Il suffit d'imaginer tout simplement un bloc. Votre page web

sera en fait constituée d'une série de blocs les uns à la suite des autres. Mais vous verrez qu'en plus, il est possible de mettre un bloc à l'intérieur d'un autre, ce qui

va augmenter considérablement nos possibilités pour créer le design de notre site !

*block* :

**<p>** : pour les paragraphes.

**<h1>** à **<h6>**: titres

**<div>** : separateur

**<header>** : l'entête.

**<footer>** : pied de page.

**<nav>** : liens principaux de naviguation (menu principale).

**<section>** : regroupement des contenus en fonction de leurs thématique.

**<aside>** : contenu complémentaire au document visualisé.

**<article>** : La balise **<article>** sert à englober une portion généralement autonome de la page.

C'est une partie de la page qui pourrait ainsi être reprise sur un autre site.

C'est le cas par exemple des actualités (articles de journaux ou de blogs).

- *inline* : une balise de type inline se trouve obligatoirement à l'intérieur d'une balise block. Une balise inline ne crée pas de retour à la ligne, le texte qui se trouve à l'intérieur s'écrit donc à la suite du texte précédent, sur la même ligne (c'est pour cela que l'on parle de balise « en ligne »).

*inline* :

**<em>**, **<strong>**, **<mark>**, **<a>**, **<span>**...

----------------

**CSS**

- **les dimensions**:

**width** : la largeur.(px ou %)

**height** : la hauteur.(px ou %)

Par défaut, un bloc prend 100% de la largeur disponible, donc les dimensions width et height seront les valeurs à l'interieur de ce bloc.

**min-width**: largeur minimale . Défini une valeur minimale à ne pas dépassé.

**min-height**: hauteur minimale . Défini une valeur maximale à ne pas dépassé.

**max-width**: largeur maximale .

**max-height**: hauteur maximale.

Ex: Si l'on défini une largeur de 50%, pourra exiger que la largeur mini sera de 400px.

- **les marges**:

- **margin**:

(top, right, bottom, left)Il est possible de mettre que 2 valeurs si son opposé est la même ex: 2px 6px(idem pour padding). Ce sont les marges exterieures d'un block. (le + souvent en px)

- **padding**:

(top, right, bottom, left) Ce sont les marges interieures d'un block. (le + souvent en px).

***Les conteneurs (flexbox)***:

les propriétés ci-dessous sont à mettre pour la plus part dans le conteneur (parent), pour diriger les éléments direct (enfant).

- **display** : type d'éléments(flex,block, inline, inline-block, none...)les valeurs:

block: ce sont les balises qui font un retour à la ligne et qui sont redimansionnable tel **<div>**, **<p>**. ...

inline : ce sont les balises qui ne font pas un retour à la ligne, tel que **<a>**, **<i>**, **<span>**... .Côte à côte sur une même ligne.

inline-block : côte à côte sur une même ligne mais avec la possiblité de lui donné des dimensions pour des balises comme **<span>,** **<a>,** **<i>**...

flex : positionne les élément d'un conteneur sur une ligne(par défaut ils sont les uns au dessus des autres).

- **flex-direction** positionne les box:

row : organisés sur une ligne (par défaut, display:flex)

column : organisés sur une colonne

row-reverse : organisés sur une ligne, mais en ordre inversé

column-reverse : organisés sur une colonne, mais en ordre inversé

- **flex-wrap**: pour que les élément du conteneur aille à la ligne automatiquement.

nowrap : pas de retour à la ligne (par défaut)

wrap : les éléments vont à la ligne lorsqu'il n'y a plus la place

wrap-reverse : les éléments vont à la ligne lorsqu'il n'y a plus la place en sens inverse.

- **justify-content** : place les éléments

flex-start : alignés au début (par défaut)

flex-end : alignés à la fin

center : alignés au centre

space-between : les éléments sont étirés sur tout l'axe (il y a de l'espace entre eux)

space-around : idem, les éléments sont étirés sur tout l'axe, mais ils laissent aussi de l'espace sur les extrémités.

space-evenly: laisse un espace d'égale distance entre chaques éléments.

Si nos éléments sont placés dans une direction horizontale (ligne), l'axe secondaire est... vertical. Et inversement, si nos éléments sont dans une direction verticale (colonne), l'axe secondaire est horizontal.

- **align-items** , nous pouvons changer leur alignement sur l'axe secondaire (axe non déclaré). Il peut prendre ces valeurs :

stretch : les éléments sont étirés sur tout l'axe (valeur par défaut)

flex-start : alignés au début

flex-end : alignés à la fin, en bas du bloc.

center : alignés au centre

baseline : alignés sur la ligne de base (semblable à flex-start).

Il est possible d'aligner un seul élément:

- **align-self**: à mètre dans un des éléments du conteneur. Ces valeurs sont les même que la propriété align-item.

- **align-content** : repartir les éléments.

flex-start : les éléments sont placés au début.

flex-end : les éléments sont placés à la fin.

center : les éléments sont placés au centre.

space-between : les éléments sont séparés avec de l'espace entre eux.

space-around : idem, mais il y a aussi de l'espace au début et à la fin.

stretch (par défaut) : les éléments s'étirent pour occuper tout l'espace.

L'alignement des éléments se fait sur l'axe principal avec **justify-content** , et sur l'axe secondaire avec **align-items**.

- On peut réorganiser les éléments avec le pseudo-format : nth-child(), en ajoutant la propriété order. Par-contre il faut obligatoirement numéroté tous les éléments du conteneur(ici il y en a 2).

expemple :

.element:nth-child(1){

order:2;

}

.element:nth-child(2){

order:1;

}

- **flex** : avec cette propriété on peut étirer un élément (sur une ligne ou une colone) pour compléter l'espace vide.

ex:

element:nth-child(1){

flex:2;

}

Ici l'élément 1 prendra 2 fois plus de place à écran que les autres éléments du même conteneur.

- **Centrer des blocs**:

Il est tout à fait possible de centrer des blocs. C'est même très pratique pour réaliser un design centré quand on ne connaît pas la résolution du visiteur.

Pour centrer, il faut respecter les règles suivantes :

donnez une largeur au bloc (avec la propriété **width**) ;

indiquez que vous voulez des marges extérieures automatiques, comme ceci : **margin**: auto;.

- **La propriete overflow**:

visible(par défaut) : si le texte dépasse les limites de taille, il reste visible et sort volontairement du bloc.

hidden: si le texte dépasse les limites, il sera tout simplement coupé. On ne pourra pas voir tout le texte.

scroll: là encore, le texte sera coupé s'il dépasse les limites. Sauf que cette fois, le navigateur mettra en place

des barres de défilement pour qu'on puisse lire l'ensemble du texte. C'est un peu comme un cadre à l'intérieur de la page.

auto: c'est le mode « pilote automatique ». En gros, c'est le navigateur qui décide de mettre ou non des barres de

défilement (il n'en mettra que si c'est nécessaire). C'est la valeur que je conseille d'utiliser le plus souvent.

- **word-wrap** : pour coupé un mot ou une adresse très longue qui se trouve dans un cadre. il est conseiller de mettre cette propriété

directement dans un block qui pourrait contenir du texte saisi par un utilisateur (forums).

- **content** : pour definir un contenu dans une balise.

elle fontionne avec un pseudo selecteur (soit: before, after).

Si le contenant est un symbole, il faut le mettre entre double cotte.

- Autres procédés que les flexbox (à évité, il faut favorisé flexbox).

- **vertical-align**: (à mettre avec display:inline-block;

baseline (par defaut): aligne la base de l'élément avec celle de l'élément parent.

bottom : en bas.

middle : au milieu.

top : en haut.

- **position**:

C'est une technique particulièrement utile et puissante dans la réalisation d'un design.

absolute : positionnement d'un élément à l'emplacement que l'on désir(le positionnement se fait du coin en haut à gauche.) (même par dessus certain éléments)à utiliser avec les élélments top, bottom, left, right (en px ou %). Il est prioritaire sur les autres valeur de position.

fixed : même chose qu'absolute mais cette fois l'élément est toujours visible dans la page à l'endoit ou l'a placé.

relative: permet de décaler un élément par rapport à sa position initiale, (pour de petits ajustements).

static (par defaut):

Les éléments positionnés en absolu sont placés par-dessus le reste des éléments de la page ! Par ailleurs, si vous placez deux éléments en absolu vers le même endroit, ils risquent de se chevaucher. Dans ce cas, utilisez la propriété z-index pour indiquer quel élément doit apparaître au-dessus des autres.

- ***HTML*** :

- **Les tableaux**

Pour mettre un tableau, il faut commencer par écrire les balises **<table>** pour déclarer un tableau, ensuite la balise **<caption>** pour mettre un titre sur le tableau ensuite **<tr>** qui indique le début et la fin d'une ligne de tableau puis **<th>** pour les valeurs titre (les entêtes) en ensuite les balises **<td>** pour les valeurs.

Il est possible de fusionné des colonnes et des lignes avec dans la balise **<td>** un attribut **colspan** pour une colonne et **rowspan** pour une ligne.ex: **<td colspan="2">**Pour toutes la famille**</td>** le 2 signifie qu'il y aura 2 colonnes a fusionné.

*Pour diviser un gros tableau* :

**<thead>** pour l'entête , **<tbody>** pour le corps du tableau et **<tfoot>** pour le pied du tableau, celui-ci est conseillé d'être placer après <thead>.

- ***CSS*** :

- **Les tableaux**

Les tableaux en HTML ne sont pas très beaux, il faut donc mettre des bordures, une fois celle-ci mises, il faut les coller ensemble, dans le sélecteur table, il faut mettre la propriété **border-collapse** qui prends en paramètre **collapse** (collée) et une autre **separate** (séparé, qui est la valeur par défaut).

- **HTML** :

- **Les formulaires** :

Tous commencent par la balise **<form>**, puis une balise **<p>**. Dans la balise **<form>** on peut rajouter deux attributs :

- method = "get"; c'est une methode assez pau adaptée car elle ne peu contenir que 255 caractères. Les données envoyées passe par une adresse http://.. donc elle ne sont pas sécurisés.

- method = "post"; c'est la méthode la plus utilisée pour les formulaires car elle permet d'envoyer un grand nombre d'informations. Les données saisies dans le formulaire ne transitent pas par la barre d'adresse.

- action : c'est l'adresse de la page ou un programme qui va traiter les informations. qui traitera le message soit par email soit par une base de données. (avec PHP, ou bien JSexpress).

ex:

**<form** *action*="traitement.php" *methode*="post">

**<p>**Ici c'est le début du formulaire**</p>**

**</form>**

Pour insérer une zone de texte monoligne, on utilise la balise **<input/>** avec ses deux attributs : type="text" name="pseudo" l'attrubut name servira pour identifier le texte dans le PHP.

Pour identifier la zone de texte, on met la balise **<label>**. Pour lier cette balise aux champs, il faut mettre l'attribut for="" qui doit avoir la même valeur que l'id du champ.

ex: **<label** for="pseudo"**>**Votre pseudo**</label>** : **<input** type="text" name="pseudo" id="pseudo" **/>**

Dans la balise **<input>** on peut mettre en plus plusieurs attributs :

size : pour agrandir la taille du champ.

maxlength : le nombre de caractères max dans le champ.

value : pour préremplir le champ avec une valeur par défaut.

placeholder : donne une indication de la valeur a renter. Cet indicateur disparaitra dès que le visiteur aura commencé à remplir le champs.

Il y a la possibilité aussi de rentrer des mots de passes, à la place de la valeur *text* de l'attribut type, il faut mettre le mot *password*.

Pour renter un email, avec l'attribut type, il faut mettre : *email*.

Pour renter une url, avec l'attribut type, il faut mettre : *url*.

Pour renter un numero de téléphone, avec l'attribut type, il faut mettre : *tel*.

Pour renter un nombre, avec l'attribut type, il faut mettre : *number*. Ici on peut rajouter les attributs min, max, step.

Pour renter un curseur, avec l'attribut type, il faut mettre : *range*. Ici on peut rajouter les attributs min, max, step.

Pour renter une couleur, avec l'attribut type, il faut mettre : *color*.

Pour renter une date, avec l'attribut type, il faut mettre : *date*.

Différents types de champs de sélection de date existent :

date : pour la date (05/08/1985 par exemple) ;

time : pour l'heure (13:37 par exemple) ;

week : pour la semaine ;

month : pour le mois ;

datetime : pour la date et l'heure (avec gestion du décalage horaire) ;

datetime-local :pour la date et l'heure (sans gestion du décalage horaire).

Pour renter une recherche, avec l'attribut type, il faut mettre : *search*.

**Selectionner un champ automatiquement** :

Pour qu'un élément soit sélectionné automatiquement avec la balise **<imput>** mettre l'attribut autofocus.

**Rendre un choix obligatoire** :

Pour qu'un élément soit sélectionné automatiquement avec la balise **<imput>** mettre l'attribut required.

**Les éléments d'option:**

Pour renter une case à cocher, avec l'attribut type, il faut mettre : *checkbox*. Avec l'attribut checked on peut mettre une case cochée par défaut.

Pour renter une zone d'options, qui permet de faire qu'un seul choix parmis plusieurs. Avec l'attribut type, il faut mettre : *radio*. Le name doit être le même, mais doit avoir une value différente.

**Les listes déroulantes** :

Tout de suite après la balise **<input>,** mettre la balise **<select>** avec comme attributs name et id. Entre ces balises il faut mettre la balise **<option>** avec l'attribut value.

ex:

**<select** name="pays" id="pays"**>**

**<option** value="france">France**</option>**

**</select>**

Si l'on veut qu'une option soit sélectionnée par défaut, utilisez cette fois l'attribut selected au lieu de checked.

Avec **<optgroup>** qui se positionne entre la balise **<select>** et **<option>** on peut mettre un titre avec l'attribut label.

Pour insérer une zone de texte multiligne, on utilise la balise **<textarea>**, et la aussi les attributs name et id.

En css il sera possible de modifier la taille de la zone de texte avec les propriétés width et height. Sinon avec le attributs rows (nombres de lignes) et cols (nombres de colones).

**Regrouper les champs** :

Si votre formulaire grossit et comporte beaucoup de champs, il peut être utile de les regrouper au sein de plusieurs balises **<fieldset>**. Chaque **<fieldset>** peut contenir une légende avec la balise **<legend>**.

**Le bouton d'envoi** :

Dans une balise **<input>** avec comme attributs type et value.

type="submit": le principal bouton d'envoi de formulaire. C'est celui que vous utiliserez le plus souvent. Le visiteur sera conduit à la page indiquée dans l'attribut action du formulaire.

type ="*reset*": remise à zéro du formulaire.

type ="*image*": équivalent du bouton submit, présenté cette fois sous forme d'image. Rajoutez l'attribut src pour indiquer l'URL de l'image.

type ="*button*": bouton générique, qui n'aura (par défaut) aucun effet. En général, ce bouton est géré en JavaScript pour exécuter des actions sur la page.

Avec l'attribut value on peut changer le mot sur le bouton.

- **Les fichiers audio et vidéo** :

Pensez à consulter [www.Caniuse.com](http://www.caniuse.com/) pour connaître la compatibilité de l'audio et de la vidéo. S’il y a un problème, il est possible de les convertir avec : <http://www.mirovideoconverter.com/>

les codecs audio les plus utilisés sont : mp3, aac, ogg..

Pour l'audio :

**<audio>** avec l'attribut src="adresse et nom du fichier" .

Quelques attribut sans valeur.

- controls : pour rajouté des boutons de contrôle.

- width : modifier la largeur de l'outil.

- loop : pour lire en boucle.

- autoplay : lit automatiquement le fichier.( à éviter).

attribut avec valeur :

- preload : indique si le fichier peut-être préchargé. Voici quelques valeurs :

- *auto* = (par defaut) c'est le navigateur qui décide ce qu'il veut charger.

- *metadata* : charge uniquement les metadonnées (durée, etc.).

- *none* : pas de préchargement.

L'emplacement entre les 2 balises sert pour un message de non fonctionnalité.

ex :

**<audio** src="hype\_home.mp3" controls**>**Veuillez mettre à jour votre navigateur !**</audio>**

Si le fichier MP3 n'est pas pris en charge par le navigateur, il faut convertir le fichier et ajouter les nouveaux fichier dans une balise **<source>** et toujours l'attribut src.

ex :

**<audio** controls>

**<source** src="hype\_home.mp3"**>**

**<source** src="hype\_home.ogg"**>**

**</audio>**

Pour la vidéo :

Les codecs les plus utilisés sont : avi, mp4 , mkv..

**<video>** avec l'attribut src. Tous est quasiment identique à l'audio. Mais ici on peut rajouter un attribut :

- poster : On peut placé une image qui défini la vidéo avant sont lancement.

Voici un exemple :

**<video** src="sintel.webm" controls poster="sintel.jpg" width="600"**></video>**

Là aussi la balise **<sources>** est possible si l'on as plusieurs extensions pour le fichier à lire.

- **CSS** :

- **Les medias Queries** :

Il y a deux façon de créer des medias queries, soit en créant un fichier spécifique en CSS, soit en le dans le fichier CSS courant.

Pour la 1ère option il faut mettre un attribut media dans le <**link**> en plus du 1er **<link>**: **<link** rel="stylesheet" media="screen and (max-width: 1280px)" href="petite\_resolution.css" />

Ici quelques unes des règles les plus importantes des medias queries:

color : gestion de la couleur (en bits/pixel).

height : hauteur de la zone d'affichage (fenêtre).

width : largeur de la zone d'affichage (fenêtre).

device-height : hauteur du périphérique.

device-width : largeur du périphérique.

orientation : orientation du périphérique (portrait ou paysage).

media : type d'écran de sortie. Quelques-unes des valeurs possibles :

*screen*: écran « classique » ;

*handheld*: périphérique mobile ;

*print*: impression ;

*tv*: télévision ;

*projection*: projecteur ;

*all*: tous les types d'écran.

2nd option, dans le fichier CSS courant:

**@media** all and (min-width: 1024px) and (max-width: 1280px){

ici la déclarations des sélécteurs et des propriétés

}

Attention de bien respecter les espaces entre les mots!!

Il peut y avoir dans la déclaration des médias queries des mots clé pour plus de précisions :

- only : uniquement.

- and : et.

- not : non.

Le viewport c'est la largeur de la fenetre du navigateur sur le mobile. Chaque navigateur a son propre viewport..

Navigateur Largeur du viewport par défaut

Opera Mobile: 850 pixels

iPhone Safari 980 pixels

Android 800 pixels

Windows Phone 1024 pixels

ex:

Un iPhone se comporte comme si la fenêtre faisait 980 px de large, tandis qu'un Android se comporte comme si la fenêtre faisait 800 px !

On peut modifier la largeur viewport du navigateur mobile avec une balise **<meta>** dans le **<head>**.

ex : **<meta** name="viewport" content="width=320" **/>**

Vous pouvez utiliser cette balise pour modifier la façon dont le contenu de votre page s'organise sur les mobiles. Pour obtenir un rendu facile à lire, sans zoom, vous pouvez demander à ce que le viewport soit le même que la largeur de l'écran :

**<meta** name="viewport" content="width=device-width" **/>**

**CSS** :

- **Les dégradés linéaire** :

*linear-gradient* est une fonction (ou methode) qui se place dans par exemple un background.Cette fonction possède des parenthèses avec en paramètre les couleurs escomptés puis sa direction. Pour utilisé cette fonction il faut donc une hauteur fixe (px). Sinon il est possible de faire autrement pour pour que le dégradé occupe toute la page. Il suffit de mettre dans la balise **<html>** **:** style="min-heigth:100%" puis dans le selecteur une hauteur de 100% au lieu d'une valeur en px.

ex :

**le selecteur**{

background:*linear-gradient*(silver, grey, black);

height:1000px;

}

Pour indiquer une direction il faut que dans les parenthèses on commence par "to" puis la ou les directions.

ex :

**le selecteur**{

background:*linear-gradient*(to bottom right, silver, grey, black);

height:1000px;

}

On peu lui donné un angle precis "180deg" par exemple à la place de la direction : (180deg, silver, grey, black);

Il est possible de choisir le pourcentage de dégradé voulu entre 2 couleur : (to bottom right, silver 30%, grey, black);

- **Le sélécteur root** :

Le sélécteur **:root** (attention à bien mettre les :)est l'équivalent de la balise ou du selecteur html, mais les propriétés mse à l'interieur sont prioritaire. Il sert surtout pour la mise en place et l'application de variables dans le code CSS.

**:root**{

background-color:red;

}

- **Les variables** :

Une variable se déclare souvent et c'est recommandé dans le **:root**, mais peut très bien être déclaré dans l'élément qui doit l'utilisé.

Déclaration d'une variable:

--nomDeLaVariable: valeur;

Pour réutiliser une variable :

var(--nomDeLaVariable);

ex :

La déclaration

:root{

--maCouleurPrimaire:silver;

}

L'utilisation

h1{

background-color: var(--maCouleurPrimaire);

- **Introduction d'icones** :

site <https://fontawesome.com/>

copier la balise **<script>** et **<link>** dans le head :

Pour la police:

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

Pour les icones:

<script src="https://kit.fontawesome.com/8b07301a9e.js" crossorigin="anonymous"></script>

Ensuite copier le lien de l'icone recherché à l'emplacement voulut.