***Cours HTML-CSS***

Table des matières

[Section 1 : Introduction 3](#_Toc93396900)

[A/ Définition 3](#_Toc93396901)

[B/ Notes 4](#_Toc93396902)

[Section 2 : HTML Basics 5](#_Toc93396903)

[A/ Définition 5](#_Toc93396904)

[B/ Notes 6](#_Toc93396905)

[Section 3 : CSS Basics 7](#_Toc93396906)

[A/ Définition 7](#_Toc93396907)

[B/ Notes 8](#_Toc93396908)

[Section 4 : Hotel Website 9](#_Toc93396909)

[A/ Définition 9](#_Toc93396910)

[B/ Notes 10](#_Toc93396911)

[Section 5 : Intro to Responsive Layouts 11](#_Toc93396912)

[A/ Définition 11](#_Toc93396913)

[B/ Notes 12](#_Toc93396914)

[Section 6 : Intro to Flexbox 13](#_Toc93396915)

[A/ Définition 13](#_Toc93396916)

[B/ Notes 14](#_Toc93396917)

[Section 7 : EdegeLedger Website (Flexbox) 15](#_Toc93396918)

[A/ Définition 15](#_Toc93396919)

[B/ Notes 16](#_Toc93396920)

[Section 8 : Website Deployment – Shared Host 17](#_Toc93396921)

[A/ Définition 17](#_Toc93396922)

[B/ Notes 18](#_Toc93396923)

[Section 9 : More CSS Concepts – Advanced Selectors, Animation & More 19](#_Toc93396924)

[A/ Définition 19](#_Toc93396925)

[B/ Notes 20](#_Toc93396926)

[Section 10 : Mini Projects With KeyFrames, Transitions, etc … 21](#_Toc93396927)

[A/ Définition 21](#_Toc93396928)

[B/ Notes 22](#_Toc93396929)

[Section 11 : CSS Grid 23](#_Toc93396930)

[A/ Définition 23](#_Toc93396931)

[B/ Notes 24](#_Toc93396932)

[Section 12 : NewsGrid Website 25](#_Toc93396933)

[A/ Définition 25](#_Toc93396934)

[B/ Notes 26](#_Toc93396935)

[Section 13 : Website Deployment With Netlify 27](#_Toc93396936)

[A/ Définition 27](#_Toc93396937)

[B/ Notes 28](#_Toc93396938)

[Section 14 : Learning Sass 29](#_Toc93396939)

[A/ Définition 29](#_Toc93396940)

[B/ Notes 30](#_Toc93396941)

[Section 15 : Portfolio Website With Sass 31](#_Toc93396942)

[A/ Définition 31](#_Toc93396943)

[B/ Notes 32](#_Toc93396944)

# Section 1 : Introduction

## A/ Définition

*HTML* :

HyperText Markup Language (d’écrire de l’hypertexte)

CSS :

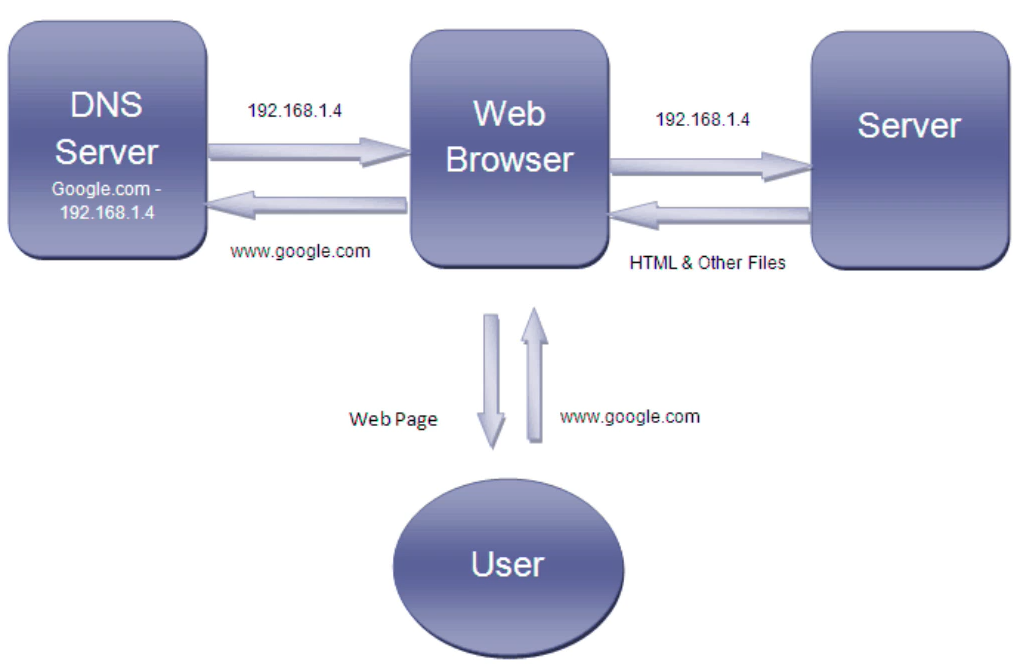
Cascading Style Sheets (Feuilles de style en cascade)

## B/ Notes

Les ordinateurs communiquent entre grâce au protocole CTP/IP

DNS est utilisé pour mapper l'adresse IP aux noms de domaine

Les noms de domaine peuvent être achetés auprès des bureaux d'enregistrement et liés à un compte d'hébergement/serveur Web



# Section 2 : HTML Basics

## A/ Définition

Charset :

Character encoding (Encodage de caractère)

## B/ Notes

La balise ( <meta name="viewport" content="width=device-width, initial-scale=1.0"> ) est là pour le responsive

La balise ( <meta name="description" content="This is my website description"> ) est là pour décrire le contenu du site

La balise ( <meta name="keywords" content="web development, web design"> ) est là pour aide le référencement du site grâce à des mots clés

<strong> & <b> => **Gras**

<em> & <i> => *Italique*

<br> => Retour à la ligne

<hr> => Ligne horizontal ( -------------------------------------------------------------------------------------------------- )

<del> => ~~Texte barré~~

<!-- Date -->

Une image contenant texte

Description générée automatiquementUne image contenant texte

Description générée automatiquementUne image contenant texte

Description générée automatiquement

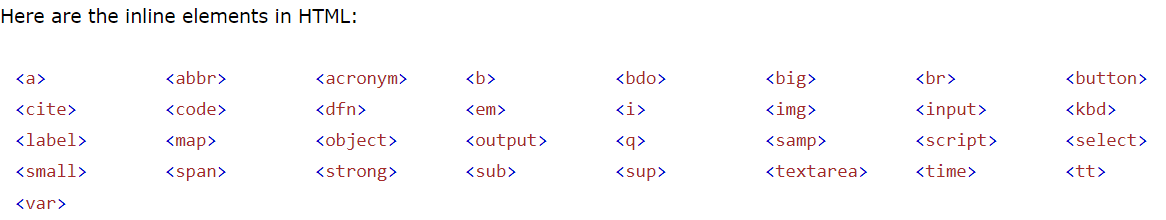
Une image contenant texte

Description générée automatiquement

*submit ==>*  envoie le formulaire

*reset* ==> réinitialise tous les champs du formulaire

Une image contenant texte

Description générée automatiquement

Utiliser *div* pour des blocks

Utiliser *span* pour des lignes (ex : sélectionner une partie du texte pour la mettre en couleur, gras, etc..)

Une image contenant texte

Description générée automatiquement

# Section 3 : CSS Basics

## A/ Définition

*Font sizes :*

* % = % de l’élément parents
* em = par rapport à la taille (de la police) de l’élément parent
* rem = par rapport à la taille de l’élément racine
* vw = 1% de la largeur de la fenêtre
* vh = 1% de la hauteur de la fenêtre

*background-position :*

background-position: 100px, 100px ; (axe des X, axe des Y)

Possède aussi les propriétés : *top* ; *bottom* ; *center*

background-size :

* *contain*  = redimensionne l'image afin qu'elle soit le plus grand possible et que l'image conserve ses proportions
* *cover* = l'image est redimensionnée pour être aussi grande que possible et pour conserver ses proportions. L'image couvre toute la largeur ou la hauteur du conteneur et les parties qui dépassent sont rognées si les proportions du conteneur sont différentes
* *auto* = redimensionne l'image d'arrière-plan afin que ses proportions soient conservées

*background-attachement :*

* *fixed* = l'arrière-plan est fixe par rapport à la zone d'affichage (viewport). Ainsi, même si l'élément dispose d'outils de défilement, l'arrière-plan ciblé ne se déplacera pas avec l'élément
* *local* = l'arrière-plan se déplace avec le contenu de l'élément associé. Ainsi, si l'élément défile, l'arrière-plan défilera avec. Les zones de positionnement et de dessin de l'arrière-plan sont relatives à la zone de l'élément plutôt qu'au cadre extérieur
* *scroll* = l'arrière-plan est fixé par rapport au contenu de l'élément (il ne défile pas avec) mais est rattaché à la bordure de l'élément

*box-sizing :*

* *content-box* = Si on définit un élément avec une largeur de 100 pixels, la boîte de contenu de cet élément mesurera 100 pixels de large et la largeur de la bordure et/ou du remplissage sera alors ajoutée pour constituer la largeur finalement affichée
* *border-box* = si on définit un élément avec une largeur de 100 pixels, ces 100 pixels incluront la bordure et le remplissage éventuellement ajoutés et c'est le contenu de la boîte qui sera compressé pour absorber cette largeur supplémentaire (on définit la taille totale (margin & padding compris))

*clear :*

L’élément est déplacé vers le bas

* *left* = dégager les flottements à gauche
* *right* = dégager les flottements à droite
* *both* = dégager les flottements à gauche & à droite
* *inline-start* = dégager le contenu vers le début du bloc englobant
* *inline-end =* dégager le contenu vers la fin du bloc englobant

*position :*

Positionnement de l’élément

* *static* = positionnement par défaut (pas d’effet particulier)
* *relative* = les propriétés [top](https://developer.mozilla.org/fr/docs/Web/CSS/top) ou [bottom](https://developer.mozilla.org/fr/docs/Web/CSS/bottom) indiquent le décalage vertical à appliquer et [left](https://developer.mozilla.org/fr/docs/Web/CSS/left) ou [right](https://developer.mozilla.org/fr/docs/Web/CSS/right) indiquent le décalage horizontal
* *absolute*  =  les propriétés [top](https://developer.mozilla.org/fr/docs/Web/CSS/top), [bottom](https://developer.mozilla.org/fr/docs/Web/CSS/bottom), [right](https://developer.mozilla.org/fr/docs/Web/CSS/right) et [left](https://developer.mozilla.org/fr/docs/Web/CSS/left) indiquent les distances entre les bords de l'élément et les bords du bloc englobant (c'est-à-dire l'ancêtre par rapport auquel l'élément est positionné)
* *fixed =* positionnement par rapport à la fenêtre (sort du flux)
* *sticky =* se comporte comme un élément positionné de façon relative jusqu'à ce que son bloc englobant dépasse un seuil donné (par exemple fourni par la valeur de [top](https://developer.mozilla.org/fr/docs/Web/CSS/top)) au sein du conteneur puis il se comporte ensuite comme un élément fixe jusqu'à atteindre le bord opposé du bloc englobant (il reste à la même position quand on scroll)

*inherit :*

Valeur qui peut être utilisée pour qu'une propriété prenne la valeur de la propriété de l'élément parent. Cette valeur peut être utilisée sur n'importe quelle propriété CSS, y compris sur la propriété raccourcie [all](https://developer.mozilla.org/fr/docs/Web/CSS/all).

L'héritage provient toujours de l'élément parent par rapport à l'arbre du document, même si cet élément n'est pas le bloc englobant

## B/ Notes

Pour les images transparentes utilisé des images au format *.pmg*

*border-style*  a trois propriétés :

* Solid = trait-plein ( ----------------------------)
* Dotted = trait composé de points ( . . . . . . . )
* Dashed = pointillés ( - - - - - - - - - - - )

z-index permet de positionner au premier plan, 2nd plan, etc… ( plus grand chiffre premier plan, 2eme plus grand chiffre 2nd plan, etc…)

En CSS l’ordre de priorité est :

* 1) id
* 2) classe
* 3) nom de l’élément (ex : div ; h1 ; etc…)

Les marges peuvent avoir des valeurs négative

Vu que position fixed dépend du body, la largeur des enfants d’un conteneur en position fixed dépendra du body et non pas du conteneur

# Section 4 : Hotel Website

## A/ Définition

*overflow :*

Définit comment gérer le dépassement du contenu d'un élément dans son bloc

* Visible

La valeur par défaut. Le contenu n'est pas rogné. Le contenu peut éventuellement être affiché en dehors de la boîte de remplissage (padding).

* hidden

Le contenu est rogné si besoin pour s'inscrire dans la boîte de remplissage (padding) et aucune barre de défilement n'est affichée.

* scroll

Le contenu est rogné pour s'inscrire dans la boîte de remplissage (padding) et les navigateurs de bureau affichent des barres de défilement dans tous les cas. Cela évite d'avoir des barres qui apparaissent et disparaissent sans cesse avec du contenu dynamique.

* auto

Le comportement est laissé à la discrétion de l'agent utilisateur. Les navigateurs comme Firefox affichent des ascenseurs si le contenu dépasse dans la boîte de remplissage (padding)

* overlay

Cette valeur se comporte comme auto sauf que les barres de défilement sont au-dessus du contenu plutôt que de prendre de la place. Cette valeur est uniquement prise en charge par les navigateurs basés sur WebKit (Safari) ou Blink (Chrome, Opera).

*FontAwesome :*

Site de fonts et d’icones

## B/ Notes

Penser à reset le css

Bien organiser son CSS (Section ( /\* Navbar \*/ /\* Main \*/ )

Utiliser des classes utilitaires pour pouvoir définir des choses récurrentes (ex : background blanc & noir, texte important, bouton, etc…)

Penser à clear après des float successifs

# Section 5 : Intro to Responsive Layouts

## A/ Définition

*Font-size :*

* *em* = Multiplication de la taille par rapport au parent

(ex : parent = 20px ; enfant = 1.5em ==> enfant = 20 \* 1.5 = 30px)

* *rem* = Multiplication de la taille par rapport à l’élément racine

(ex : racine = 16px ; enfant = 1.5em ==> enfant = 16 \* 1.5 = 24px)

*Element-size :*

* *vw* = pourcentage de la largeur du viewport (fenêtre) (100 vw = toute la largeur)
* *vh* = pourcentage de la hauteur du viewport (fenêtre) (100 vh = toute la hauteur)

## B/ Notes

Taille des medias :

* Smartphone = - de 500px
* Tablette = 501px à 768px
* Normal = 769 à 1200px
* Ecran large = + de 1201 px

<link rel="stylesheet" media="screen and (max-width: 768px)" href="mobile.css">

Appel du fichier CSS selon la taille de l’écran

On peut changer la font-size de l’élément racine en changeant la font-size du html

html{

  font-size: 10px;

}

Ici la font-size de l’élément racine devient 10px (ex : 1.5rem = 15px)

# Section 6 : Intro to Flexbox

## A/ Définition

*flex-direction :*

* column = éléments en colonne
* row = éléments en ligne
* reverse = commence par le dernier élément (fonctionne sur colums et row)

*flex-wrap :*

* wrap = les éléments passent à la prochaine ligne/colonne si leur taille min diminue
* non-wrap = les éléments ne passent pas à la prochaine ligne/colonne si leur taille min diminue

*flex-grow :*

Définit le facteur d'expansion d'un élément flexible selon sa dimension principale. Elle indique la quantité d'espace restant que l'élément devrait consommer dans un conteneur flexible relativement à la taille des autres éléments du même conteneur

*flex-shrink :*

Définit le facteur de rétrécissement d'un élément flexible. Si la taille de l'ensemble des éléments flexibles est supérieure à la taille du conteneur, les éléments seront comprimés selon leur facteur

*flex-basis :*

Détermine la base de flexibilité utilisée comme taille initiale principale pour un élément flexible. Cette propriété détermine la taille de la boîte de contenu sauf si une autre boîte est visée par [*box-sizing*](https://developer.mozilla.org/fr/docs/Web/CSS/box-sizing)

*align-items :*

* center = les éléments flexibles sont regroupés au centre
* flex-start = les éléments flexibles sont regroupés au début
* flex-end = les éléments flexibles sont regroupés à la fin
* stretch = Les éléments flexibles sont étirés afin qu’ils prennent toute la place du container

*jutify-content :*

* space-between = espacement égale entre les éléments égales avec 1er éléments au début et dernier à la fin du conteneur
* space-around = espacement égale entre les éléments égales et les bors du conteneur

## B/ Notes

# Section 7 : EdegeLedger Website (Flexbox)

## A/ Définition

## B/ Notes

a href="index.html#about"

Va à la section *about* de la page *index*

# Section 8 : Website Deployment – Shared Host

## A/ Définition

*SSL (*[*Secure Sockets Layer*](https://fr.wikipedia.org/wiki/Secure_Sockets_Layer)*) :*

Protocole de sécurisation des échanges sur Internet, devenu [Transport Layer Security](https://fr.wikipedia.org/wiki/Transport_Layer_Security) en 2001

## B/ Notes

***Port FTP*** : 21

# Section 9 : More CSS Concepts – Advanced Selectors, Animation & More

## A/ Définition

*Sélecteurs :*

* > : sélectionner les enfants directs (a 1 branche dans l’arborescence)
* + : juste après l’élément (mais pas un enfant)

(exemple : div + p 🡺 on sélectionne tous les p comme ça : <div></div> <p></p> ; il est juste après mais ce n’est pas un enfant de la div)

* X [] : sélectionne les éléments X qui ont l’attributs définit dans les crochets. On peut aussi spécifier la valeur de l’attributs (exemple : X[y=z] 🡺on sélectionne tous les éléments X qui ont l’attribut y de valeur z)
* , : permet de sélectionner plusieurs éléments
* X ~ : sélectionne tous les éléments précédés par un X qui a le même parent
* first-child : sélectionne le 1er enfant
* last-child : sélectionne le dernier enfant
* nth-child : sélectionne le nième enfant selon ce qui est spécifier entre parenthèse

Si on met (Xn), on sélectionne les enfants tous les X enfants

Si on met (Yn + z), on sélectionne les enfants tous les Y enfants à partir du zième

(even) sélectionne tous les nièmes enfants pairs (n = 2 ; 4 ; 6 ; etc…)

(odd) sélectionne tous les nièmes enfants impairs (n = 1 ; 3 ; 5 ; etc…)

* after : crée un [pseudo-élément](https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements) à la suite l'élément sélectionné. Il est souvent utilisé pour ajouter du contenu cosmétique à un élément, en utilisant la propriété CSS [content](https://developer.mozilla.org/fr/docs/Web/CSS/content). Par défaut, ce contenu est de type « [en ligne](https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/The_box_model#les_types_de_bo%c3%aete) ».
* before : crée un [pseudo-élément](https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-elements) qui précède l'élément sélectionné. Il est souvent utilisé pour ajouter du contenu cosmétique à un élément, en utilisant la propriété CSS [content](https://developer.mozilla.org/fr/docs/Web/CSS/content). Par défaut, ce contenu est de type « [en ligne](https://developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/The_box_model#les_types_de_bo%c3%aete) ».

Permet aussi de créer des fond sombre en les plaçant avec absolute et un z-index de -1 et un content vide

*box-shadow :*

Ajoute des ombres à la boîte d'un élément via une liste d'ombres séparées par des virgules. Une boîte d'ombre est définie avec des décalages horizontal et vertical par rapport à l'élément, avec des rayons de flou et d'étalement et avec une couleur

/\* offset-x | offset-y | blur-radius | spread-radius | color \*/

box-shadow: 2px 2px 2px 1px rgba(0, 0, 0, 0.2);

* inset : Modifie l'ombre afin qu'elle soit tournée vers l'intérieur du cadre (comme si le contenu était enfoncé dans la boîte). Les ombres tournées vers l'intérieur sont dessinées à l'intérieur de la bordure (même les transparentes), au-dessus de l'arrière-plan mais sous le contenu.
* <offset-x> <offset-y> : Deux valeurs de longueur ([<length>](https://developer.mozilla.org/fr/docs/Web/CSS/length) qui permettent de définir le décalage de l'ombre. <offset-x> définit la distance horizontale du décalage. <offset-y> définit la distance verticale
* <blur-radius> : Une troisième valeur de longueur ([<length>](https://developer.mozilla.org/fr/docs/Web/CSS/length)). Plus cette valeur sera grande, plus le flou de l'ombre sera diffus : l'ombre sera donc plus étalée et plus légère.
* <spread-radius> : Une quatrième valeur de longueur ([<length>](https://developer.mozilla.org/fr/docs/Web/CSS/length)). Les valeurs positives étaleront l'ombre et les valeurs négatives rétréciront l'ombre.
* <color> : Une valeur de couleur ([<color>](https://developer.mozilla.org/fr/docs/Web/CSS/color_value)) qui permet de définir l couleur de l’ombre

*text-shadow :*

Comme box-shadow mais pas inset et spread-raduis

*Variables :*

Entités définies par les développeurs ou les utilisateurs d'une page Web, contenant des valeurs spécifiques utilisables à travers le document. Elles sont initialisées avec des propriétés personnalisées (par exemple --main-color: black;) et accessibles en utilisant la notation spécifique [var() (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/var()) (par exemple : color: var(--main-color))

*Animations :*

* animation-delay : temps avant que l’animation se lance
* animation-duration : temps que met l’animation à s’effectuer
* animation-iteration-count : nombre de fois que l’animation s’effectue
* animation-fill-mode : permet de définir comment l’élément animer se comporte après l’animation
* animation-direction : permet de définir dans quel sens l’animation s’effectue
* animation-timing-function : permet de définir comment l’animation va s’effectuer (vite puis lent, linéaire, lent puis vite, etc…)
* animation-name : permet de lier l’élément animer à une keyframe

*Keyframes :*

Permet aux auteurs de définir les étapes qui composent la séquence d'une animation CSS. Cela permet de contrôler une animation plus finement que ce qu'on pourrait obtenir avec [les transitions](https://developer.mozilla.org/fr/docs/Web/CSS/CSS_Transitions)

* <identifier> : Un nom qui permet d'identifier la keyframes afin de l’utiliser dans le reste du code CSS avec (animation-name : <identifier>)
* from : Indique le point de départ de l'animation (correspond à un avancement de 0%).
* to : Indique la fin de l'animation (correspond à un avancement de 100%).
* [<percentage>](https://developer.mozilla.org/fr/docs/Web/CSS/percentage) : Le pourcentage d'avancement de l'animation auquel l'étape décrite s'applique

*Transitions :*

Permettent de contrôler la vitesse d'animation lorsque les propriétés CSS sont modifiées

* transition-delay : temps avant que la transition se lance
* transition-duration : temps que met la transition à s’effectuer
* transition-property : désigne la propriété qui va être modifier
* transition-timing-function : permet de définir comment l’animation va s’effectuer (vite puis lent, linéaire, lent puis vite, etc…)

*transform :*

Modifie l'espace de coordonnées utilisé pour la mise en forme visuelle. Grâce à cette propriété, il est possible de translater les éléments, de les tourner, d'appliquer des homothéties, de les distordre pour en changer la perspective

* matrix(x, x, x, x, x, x) : Il spécifie une transformation matricielle de type 2-D
* translate(x, y) : Il spécifie une translation sur les axes X et Y
* rotate(angle) : Il precise l’angle de rotation.
* scale(x, y) : Il spécifie la transformation d'échelle le long des axes X et Y
* skew(angle, angle) : Il spécifie la transformation d'inclinaison le long des axes X et Y correspondant aux angles d'inclinaison
* perspective(x) : Il spécifie la perspective d'un élément
* initial : Il initialise l'élément à sa valeur par défaut

## B/ Notes

!important : permet que la spécification CSS écrase toutes les autres

Naturellement les ombres de box vont se placer à droite et en bas, mais si ont mets des valeurs négative, elles peuvent se placer à gauche et en haut

On peut cumuler les ombres des box en les séparant par des virgules

On crée les variables dans la racine pour pouvoir les utilisés partout dans le code CSS. On fait donc :

: root {

--Variable1 : valeur

--Variable2 : valeur

--VariableN : valeur

}

On peut quasiment tous régler dans les keyframes (couleurs, position, etc…) tant que ce sont des propriétés qui sont autorisé pour l’éléments qui est animées à l’aide de la keyframe

Toutes les propriétés de transform écrasent la précédente (exemple : on ne peut pas faire rotate et skew en même temps)

# Section 10 : Mini Projects With KeyFrames, Transitions, etc …

## A/ Définition

## B/ Notes

# Section 11 : CSS Grid

## A/ Définition

## B/ Notes

# Section 12 : NewsGrid Website

## A/ Définition

## B/ Notes

# Section 13 : Website Deployment With Netlify

## A/ Définition

## B/ Notes

# Section 14 : Learning Sass

## A/ Définition

## B/ Notes

# Section 15 : Portfolio Website With Sass

## A/ Définition

## B/ Notes