

# Les langages HTML et CSS

C. BENSARI

# Plan

- Historique
- Le concept des balises
- Structure de base d'un document Html
- Rédiger son texte dans une page web
- Définir des liens dans une pages web
- Afficher des images dans une page web
- Structurer une page web en Html 5
- Le langage de feuilles de style CSS
- Les tableaux
- Les formulaires
- Le multimédia
- L'accessibilité

# Historique de HTML

- HTML est le langage incontournable et universel du Web
- Inventé en 1991 par Tim Berners-Lee afin de gérer et organiser du contenu (Html 1.0)
- Html est la base du fonctionnement des sites Web
- Le navigateur est chargée de transformer le code HTML en éléments qu'on peut voir sur les sites web
- Le langage Html a connu plusieurs versions dans l'histoire:
  - 1995 : Html 2.0
  - 1997 : Html 3.2 et 4.0
  - 2000: Xhtml (abandonné ensuite)
  - De 2007 jusqu'à présent : Html 5

# Le concept des balises

- Le langage Html est dit un langage de balisage. Le contenu (les éléments) d'un site web est définie entre des balises
- Une balise se comporte d'un nom et elle a la forme suivante  
`<nom_balise>`
- Chaque nom de balise a une signification pour le langage Html. Elle indique la nature de l'élément au navigateur
- C'est la navigateur que se charge de faire la transformation de cette balise en un élément visuel
- Les balises peuvent contenir des attributs qui permettent d'ajouter plus d'informations sur l'élément visuel

# Le concept des balises

- Deux types de balises: en paires et orphelines

- Les balises en paires

Elles s'ouvrent, contiennent du texte (ou d'autres éléments) et se ferment plus loin:

**<titre>** ceci est un message **</titre>**

On distingue une balise ouvrante (à gauche) et une balise fermente (à droite) qui indique que le titre se termine ici

Ceci signifie pour la navigateur que tous ce qui est en dehors des deux balises n'est pas un titre

- Les balises orphelines

Utilisées généralement pour des éléments qui ne nécessitent pas de délimiter le début et la fin :

**<image .../>** est équivalent à **<image..>**

# Le concept des balises

- **Les attributs**: ajoutent plus d'informations aux balises? Ils se placent après le nom de balise ouvrante et ont souvent une valeur comme ceci :

`<image src='`photo.jpg`' />`

*L'attribut « **src** » indique au navigateur que l'image se trouve dans le fichier photo.jpg*

# Structure de base d'un document Html

- Un document Html a la structure minimale suivante :

```
<!DOCTYPE html>
```

```
<html>
```

```
  <head>
```

```
    <meta charset='`utf-8`' />
```

```
    <title>Titre</title>
```

```
  </head>
```

```
  <body>
```

```
  </body>
```

```
</html>
```

# Structure de base d'un document Html

- La ligne doctype est essentielle car c'est elle qui indique qu'il s'agit d'une page Html
- La balise html est la balise principale du document html. Elle regroupe tous les éléments de la page web
- La balise head est l'entête du document, elle contient quelques informations générale sur le document. Son contenu n'est pas affiché dans une page web
- La balise body englobe la partie principale de notre page web. Tous ce qui est écrit entre ses deux balises sera affiché dans la page



# Structure de base d'un document Html

- L'encodage (charset) indique au navigateur comment les caractères spéciaux (accents, idéogrammes chinois, japonais, lettres arabes, ..) sont affichés. Il existe plusieurs types d'encodage: UTF-8, ISO-8859-1, OEM 755, .... Mais une seule doit être utilisé par document Html
- La balise *title* qui se trouve à l'intérieur de l'entête head permet d'afficher un titre qui s'affichera sur la fenêtre/onglet

# Structure de base d'un document Html

## Les commentaires

- Un commentaire en Html est un texte permettant de laisser des indications sur le fonctionnement de la page. C'est l'équivalent de mémo qui n'ajoute rien à la page et ne sera pas affiché sur le site
- Pour insérer un commentaire dans une page Html il faut utiliser les deux balises `<!-- mon commentaire -->`
- Il faut savoir que tout le monde peut voir le code Html d'une page web. Il suffit de faire un clique droit sur la page et sélectionner « afficher le code source de la page »
- **Attention !** Il ne faut pas écrire des informations sensibles dans les commentaires !

# Rédiger son texte dans une page web

## Les paragraphes

- Une grande partie du contenu web est du texte. Il existe des balises spécifiques pour insérer du texte sous différentes formes (paragraphes, titres, ..)
- Html propose la balise `<p>` pour délimiter un paragraphe dans une page web

`<p>` mon paragraphe `</p>`

`<p>`: début du paragraphe

`</p>` :fin du paragraphe

Pour sauter une ligne dans une page Html il faut insérer la balise orpheline `<br />`

# Rédiger son texte dans une page web

## Les titres

- Html propose plusieurs balises pour afficher des titres de différentes importances :
  - `<h1> </h1>` : **titre très important**
  - `<h2> </h2>` : **titre important**
  - `<h3> </h3>` : **titre moins important**
  - `<h4> </h4>` : **titre encore moins important**
  - `<h5> </h5>` : **titre beaucoup moins important**
  - `<h6> </h6>` : **titre pas important**
- Il est possible en Html d'utiliser des balises pour mettre en valeur certaines partie du texte. Pour cela il faut utiliser les balises `<em>`, `<strong>`, `<mark>`. Ces balises peuvent être utile pour les moteurs de recherches (google, yahoo , ...)

# Rédiger son texte dans une page web

## Les listes

- Pour mieux structurer du contenu textuel, nous utilisons généralement le système de liste afin d'ordonner le contenu
- Il existe deux types de listes : ordonnées ou non ordonnées
- En Html, créer une liste non ordonnée est simple. Il suffit d'utiliser la balise `<ul>` `</ul>`
- Pour écrire chacun des éléments de la liste, il faut utiliser la balise `<li>` `</li>`. Exemple:

`<ul>`

`<li>France</li>`

`<li>Angleterre</li>`

`<li>Italie</li>`

`</ul>`

- France
- Angleterre
- Italie

# Rédiger son texte dans une page web

## Les listes

- Pour une liste ordonnée en Html, il faut juste remplacer la balise `<ul>` par la balise `<ol>` et ainsi on peut écrire:

`<ol>`

`<li>J'accède au site</li>`

`<li>Je m'authentifie</li>`

`<li>Je consulte mon compte</li>`

`</ol>`

1. J'accède au site

2. Je m'authentifie

3. Je consulte mon compte

# Définir des liens dans une page web

- Le lien hypertexte est une notion fondamentale dans le web. Généralement, on différencie les liens hypertextes par la couleur de la police ainsi que le curseur de la souris au survol du lien
- Pour définir un lien en Html, il suffit d'utiliser la balise `<a>` `</a>` et lui ajouter l'attribut href qui indique vers quelle page conduit le lien. Exemple:

`<a href="http://google.fr">Cliquez ici</a>`

[Cliquez ici](http://google.fr)

- La valeur de l'attribut href peut prendre différentes formes selon la cible du lien

# Définir des liens dans une page web

- Définir un lien vers une autre page de notre site :

`<!-- Page dans le même répertoire que la page actuel -->`

`<a href="`page2.html`">Page2</a>`

`<!-- Page dans le dossier parent du répertoire de la page actuel -->`

`<a href="`../page3.html`">Page3</a>`

`<!-- Page dans un sous dossier du répertoire de la page actuel -->`

`<a href="`sous_dossier/page4.html`">Page4</a>`

- Définir un lien vers une ancre de la page. Une ancre est une sorte de repère qu'on peut mettre dans nos pages:

`<h3 id="`mon_ancre`">Grand titre</h3>`

`<a href="`#mon_ancre`">Aller vers mon ancre</a>`



# Définir des liens dans une page web

- Définir un lien vers un numéro de téléphone (pour téléphone)

`<a href="tel:+654852225">+6 54 85 22 25</a>`

- Un lien pour écrire un email:

`<a href="mailto:person_mail@serveur.com">Envoyer un mail</a>`

- La balise `<a>` possède d'autres attributs qui ajoutent d'autres informations à l'élément Html:

- Title : affiche la valeur de l'attribut au survol du lien
- Target : indique où afficher la ressource liée
  - \_blanc : dans un nouvel onglet
  - \_self : dans la même page (valeur par défaut)

N.B: Lors de l'utilisation de **target**, pensez à ajouter l'attribut `rel="noopener noreferrer"` afin d'éviter la fuite de données

# Afficher des images dans une page web

- Il existe différents formats d'images et le poids de l'image peut changer selon la qualité et le format choisi
- Les images qui circulent sur internet sont tous compressées pour qu'elles soient moins lourdes et rapides à charger
- Le format d'image JPEG est le plus répandu sur le web car son algorithme de compression est efficace de telle sorte que la détérioration de l'image est peu perceptible. Il est préférable de l'utiliser pour les images de type photos
- Le format PNG est le plus adapté pour tous type d'image qui n'est pas une photo. Le format reste cependant moins performant que celui du JPEG

# Afficher des images dans une page web

- Le format GIF est assez vieux mais il garde la particularité du fait qu'il peut être animé
- En Html, pour insérer une image, il suffit d'utiliser la balise orpheline `<img/>` avec deux attributs obligatoires :
  - `src` : permet d'indiquer le chemin vers l'image
  - `alt` : affiche le texte si l'image est introuvable

Exemple

```

```

- Il est possible d'afficher une infobulle sur image pour cela il faut utiliser comme pour les liens, l'attribut `title`

# Afficher des images dans une page web

- Il est possible en Html 5 d'insérer des figures dans une page web
- Une figure peut être une image, une citation un code source,...
- Pour la création d'une figure, il suffit d'utiliser la balise `<figure>`:

```
<figure>
```

```
    <img src='graphe.png' alt='' graphe' />
```

```
    <figcaption> Titre de la figure </figcaption>
```

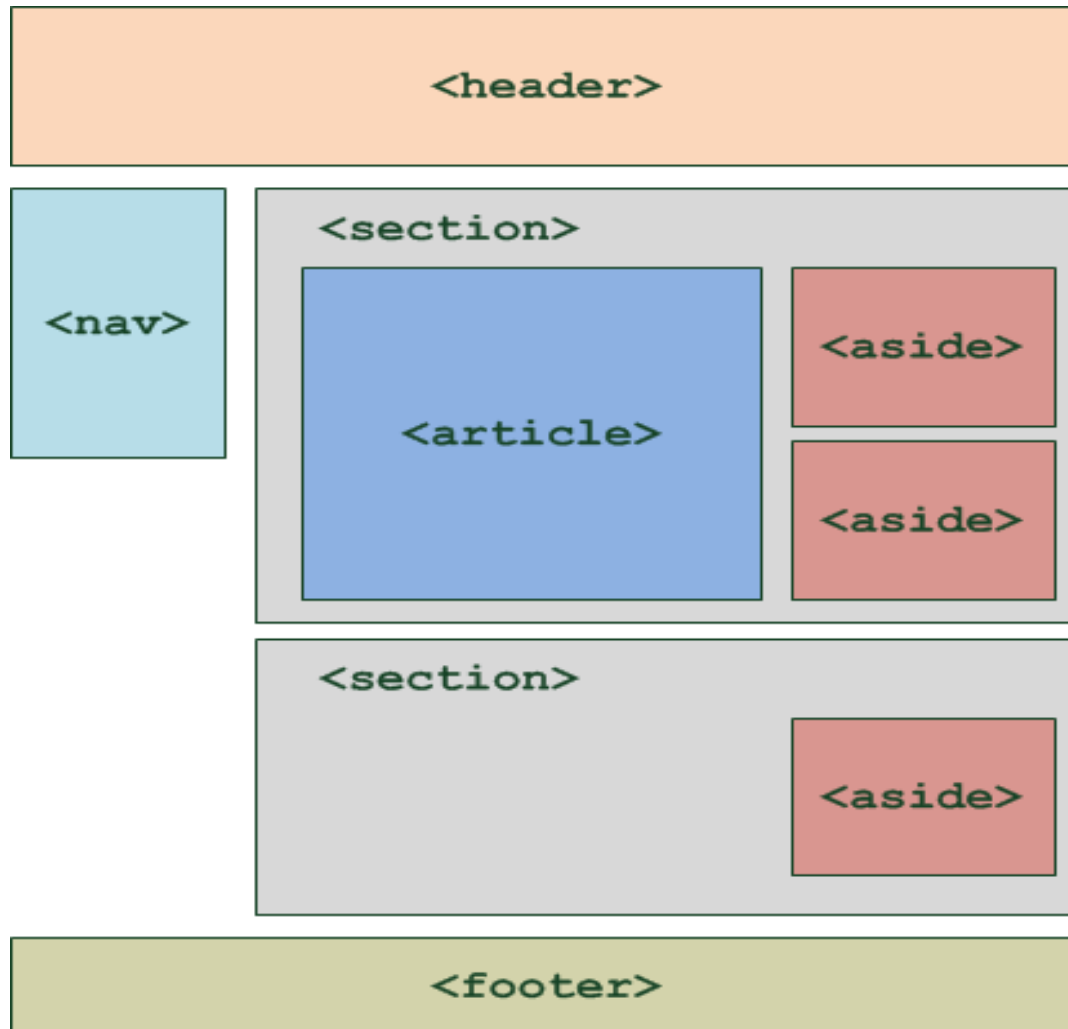
```
</figure>
```

- Sémantiquement, une figure apporte des information sur un texte tandis qu'une image n'y apporte pas

# Structurer une page web en Html 5

- En général, une page web est constituée d'un entête (header), menus de navigation, de différentes sections au centre et d'un pied de page (footer)
- Html 5 a introduit de nouvelles balises permettant la structuration de la page web et d'indiquer au navigateur ce que représente chaque section de la page
- Les balises introduites sont les suivantes :
  - **<header>** : elle contient tous les éléments de l'entête de la page (logo, utilisateur connecté, ..)
  - **<footer>**: elle contient des éléments de fin de pages (contacts, copyright,..)
  - **<nav>** : regroupe les principaux liens de navigation du site
  - **<section>** : elle regroupe des contenus selon une thématique
  - **<aside>** : conçue pour afficher des informations complémentaires. Elle se trouve généralement à droite de l'écran
  - **<article>** : sert à englober une partie autonome de la page qui peut être reprise sur un autre site

# Exemple d'une structure d'une page Html



# Le langage de feuilles de style CSS

# Introduction

- Grace à Html nous sommes capables d'écrire du contenu dans notre site mais il est brut et sans aucun style
- CSS (Cascading Style Sheet) est un langage qui vient compléter Html afin de gérer la mise en forme du contenu (alignement, couleurs, police, ..)
- Au début, le Html fonctionnait tout seul sans CSS. Il existait une balise appelée « font » qui permettait de définir une couleur d'un texte
- Html était devenu complexe avec l'apparition de plusieurs balises de style et le mélange entre le fond et la forme
- La première version de CSS est apparue en 1996 et a connu 4 versions jusqu'à présent (version 1, version 2, version 2,1 et la version 3)



# La compatibilité des navigateurs

- Au début des années 2000, Internet Explorer (version 6) était le navigateur le plus répandu mais sa gestion des CSS était médiocre
- L'arrivée de Mozilla, Chrome et Safari a incité Microsoft de publier des nouvelles versions de IE (IE 7, IE 8, IE10, ..)
- Il faut faire attention aux compatibilités des fonctionnalités CSS (Html) avec les différents navigateurs

# Rédiger des règles de style CSS

- Il est possible d'écrire des règles de styles dans trois endroits différents:
  - Dans un fichier \*.css (méthode recommandée)
  - Dans l'entête <head> du fichier Html
  - Directement dans la balise html concernée via l'attribut **style**
- Pour écrire des règles sur un fichier \*.css nous avons besoin de déclarer ce fichier dans notre fichier Html. Ceci est fait au niveau de la balise d'entête <head> du document html :

```
<head>
```

```
.....
```

```
<link rel='`stylesheet`' href='`style.css`' />
```

```
</head>
```

# Rédiger des règles de style CSS

- Pour écrire du CSS dans le head du fichier html, il suffit d'utiliser la balise `<style> </style>` et de mettre à l'intérieur les règles de style
- Pour écrire du CSS pour une balise donnée, il suffit d'utiliser l'attribut **style** dans la balise ouvrante et de mettre à l'intérieur vos règles séparées par des « ; »

# Rédiger des règles de style CSS

- Pour rédiger une règle de style CSS il faut suivre la syntaxe suivante :

```
balise {  
    propriété1 : valeur1;  
    propriété2 : valeur2;  
}
```

balise : est appelé le « sélecteur CSS »

propriété1, propriété2 : sont les « propriétés CSS »

valeur1, valeur2 : sont des « valeurs de propriétés »

Exemple : mettre tout les paragraphes en couleur bleu avec une police de 18 pixels

```
p {  
    color :blue;  
    font-size : 18px;  
}
```

# Rédiger des règles de style CSS

- Il est possible de combiner plusieurs balises et leur appliquer les même règles CSS :

```
h1, p {  
    color : blue ;  
}
```

- Il est possible d'appliquer des règles de style sur des éléments précis au lieu d'utiliser la règle sur l'ensemble des éléments. Il suffit d'utiliser les attributs HTML « id » et « class » qui fonctionnent sur toutes les balises

# Rédiger des règles de style CSS

Exemple d'utilisation de « id » et « class »

```
<h1 id="titre-principal-meteo">Météo</h1>
<article class="article-meteo">
  <h2>09 Septembre 2019</h2>
  <p>Ciel couvert partiellement..</p>
</article>
```

```
.article-meteo {
  border : 2px solid black;
}

#titre-principal-meteo {
  color : green
}
```

← → ↻ ⓘ Fichier | C:/Users/Utilisateur/Deskto... ☆ ⓘ

**Météo**

**09 Septembre 2019**

Ciel couvert partiellement..

# Rédiger des règles de style CSS

- L'attribut « id » peut prendre une valeur unique dans le document (attention aux espaces et aux caractères spéciaux)
- L'attribut « class » peut prendre plusieurs valeurs séparées par des espaces.
- Les deux attributs sont quasiment identiques. La seule différence est que la valeur de « id » doit être unique dans le document (on ne peut pas avoir deux balises qui ont la même valeur de l'attribut id)
- Les valeurs de l'attribut « class » peuvent être utilisées par plusieurs balises

# Rédiger des règles de style CSS

- Nous sommes capables d'appliquer des règles à des balises précises, mais comment peut-on appliquer des règles sur une partie du texte ?
- Pour remédier à ce genre de problème Html propose deux balises appelées « Balises universelles » : la balise `<span>` et la balise `<div>`
- Ces balises n'ont aucune signification particulière, elles ont une petite différence mais significative :
  - Div : est un élément **block** qui force un retour à la ligne pour l'élément suivant
  - Span : est un élément **inline** qui ne force pas un saut de ligne pour l'élément suivant



# Rédiger des règles de style CSS

- Exemple de mise en forme d'une partie d'un texte :

```
<!-- quelque part dans mon fichier html -->
```

```
<p class=''resume''>
```

```
    <span class=''salutations''> Bonjour </span> à tous
```

```
</p>
```

```
*/ Quelques part dans mon fichier css */
```

```
.salutations {
```

```
    color: red;
```

```
}
```

# Rédiger des règles de style CSS

## Autres sélecteurs

- Il existe d'autres types de sélecteurs en CSS :
  - **\*** : il applique les règles à tous les éléments de la page
  - **A B**: il applique les règles sur les balises « B » qui se trouvent à l'intérieur d'une balise « A »
  - **A + B**: il applique les règles sur les éléments « B » qui se trouvent juste après l'élément « A »
  - **A [title]** : il applique les règles sur tous les éléments « A » qui ont un attribut **title**
  - **B[title = ``Cliquez ici``]** : il applique les règles sur tous les éléments B qui ont un attribut title égale à la valeur « Cliquez ici »
  - **A[title \*= ``ici``]**: il applique les règles à tous les éléments A qui ont un attribut title avec une valeur contenant le mot « ici »
  - Il existe encore d'autres types de sélecteurs..

**Note:** A et B sont des sélecteurs CSS

# Formatage du texte

- Le langage CSS nous donne la possibilité de mettre en forme le texte de notre page web (taille, police, forme, couleur..)
- La modification de la taille de la police se fait avec la propriété « font-size » qui peut prendre deux types de valeurs :
  - Une valeur **absolue** : l'unité de grandeur pour ce type est le pixel (px). On peut écrire par exemple : `font-size : 15px;`
  - Une valeur **relative** (recommandé): il y a plusieurs moyens d'indiquer une valeur relative :
    - Mots en anglais : xx-small, x-small, small, medium, large, x-large, xx-large. On peut écrire par exemple : `font-size : xx-small;`
    - X em : em (emphemeral unit) représente la taille actuelle de la police. Pour agrandir la taille de la police on peut écrire « 2 em » par exemple et pour minimiser on peut écrire « 0.5 em ». On peut écrire par exemple : `font-size : 0.3 em;`

# Formatage du texte

- La modification de la police se fait avec la propriété CSS « font-family ». Le problème qui se pose est que nous ne sommes pas sûre si l'internaute possède la police choisie sur son navigateur !
  - Depuis CSS3, il est possible de préciser plusieurs polices et le navigateur prendra la première qu'il possède :

```
p {  
    font-family : Impact, ``Arial Black``, Arial, Verdana, Serif;  
}
```
- Pour la mise en forme du texte (italique, gras, souligné, ..) plusieurs propriétés CSS peuvent être utilisées:
  - **font-style** : permet le choix du style du texte. Peut prendre italic, oblique, normal
  - **font-weight** : permet le choix de l'épaisseur bold, normal, ..
  - **text-decoration** : permet le choix de la décoration du texte underline, overline, ..

# Formatage du texte

- CSS nous permet de préciser l'alignement du texte (gauche, droite, centré et justifié), Il suffit d'utiliser la propriété « `text-align` » qui peut prendre l'une des valeurs : `left`, `center`, `right` ou `justify`



**Attention !** L'alignement et la redimension ne fonctionnent que sur des éléments de type **block**

# Formatage du texte

- Il est possible de faire flotter un élément autour d'un texte. Il suffit d'utiliser la propriété « **float** » qui peut prendre que deux valeurs possibles (right ou left). Exemple:

<p>



Un long paragraphe.. Un long paragraphe.. Un long paragraphe..

Un long paragraphe.. Un long paragraphe..

</p>

**#img-flottante** { **float** : left; } <== dans le fichier CSS



Un long paragraphe.. Un long  
paragraphe.. Un long  
paragraphe.. Un long  
paragraphe.. Un long paragraphe.. Un  
long paragraphe.. Un long  
paragraphe.. Un long paragraphe.. Un  
long paragraphe.. Un long  
paragraphe.. Un long  
paragraphe.. Un long  
paragraphe.. Un long  
paragraphe..

# Formatage du texte

- Pour changer la couleur d'un texte il suffit d'utiliser la propriété « `color` ». Les valeurs que peut prendre la propriété peuvent être exprimées de différentes manières :
  - Nom de la couleur (en anglais): la liste est limitée à 16 couleurs
  - La notation en hexadécimal : faire suivre le caractère « # » de 6 chiffres ou lettres (chiffres entre 0 et 9 et des lettres entre A et F). Exemples : `#259847`, `#ffffff`, `#000000`, `#ab98fc`
  - La méthode RGB (Red Green Blue): utilisation d'une suite de trois nombres entre 0 et 255 de la manière suivante :  
`color : rgb(240, 253, 2);`
- Pour changer la couleur d'arrière plan d'un texte, il suffit d'utiliser la propriété « `background-color` » qui fonctionne de la même manière que la propriété « `color` »

# Le modèle des boîtes

- Une page html peut être imaginée comme une succession et un empilement de boîtes appelées « blocs »
- En Html, la plupart des balises peuvent être catégorisées dans deux types :
  - Bloc : Une balise de type block ajoute automatiquement un saut de ligne avant et après la balise
  - Inline : se trouve systématiquement dans une balise de type block et ne crée pas de sauts de lignes
- Il est possible d'imbriquer des balises de type bloc pour avoir plus de possibilités de mise en page du site
- Il n'est pas possible d'imbriquer une balise de type bloc dans une balise de type inline



# Le modèle des boîtes

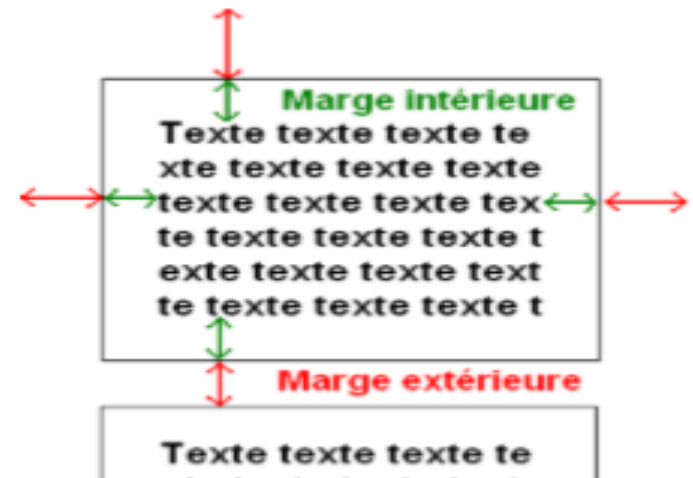
- Exemples de balises de types bloc : `<div>`, `<p>`, `<footer>`, `<h1>`, `<h2>`, `<article>`, ..
- Exemples de balises de type inline : `<span>`, `<em>`, `<strong>`, `<mark>`, `<a>`, `<img>`, ..
- Les balises de type bloc dispose d'une dimension précise et donc de deux propriétés CSS :
  - `width` : représente la largeur du bloque exprimée en pixel ou en pourcentage (100% par défaut)
  - `Height` : c'est la hauteur du bloc exprimée aussi en px ou en pourcentage
- Il est possible aussi d'indiquer des limites de dimension avec les propriétés « `min-width` », « `max-width` », « `min-height` » et « `max-height` »

# Les marges

- Les éléments bloc possèdent deux types de marges :
  - Marges extérieures : exprimées avec la propriété CSS « `margin` », elles indiquent l'espace entre la bordure et les blocs qui se trouvent autour
  - Marges intérieures : exprimées avec la propriété CSS « `padding` », elles indiquent l'espace entre le texte et la bordure

Exemple d'utilisation sur les paragraphes:

```
p {  
    width : 50%;  
    border: 1px solid black;  
    text-align : justify;  
    padding : 20px;  
    margin : 25px;  
}
```



# Les marges

- En utilisant les propriétés « margin » et « padding » tout court, les marges sont ajoutées sur les quatre côtés de l'élément en question
- Il est possible d'utiliser les propriétés suivantes :
  - padding-top : marge intérieure en haut
  - padding-bottom : marge intérieure en bas
  - padding-left : marge intérieure à gauche
  - padding-right : marge intérieure à droite
  - margin-top : marge extérieure en haut
  - margin-bottom : marge extérieure en bas
  - margin-left : marge extérieure à gauche
  - margin-right : marge extérieure à droite

# Les marges

- Il est possible de centrer des blocs en utilisant uniquement la propriété « **margin** : auto »
- À partir du moment où on fixe des dimensions on risque d'avoir des débordements (de texte par exemple). Pour remédier à ce problème, CSS nous propose la propriété « **overflow** » qui prend l'une des valeurs suivantes :
  - **visible** : elle affichera le débordement (valeur par défaut)
  - **hidden** : cache le débordement
  - **scroll** : cache le débordement et crée une barre de défilement
  - **auto** : c'est au navigateur de décider
- Pour les mots trop longs susceptibles de dépasser la limite, il faut utiliser la règle CSS « **word-wrap** : break-word; » afin de faire un retour à la ligne avec la suite du mot

# Le positionnement

- Pour repositionner des éléments en Html il faut faire des combinaisons des propriétés CSS suivantes :
  - **display** : elle prends les valeurs : inline, block, inline-block, none et bien d'autres (flex, grid à voir !)
  - **float** : nous permet de positionner des éléments blocs l'un à côté de l'autre (avec quelques inconvénients)
  - **margin** et padding pour faire des décalages
- La propriété « display » est puissante, elle peut changer la nature d'un élément html (exemple: transformer l'élément <a> qui est un élément inline en un élément bloc)
- La propriété « display » peut aussi cacher un élément avec la valeur **none**

# Le positionnement

```
<div>
  <article class="article-meteo">
    <h1 id="titre-principal-meteo">Météo</h1>
    <article class="meteo-du-jour meteo-test meteo-aujourd'hui">
      <h2>09 septembre 2019</h2>
      <p data-test="text">Ciel partiellement couvert</p>
    </article>
    <article class="meteo-du-jour meteo-test meteo-plutard">
      <h2>10 septembre 2019</h2>
      <p data-test="text">Temps pluvieux</p>
    </article>
    <article class="meteo-du-jour meteo-test meteo-plutard">
      <h2>11 septembre 2019</h2>
      <p data-test="cool">Ensoleillé</p>
    </article>
    <article class="meteo-du-jour meteo-meteo meteo-plutard">
      <h2>11 septembre 2019</h2>
      <p data-test="cool">Ensoleillé</p>
    </article>
  </article>
</div>
```

```
.meteo-du-jour {
  display: inline-block;
  margin-left : 15px;
}
```



Sur une largeur réduite de l'écran

# Le positionnement

- La valeur **inline-block** de la propriété « **display** » permet d'afficher des éléments de type block l'un à côté de l'autre comme si ils étaient des éléments de type inline et donne la possibilité de pouvoir modifier leur dimension (car il n'est pas possible de redimensionner des éléments inline)
- Utiliser cette valeur positionnera les éléments sur une même ligne de base (baseline). Pour changer ce comportement, il faut utiliser la propriété « **vertical-align** » qui peut prendre l'une des valeurs suivantes :
  - **baseline** : alignement sur la base (par défaut)
  - **top**: aligne en haut
  - **middle** : aligne au centre
  - **bottom** aligne en bas
  - Une valeur en pourcentage à partir de la base

# Le positionnement

- Il est possible de positionner des éléments autrement :
  - Positionnement absolu : nous permet de positionner l'élément n'importe où dans la page
  - Positionnement fixe : même fonctionnement que l'absolu sauf que l'élément reste fixé même si on descend plus bas dans la page
  - Positionnement relatif : permet de positionner un élément par rapport à sa position normal
- Ces types de positionnement sont faits avec la propriété «**position**» qui peut prendre les valeurs (absolute, fixed, relative)
- La valeur « absolute » ne peut pas fonctionner toute seule (il va falloir préciser où positionner l'élément). Il suffit d'utiliser les 4 propriétés CSS « left », « right », « bottom » et « top » avec des valeurs en px (exemple: si right : 0px; l'élément sera complètement à droite)



# Le positionnement

```
.article-meteo {  
  height : 45%;  
  width : 50%;  
  border : 2px solid red;  
  position : absolute;  
  left : 2%;  
  top: 2%;  
}
```



# Le positionnement

- Le positionnement fixe fonctionne exactement de la même manière que celui de l'absolu mais garde l'élément dans le même endroit même si on descend tout en bas de la page
  - Pour le positionnement absolu ou fixe, le point de départ est à la position 0 à gauche ,0 en haut de la page si l'élément parent n'est pas positionné en relatif, absolu ou fixe
- Le positionnement relatif utilise également les propriétés utilisées dans les autres types de positionnement sauf que le point de départ ce n'est pas à gauche en haut de l'élément **parent** mais plutôt en haut à gauche de l'élément lui-même en position normale

# Le positionnement

<p>

Un long paragraphe.. Un long paragraphe.. Un long paragraphe..  
Un long paragraphe.. <strong>Un long paragraphe</strong> Un long paragraphe..  
.. Un long paragraphe.. Un long paragraphe..  
Un long paragraphe.. Un long paragraphe.. Un long paragraphe..  
Un long paragraphe.. Un long paragraphe.. Un long paragraphe..

</p>

```
strong {  
  color : white;  
  background-color : blue;  
  position : relative;  
  left : 30px;  
  top : 20px;  
}
```

he.. Un long pa  
aphe.. UnUn long paragraphe long t

```
strong {  
  color : white;  
  background-color : blue;  
  position : relative;  
  right : 30px;  
  bottom : 20px;  
}
```

Un long paragraphe  
graphe..  
ragraphe.. Un long paragraphe.

```
strong {  
  color : white;  
  background-color : blue;  
}
```

ne.. Un long paragraphe Un l  
phe.. Un long paragraphe.. Un

# Les tableaux en Html

- Les tableaux sont des moyens efficaces pour la structuration du contenu
- En Html, pour créer un tableau on utilise la paire de balises `<table>` `</table>` de type block
- En Html, le tableau se construit ligne par ligne. Pour insérer une nouvelle ligne dans le tableau, on utilise la balise `<tr>` `</tr>`
- Dans chaque ligne on peut insérer des cases (colonnes) avec la balise `<td>` `</td>`
- Par défaut, le tableau n'a pas de bordure, il faut en ajouter avec la propriété CSS « border » sur chaque `<td>`
- Pour éviter la présence de bordures en doubles, il suffit d'utiliser la propriété CSS « border-collapse : collapse » sur la balise `<table>`

# Les tableaux en Html

- Pour donner des noms aux colonnes d'un tableau, il faut insérer des cases avec la balise `<th></th>` dans la première ligne du tableau (ne pas oublier les bordure pour les nouveaux types de cellules)
- Pour donner un titre au tableau, il faut utiliser la balise `<caption>` au début du tableau avant la première ligne
- Par défaut, le titre est mis en haut du tableau. Il est possible de changer la position avec la propriété CSS « `caption-side` » qui prend l'une des valeurs : `top` ou `bottom`
- Il est fréquent de rencontrer des tableaux plus complexes, avec des fusion de colonnes ou de lignes, des tableaux divisés en plusieurs parties

# Des tableaux plus complexes

- En Html, il est possible de diviser un tableau en plusieurs parties grâce aux balises `<thead>`, `<tbody>` et `<tfoot>` ceci donnera plus du sens à notre page pour le navigateur
- Dans le code on affichera l'entête du tableau ensuite le pied du tableau et enfin le corps du tableau et le navigateur se chargera d'afficher les éléments aux bons endroits
- Dans certains cas, les tableaux ont certaines cellules fusionnées. Il existe deux types de fusions :
  - Fusion de colonnes : c'est le fait de regrouper deux colonnes en une seule. Ceci est fait en html grâce à l'attribut « colspan » sur le `<td>`
  - Fusion de lignes : le regroupement de deux ligne en une seule. E fait grâce à l'attribut Html « rowspan » sur le `<td>`

# Les formulaires

- La principale utilité d'un site web est d'afficher du contenu dans différents formats. Depuis quelques années, le web est devenu le cœur des systèmes d'informations pour les entreprises
- Grâce au web nous sommes capables d'afficher du contenu de manière dynamique en utilisant la notion des bases de données mais ce n'est pas seulement, nous sommes aussi capables d'en créer ces données, les modifier et les supprimer
- La création ou la modification du contenu est généralement fait e à partir d'un formulaire Html
- Un formulaire Html peut être imaginé exactement comme un formulaire papier d'une administration. Il contient des champs à remplir, des cases à cocher, ..etc

# Les formulaires

- Avec les formulaires nous arrivons à découvrir les limites du langage Html car en Html, on ne peut pas analyser les informations saisies par l'internaute et les stocker dans une base de données. Ce genre d'opérations est fait généralement avec des technologies du côté serveur : Java, PHP, ..
- Avant de découvrir ces technologies, il faut tout d'abord commencer par afficher des formulaires aux internautes avec Html et CSS
- Pour insérer un formulaire dans une page Html il suffit d'utiliser la balise `<form></form>` et de mettre à l'intérieur les différents éléments de saisie et de sélection



# Les formulaires

- Pour la saisie de formulaire, deux questions importantes se posent :
  - Par quel moyen envoyer les informations saisies ?
  - Une fois envoyées, comment les traiter ?
- Pour répondre à ces deux questions, il faut ajouter deux attributs à la balise **<form>** :
  - **method** : méthode HTTP utilisée. Il prend comme valeurs :
    - **GET** : envoyer les informations via l'URL du navigateur (peu adaptée)
    - **POST** : les données ne sont pas envoyées via l'URL mais plutôt dans le corps (body) de la requête. Cette méthode est la plus utilisée pour les formulaires
  - **action** : C'est le chemin de la page ou du programme qui va traiter les informations une fois reçues (répond à la question 2)

# Les formulaires

- Pour insérer des champs de saisie, Html propose deux types d'éléments :
  - Champs mono-ligne : on ne peut écrire qu'une seule ligne
  - Champs multilignes : permet d'écrire une quantité importante de texte et en plusieurs lignes
- Pour insérer une zone de texte d'une seule ligne il suffit d'utiliser la balise orpheline **<input/>** avec un attribut « **type** » qui prendra la valeur **text**
- Un attribut essentiel à ajouter dans le champs input, il s'agit de l'attribut « **name** » qui permet de faire l'association avec la valeur saisie. La valeur de l'attribut name n'apparaît pas sur la page

# Les formulaires

- Un attribut important à ajouter à l'élément input : « id » qui va nous permettre de lier ce champs de texte avec le libellé du champs
- Nous arrivons à avoir le champs input dans le formulaire mais que faut-il saisir ?
- Pour répondre à cette question nous utilisons un nouveau élément html appelé « label » qui va utiliser un attribut « for » qui prendra la même valeur « id » du champs de texte (le lien est fait !)
- Il est possible d'ajouter d'autres attributs à l'élément « input » comme le « size », « maxlength », « value », « placeholder », ..

# Exemple d'input monoligne

```
<label for="nomPersonne">Saisir votre nom : </label>  
<input id="nomPersonne" placeholder="Saisissez votre nom ici"  
size="50" maxlength="10" type="text" name="nom" />
```

Saisir votre nom :



# Les formulaires

- Il est possible de mettre le type de l'élément « input » en **password** cela permet au champs de se comporter comme une zone de mot de passe (la valeur saisie ne sera pas visible sur l'écran)
- Il est possible d'utiliser plusieurs types pour le champs input : email, url, tel, number, range (curseur), color, date, time, search, ..
- Les types range et number peuvent utiliser les attributs min et max pour fixer des limites aux valeurs saisie
- Pour insérer un champ de type multilignes il faut utiliser l'élément `<textarea>` qui possède des attributs « rows » et « cols » pour préciser la taille de la zone de saisie

# Autres exemples d'input monoligne

```
<label for="mailPersonne">Adresse mail : </label>
<input id="mailPersonne" size="50" maxlength="100" type="email"
placeholder="Votre adresse mail ici" name="email" />
<br/>
<label for="tlfPersonne">Téléphone : </label>
<input id="tlfPersonne" size="50" maxlength="13" type="tel"
placeholder="Votre téléphone ici" name="telephone" />
<br/>
<label for="pswPersonne">Mot de passe : </label>
<input id="pswPersonne" placeholder="Votre password ici" size="50"
maxlength="100" type="password" name="mot_de_passe" />
<br/>
<label for="nombre">Saisir un nombre : </label>
<input id="nombre" min="0" max="100" type="number" name="numero" />

<br/>
<label for="nombre">Choisir un nombre : </label>
<input id="nombre" min="0" max="100" type="range" name="intervalle" />
```

Adresse mail :

Téléphone :

Mot de passe :

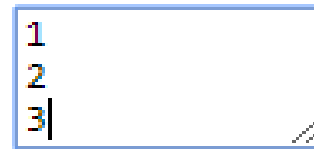
Saisir un nombre :

Choisir un nombre :

# Exemple d'un input multilignes

```
<label for="multi">Saisir un texte :</label>  
<textarea id="multi" rows="3" cols="10"></textarea>
```

Saisir un texte :



1  
2  
3|

# Les formulaires

## Les éléments d'options

- Les éléments d'options permettent à l'utilisateur de faire des choix parmi une liste
- Pour créer un élément case à cocher il suffit de mettre le type du champ input à « checkbox »
- Pour afficher les différentes options, il faut ajouter l'élément `<label />` avec l'attribut « for » et un texte qui sera affiché à côté de la case à cocher. Répéter ces deux champs plusieurs fois pour autant d'options
- Pour chaque input on doit avoir un « name » différent
- Il est possible de mettre une case cochée par défaut, il suffit d'ajouter l'attribut « checked » sans valeur



# Exemple d'un input de type checkbox

```
<input type="checkbox" name="France" id="france" checked />
```

```
<label for="france">France</label>
```

```
<input type="checkbox" name="Bresil" id="bresil" />
```

```
<label for="bresil">Bresil</label>
```

```
<input type="checkbox" name="Australie" id="australie"/>
```

```
<label for="australie">Australie</label>
```

☒ France ☐ Bresil ☐ Australie

# Les formulaires

## Les éléments d'options

- Pour créer des zones d'options à choix unique parmi une liste de possibilité il suffit d'utiliser le type « radio » de l'élément input
- Les radios fonctionnent de la même manière que les checkboxes avec une petite différence :
  - Il faut regrouper plusieurs input avec la même valeur de l'attribut « name »
- L'attribut « checked » peut être aussi utiliser pour sélectionner une option par défaut

# Exemple d'input radio

```
<input type="radio" name="Fruit" id="orange" checked />  
<label for="orange">Orange</label>  
<input type="radio" name="Fruit" id="fraise" />  
<label for="fraise">Fraise</label>  
<input type="radio" name="Fruit" id="banane"/>  
<label for="banane">Banane</label>
```

• Orange • Fraise • Banane

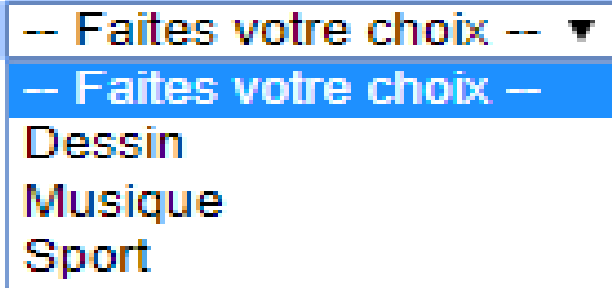
# Les formulaires

## Les listes déroulantes

- Les liste déroulantes sont aussi un moyen de faire un choix pour l'internaute
- En Html, pour insérer une liste déroulante il suffit d'utiliser la balise `<select></select>` avec un attribut « name »
- Pour ajouter les différents choix de la liste il faut insérer plusieurs éléments `<option> </option>` avec l'attribut « value » (obligatoire)
- Pour faire un choix par défaut, il suffit de mettre l'attribut « selected » qui ne prend pas de valeur

# Exemple d'un select

```
<label for="hobbies" />
<select id="hobbies" name="hobbies">
  <option value="">-- Faites votre choix --</option>
  <option value="dessin">Dessin</option>
  <option value="musique">Musique</option>
  <option value="sport">Sport</option>
</select>
```



The image shows a web browser window with a dropdown menu. The menu is open, displaying four options: "-- Faites votre choix --", "Dessin", "Musique", and "Sport". The first option is highlighted in blue, and a small downward arrow is visible next to it.

# Les formulaires

## Les listes déroulantes

- Il est possible de regrouper plusieurs options dans le même groupe. Pour le faire il faut insérer une balise `<optgroup>` `</optgroup>` qui va contenir les balises `<option>` groupées
- Il est préférable de donner un nom pour chaque regroupement. Ceci se fait avec l'attribut « label » sur la balise `<optgroup>`

# Exemple d'un select regroupé

```
<label for="hobbies" />
<select id="hobbies" name="hobbies">
  <option value="">-- Faites votre choix --</option>
  <optgroup label="Dessin">
    <option value="Artistique">Artistique</option>
    <option value="3D">3D</option>
    <option value="Industriel">Industriel</option>
  </optgroup>
  <optgroup label="Musique">
    <option value="Rock">Rock</option>
    <option value="Douce">Douce</option>
    <option value="Jazz">Jazz</option>
  </optgroup>
  <optgroup label="Sport">
    <option value="Tennis">Tennis</option>
    <option value="Golf">Golf</option>
    <option value="Boxe">Boxe</option>
  </optgroup>
</select>
```

-- Faites votre choix --

**Dessin**

Artistique

3D

Industriel

**Musique**

Rock

Douce

Jazz

**Sport**

Tennis

Golf

Boxe

-- Faites votre choix -- ▼

# Les formulaires

## Regrouper des champs d'un formulaire

- Généralement, les formulaires (papier) que nous remplissons sont divisées en plusieurs parties. C'est une manière de structurer le formulaire
- En Html, il est possible de structurer les formulaires en utilisant la balise `<fieldset></fieldset>` qui peut regrouper plusieurs champs
- Chaque balise `<fieldset>` peut avoir un titre pour décrire par exemple la catégorie des champs. Ceci est fait avec la balise `<legend></legend>` qui doit se trouver à l'intérieur de la balise `<fieldset>`



# Exemple de formulaire avec des regroupements

```
<form action="traitement.php" method="get">
  <fieldset>
    <legend>Vos coordonnées</legend>
    <label for="nomPersonne">Saisir votre nom : </label>
    <input id="nomPersonne" placeholder="Votre nom ici"
      size="50" maxlength="10" type="text" name="nom" />
    <br/>
    <label for="prenomPersonne">Saisir votre prénom : </label>
    <input id="prenomPersonne" maxlength="10" name="prenom"
      size="50" type="text" placeholder="Votre prénom ici"/>
  </fieldset>

  <fieldset>
    <legend>Votre pays de naissance</legend>
    <input type="checkbox" name="France" id="france" checked />
    <label for="france">France</label>
    <input type="checkbox" name="Bresil" id="bresil" />
    <label for="bresil">Bresil</label>
    <input type="checkbox" name="Australie" id="australie"/>
    <label for="australie">Australie</label>
  </fieldset>
</form>
```

Vos coordonnées

Saisir votre nom :

Saisir votre prénom :

Votre pays de naissance

☒ France ☐ Bresil ☐ Australie

# Les formulaires

## Quelques particularités

- Il est possible d'indiquer au navigateur sur quel champ placer le curseur à l'ouverture de la page du formulaire. Il suffit d'utiliser l'attribut « autofocus » dans l'élément `<input>` souhaité

```
<fieldset>
  <legend>Vos coordonnées</legend>
  <label for="nomPersonne">Saisir votre nom : </label>
  <input id="nomPersonne" placeholder="Votre nom ici" autofocus
        size="50" maxlength="10" type="text" name="nom" />
  <br/>
  <label for="prenomPersonne">Saisir votre prénom : </label>
  <input id="prenomPersonne" maxlength="10" name="prenom"
        size="50" type="text" placeholder="Votre prénom ici"/>
</fieldset>
```

Vos coordonnées

Saisir votre nom :

Saisir votre prénom :

# Les formulaires

## Quelques particularités


- Pour mettre des champs obligatoires dans un formulaire, il suffit d'utiliser l'attribut « required » sur la balise <input> du champ en question. Le navigateur indiquera qu'il faut renseigner ce champ si l'internaute tente de valider le formulaire sans remplir ce champs

```
<label for="prenomPersonne">Saisir votre prénom : </label>  
<input id="prenomPersonne" maxlength="10" name="prenom" required  
size="50" type="text" placeholder="Votre prénom ici"/>
```

Vos coordonnées

Saisir votre nom :

Saisir votre prénom :

 Veuillez renseigner ce champ.

# Les formulaires

## Quelques particularités


- Si à la validation un champ n'est pas renseigné ou est mal renseigné il est possible d'indiquer des style sur ce champ en question avec les pseudo formats : « :required » « :invalid »

```
<label for="emailPersonne">Saisir votre email : </label>  
<input id="emailPersonne" placeholder="Votre email ici"  
      size="50" maxlength="100" type="email" name="email" />
```

```
input:invalid {  
    background-color : red;  
}
```

Vos coordonnées

Saisir votre nom :	Dupond
Saisir votre prénom :	David
Saisir votre email :	ab

 Veuillez inclure "@" dans l'adresse e-mail. Il manque un symbole "@" dans "ab".

# Les formulaires

## Quelques particularités

- Il est aussi possible de donner un style au champ sur lequel est le focus. Il suffit d'utiliser le pseudo format CSS « :focus »

```
input:focus {  
    color : rgb(43, 255, 189);  
}
```

Vos coordonnées

Saisir votre nom : Dupond

Saisir votre prénom : David

Saisir votre email : Votre email ici

# Les formulaires

## Le bouton d'envoi de données

- Pour envoyer les données, nous avons besoin que l'utilisateur fasse une action (click sur un bouton ...)
- Pour ajouter le bouton d'envoi de données il suffit d'ajouter une nouvelle balise `<input>` avec l'un des types suivants :
  - `submit` : envoi des données à la page indiquée dans « action » de la balise `<form>`
  - `image` : fonctionne comme le `submit` mais on peut utiliser une image comme bouton (il faut penser à ajouter le `src`)
  - `button` : qui n'aura aucun effet au click (il faut le programmer en JavaScript)
- Il est possible d'ajouter un bouton qui permet de réinitialiser le formulaire. Il suffit d'utiliser un élément `input` avec le type « reset »

# Exemple de boutons d'envoi du formulaire

```
<input type="image" src="image.png"/>  
<!-- <span> OU <span> -->  
<!-- <input type="submit" value="Envoyer"/> -->  
<!-- <span> OU <span> -->  
<!-- <input type="button" value="Envoyer si Possible"/> -->  
<input type="reset" value="Réinitialiser"/>
```

Vos coordonnées

Saisir votre nom : Dupond

Saisir votre prénom : David

Saisir votre email : David.Dupond@gmail.com



Réinitialiser

# Envoi des données du formulaire

## Méthode GET

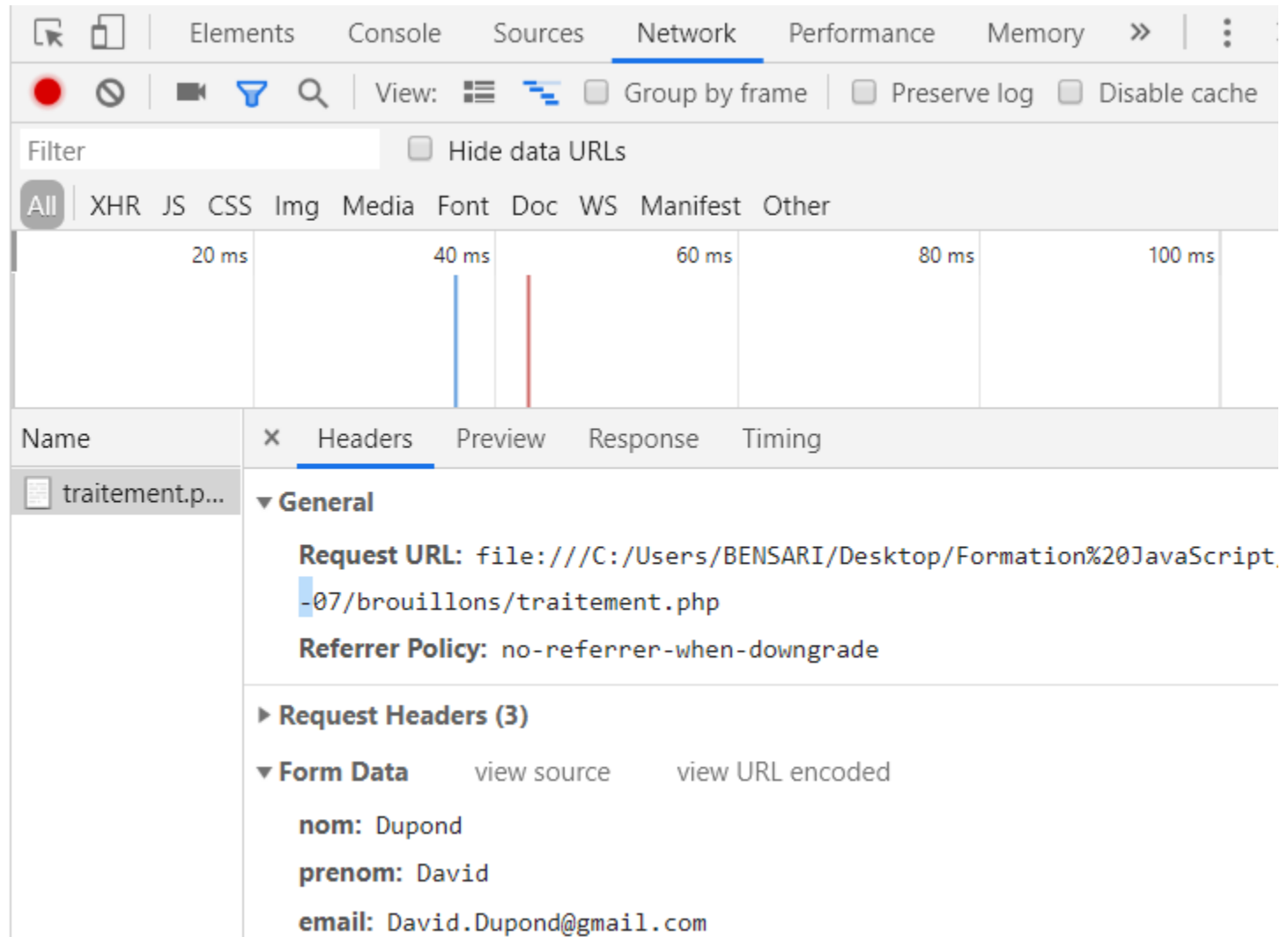
The screenshot shows a web browser's developer tools with the Network tab selected. A request to `traitement.php` is highlighted. The request is a GET method. The URL is `file:///C:/Users/BENSARI/Desktop/Formation%20JavaScript/1-07/brouillons/traitement.php?nom=Dupond&prenom=David&email=David.Dupond%40gmail.com&x=31&y=19`. The Referrer Policy is `no-referrer-when-downgrade`. The Query String Parameters are `nom: Dupond`, `prenom: David`, and `email: David.Dupond@gmail.com`.

Network tab showing a request to `traitement.php`. The request is a GET method. The URL is `file:///C:/Users/BENSARI/Desktop/Formation%20JavaScript/1-07/brouillons/traitement.php?nom=Dupond&prenom=David&email=David.Dupond%40gmail.com&x=31&y=19`. The Referrer Policy is `no-referrer-when-downgrade`. The Query String Parameters are `nom: Dupond`, `prenom: David`, and `email: David.Dupond@gmail.com`.



# Envoi des données du formulaire

## Méthode POST



The screenshot shows the Network tab of a web browser's developer tools. The 'Headers' sub-tab is selected for the request to 'traitement.p...'. The 'General' section shows the 'Request URL' as 'file:///C:/Users/BENSARI/Desktop/Formation%20JavaScript-07/brouillons/traitement.php' and the 'Referrer Policy' as 'no-referrer-when-downgrade'. The 'Request Headers (3)' section is expanded, showing 'Form Data' with the following values: 'nom: Dupond', 'prenom: David', and 'email: David.Dupond@gmail.com'. The 'Timing' section shows a total time of 100 ms, with a breakdown of 20 ms for the request and 40 ms for the response.

Name	Headers	Preview	Response	Timing
traitement.p...	<b>General</b> <b>Request URL:</b> file:///C:/Users/BENSARI/Desktop/Formation%20JavaScript-07/brouillons/traitement.php <b>Referrer Policy:</b> no-referrer-when-downgrade <b>Request Headers (3)</b> <b>Form Data</b> view source   view URL encoded <b>nom:</b> Dupond <b>prenom:</b> David <b>email:</b> David.Dupond@gmail.com			20 ms   40 ms   60 ms   80 ms   100 ms

# Encore du Html !

- Les formulaires cachés :

```
<input type="hidden" value="215436" name="numeroClient"/>
```

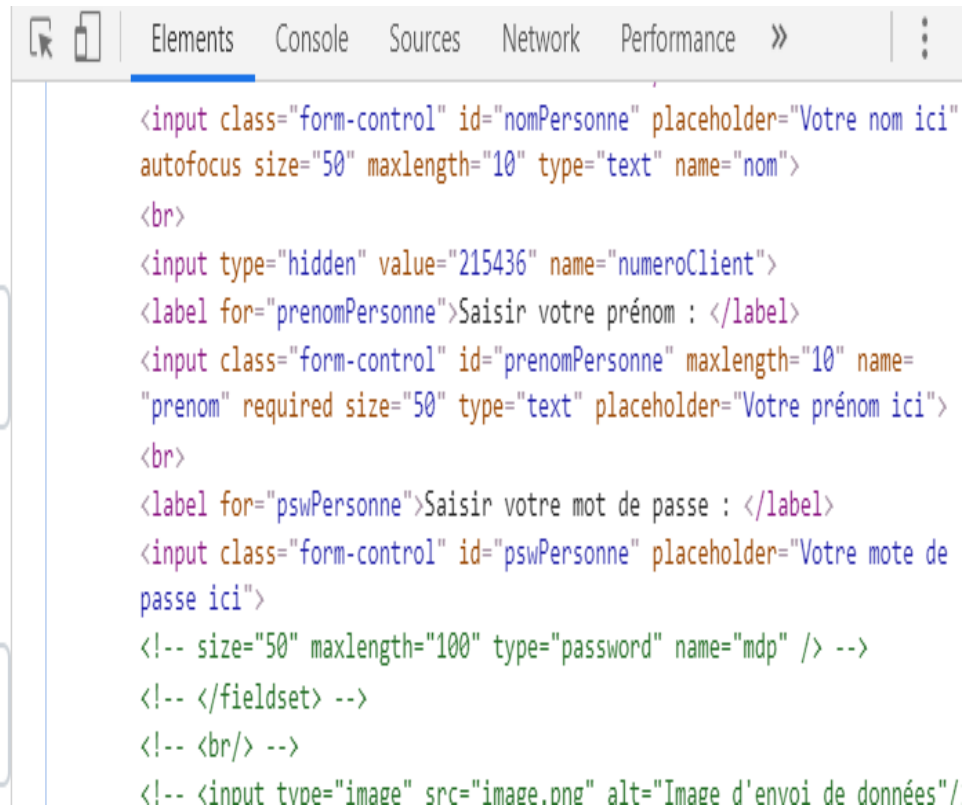
## Vos coordonnées

Saisir votre nom :

Votre nom ici

Saisir votre prénom :

Votre prénom ici



```
Elements Console Sources Network Performance »  
<input class="form-control" id="nomPersonne" placeholder="Votre nom ici"  
autofocus size="50" maxlength="10" type="text" name="nom">  
<br>  
<input type="hidden" value="215436" name="numeroClient">  
<label for="prenomPersonne">Saisir votre prénom : </label>  
<input class="form-control" id="prenomPersonne" maxlength="10" name=  
"prenom" required size="50" type="text" placeholder="Votre prénom ici">  
<br>  
<label for="pswPersonne">Saisir votre mot de passe : </label>  
<input class="form-control" id="pswPersonne" placeholder="Votre mote de  
passe ici">  
<!-- size="50" maxlength="100" type="password" name="mdp" /> -->  
<!-- </fieldset> -->  
<!-- <br/> -->  
<!-- <input type="image" src="image.png" alt="Image d'envoi de données"/>
```

# Encore du Html !

- Les formulaires de fichier :

```
<input type="file" name="selectedFile"/>
```

## Vos coordonnées

Saisir votre nom :

 Aucun fichier choisi

Saisir votre prénom :

# Html et le multimédias

- Html peut avoir différents types de contenu y compris les vidéos et les audios
- Avant Html5 on devait passer par le plugin **flash player** de Adobe afin de pouvoir regarder des vidéos sur les sites web
- Avec Html 5 il est possible de voir les vidéos et audios en utilisant les deux balises standards `<audio>` et `<video>`
- Ils existent différents types de formats audios : mp3, aac, ogg, wav...  
Aucun type n'est compatible sur tous les navigateurs mais les plus compatibles sont le mp3 et le ogg
- Le format vidéo est plus complexe, il regroupe le format audio (mp3, ogg, ..), le format vidéo (WebM, H.264, ..) et le type de leur conteneur (AVI, MP4, MKV, 3GPP, ..)
- Il est conseiller d'utiliser plusieurs format pour assurer la compatibilité avec les navigateurs

# Insertion d'éléments audio

- Pour insérer un élément audio sur une page web, il suffit d'utiliser la balise « audio » avec l'attribut « src » qui indiquera le chemin vers le fichier audio

```
<audio src="mon_audio.mp3" > </audio>
```

- Ceci ne suffit pas pour afficher quelque chose sur la page web. Il faut ajouter d'autres informations à la balise via l'attribut «controls» qui permet de mettre le lecteur audio ainsi que ses commandes
- Il existe d'autres attributs qui peuvent être utilisés : loop, autoplay, ..

# Assurer la lecture du fichier par les navigateurs

- Certains navigateurs ne sont pas capables de lire certains formats audio. Pour remédier à ce problème il faut proposer plusieurs formats et donc de fournir chaque chemin vers chaque fichier :

```
<audio controls >  
  <source src="audio.mp3"> </source>  
  <source src="audio.ogg"> </source>  
</audio>
```

# Insertion d'éléments vidéo

- Pour insérer des vidéos dans une page web il suffit d'utiliser la balise `<video>` avec l'attribut « src » de la même manière que la balise `<audio>`
- Il faut aussi ajouter l'attribut « controls » pour afficher le lecteur vidéo avec ses commandes
- L'attribut « poster » permet d'afficher une image si la vidéo n'est pas jouée
- Il existe également d'autres attributs qui permettent de donner plus d'informations et options sur la vidéo : preload, autoplay, loop

# Exemple d'insertion d'une vidéo

```
<video src="video.mp4" poster="image.png" controls> </video>
```





# Le fond d'une page web

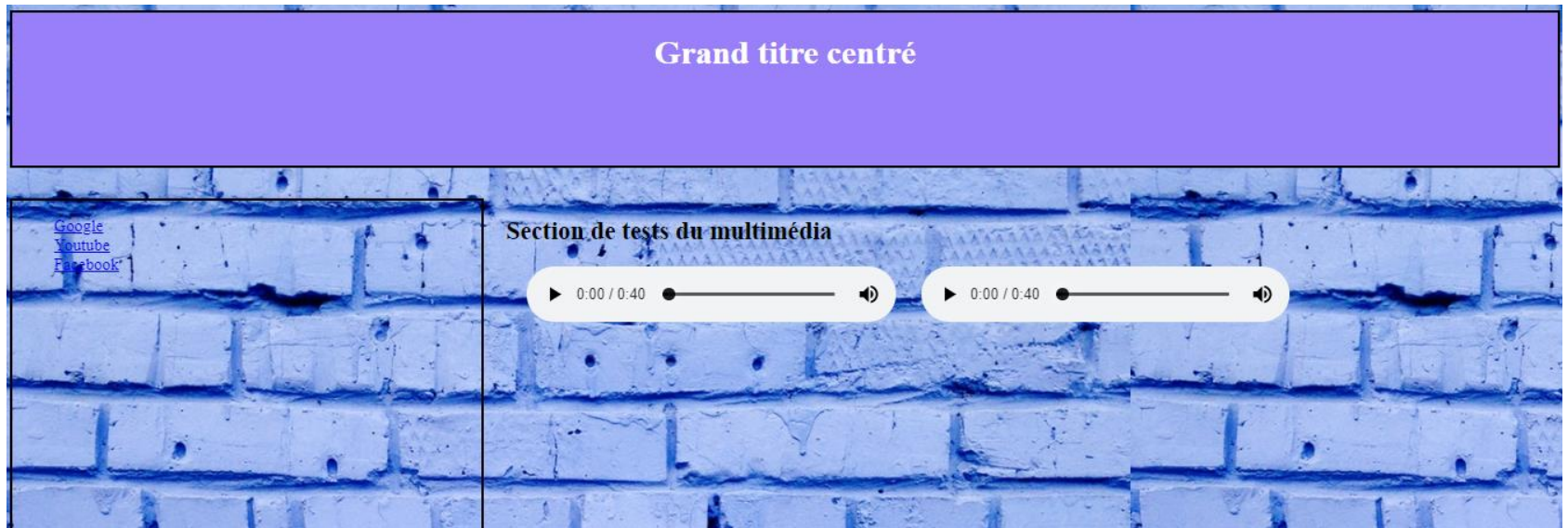
- Il est possible de changer la couleur de fond d'une page web ou d'un élément avec la propriété CSS « background-color »
- Pour mettre une image de fond de la page ou de l'élément il suffit d'utiliser la propriété CSS « background-image » de la manière suivante :

background-image : url("image.jpg");

- Il existe d'autres options CSS applicables sur les images de fond :
  - background-attachment : permet de fixer le fond grâce à la valeur « fixed » (la valeur « scroll » est utilisée par défaut)
  - background-repeat : permet de préciser l'option de répétition de l'image (no-repeat, repeat-x, repeat-y, repeat)
  - background-position : indique la position de l'image de fond en pixels par rapport à l'origine (à gauche en haut). Il est possible d'utiliser les valeurs « top, right, bottom, center et left »

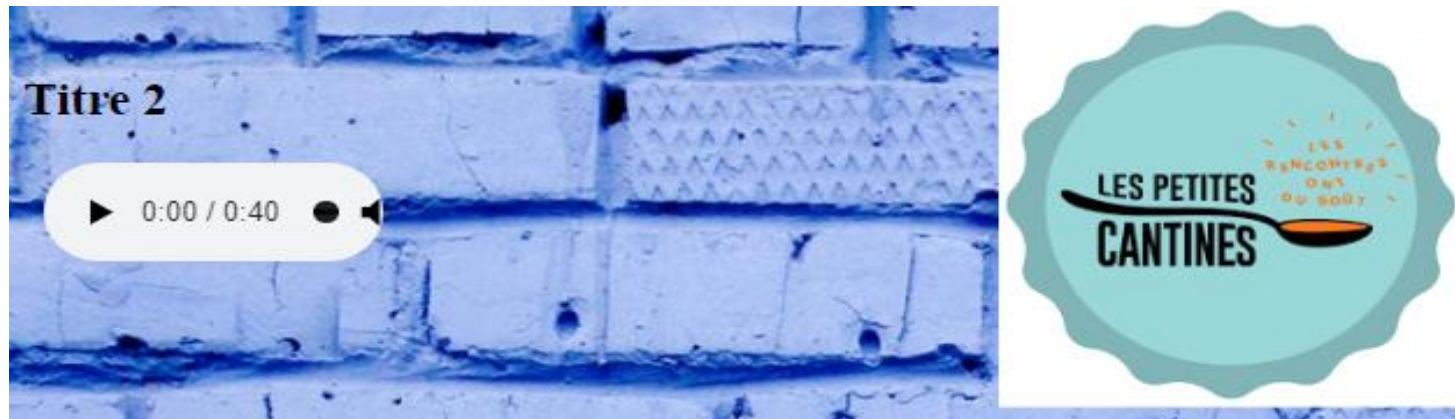
# Exemple d'insertion d'images de fond

```
body{  
    background-image : url("photo.jpg") ;  
    background-attachment : fixed;  
}
```



# Exemple d'insertion d'images de fond

```
body{  
    background : url("image.png") fixed no-repeat top right,  
                url("photo.jpg") fixed;  
}
```



# Gestion de la transparence

- Il est possible en CSS de modifier la transparence d'éléments d'une page web
- Ceci se fait de deux manières :
  - Avec la propriété CSS « opacity » qui prend une valeur entre 0 et 1
    - Si la valeur est égale à 0, l'élément sera complètement transparent
    - Si la valeur est égale à 1, l'élément sera complètement opaque (comportement par défaut)
  - Avec la règle RGBa (CSS 3) qui est la même utilisée pour la couleur avec un quatrième paramètre qui représente l'opacité
- Sur IE8, la propriété « opacity » est mal supportée. Pour résoudre ce problème il faut utiliser le filtre alpha de microsoft :

`filter: alpha(opacity=50);`

# Exemple d'utilisation

```
h2 {  
  background-color : black;  
  color: red;  
  opacity : 0.4;  
}
```



```
h2 {  
  background-color : rgba(0, 0, 0, 0.4);  
  color: rgba(255, 0, 0, 0.4);  
}
```



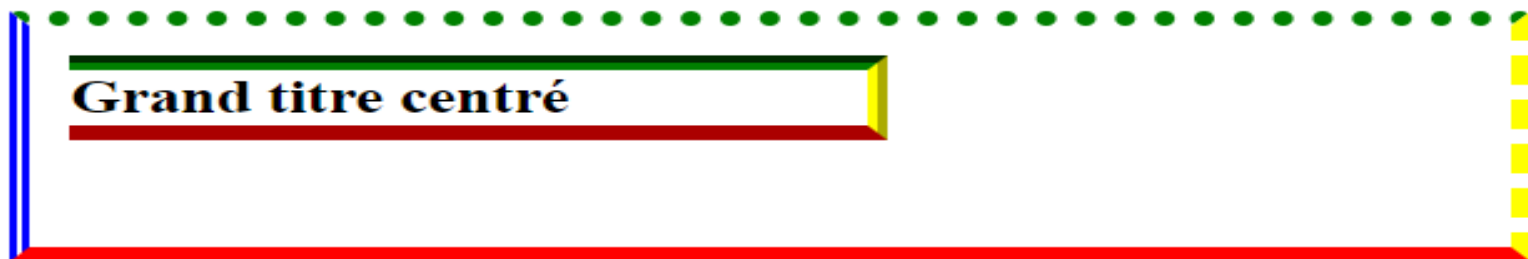
# Les bordures

- En Html, le bordures d'un élément sont ajoutées de la manière suivante :

```
<header>
  <h1>Grand titre centré</h1>
</header>
```

```
header {
  margin : 20px;
  border-left :10px double blue;
  border-top :10px dotted green;
  border-right :10px dashed yellow;
  border-bottom :10px inset red;
  height : 150px;
}

h1 {
  width : 400px;
  margin : 20px;
  border-left :10px none blue;
  border-top :10px groove green;
  border-right :10px ridge yellow;
  border-bottom :10px outset red;
}
```



# Les bordures arrondies

- Depuis CSS 3 il est possible de définir des bordures avec des coins arrondis grâce à la propriété « border-radius » qui définit l'importance de l'arrondi en pixel, en pourcentage ou en em

```
header {  
    margin : 20px;  
    border-left :10px double blue;  
    border-top :10px dotted green;  
    border-right :10px dashed yellow;  
    border-bottom :10px inset red;  
    height : 150px;  
    border-radius : 50px;  
}
```



# Les bordures arrondies

```
border-radius: 30px;
```

```
border-radius: 25% 10%;
```

```
border-radius: 10% 30% 50% 70%;
```

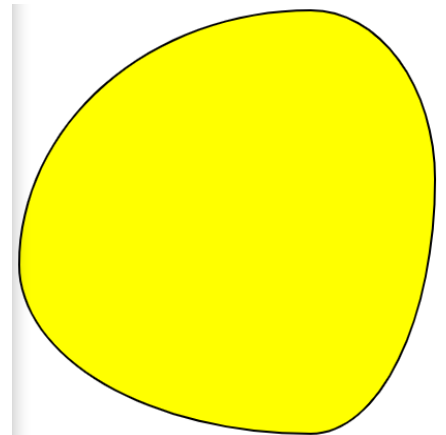
```
border-radius: 10% / 50%;
```

```
border-radius: 10px 100px / 120px;
```

```
border-radius: 50% 20% / 10% 40%;
```



```
.arrondi {  
  width : 400px;  
  height : 400px;  
  border : 2px solid black;  
  border-radius : 70% 30% 30% 70% / 60% 40% 60% 40%;  
  background-color : yellow;  
}
```





# Les ombres

- En CSS on peut mettre des ombres autour des éléments (blocs ou textes) avec les propriétés « box-shadow » et « text-shadow »
- La valeur est constituée de 4 parties obligatoires:
  - Le décalage par rapport à l'horizontale de l'ombre
  - Le décalage vertical de l'ombre
  - L'adoucissement du dégradé
  - La couleur de l'ombre

# Exemple d'ajout d'ombres

```
header {  
  margin : 20px;  
  border-left : 1px solid red;  
  box-shadow: 10px 20px 20px black;  
  height : 150px;  
  border-radius : 50px;  
}
```

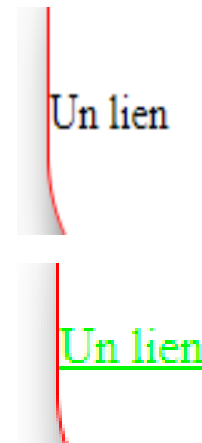
```
h1 {  
  width : 400px;  
  margin : 40px;  
  text-shadow: 5px 5px 5px blue;  
}
```

**Grand titre centré**

# Des styles dynamiques

- CSS gère aussi le style de manière dynamique lorsqu'une action est faite par l'utilisateur (survol, clic, focus, ..)
  - Dans le cas d'un survol (pointer avec la souris), utiliser le pseudo format « :hover »
  - Si on souhaite donner un style personnalisé à un lien déjà consulté, utiliser « :visited »

```
a {  
    text-decoration : none;  
    color : black;  
    font-style : normal;  
}  
  
a:hover {  
    text-decoration : underline;  
    color : #04f906;  
    font-style : normal;  
}
```



# Des styles dynamiques

- Dans le cas du clic sur un élément, utiliser le pseudo format « :active » affichera l'élément avec le style souhaité au moment du clic
- Lorsque un élément est sélectionné, mettre le pseudo format « :focus » (exemple vu avec les formulaires)

# Html : les caractères spéciaux

- Pour afficher les caractères spéciaux non accessibles par le clavier, Html liste un ensemble d' « entités » qui correspondent aux caractères spéciaux

Caractère	Code texte	commentaire
§	&sect;	
¶	&para;	
©	&copy;	
®	&reg;	
™	&trade;	
.....	....	....

# Les médias queries

- Pour assurer un affichage adapté à chaque type d'écran (ordinateur, tv, smartphone, ..), CSS 3 proposent des nouvelles règles qui peuvent être appliquées selon certaines conditions
- Deux manières d'écrire des médias queries :
  - Sur le document Html : charger un fichier CSS selon des conditions

```
<link rel="stylesheet" href="resolution_1024px.css"
      media="screen and (max-width : 1024px)" />
```

- Sur le document CSS : appliquer des règles sous certaines conditions

```
@media screen and (max-width : 1280px )
{
    body {
        background-color : grey;
    }
}
```

# Les médias queries

Résolution < 1280px



Résolution > 1280px



# Technologies liées à Html 5

- Javascript : langage de programmation permettant de dynamiser encore plus nos pages web en proposant des techniques au-delà des limites de Html 5 et CSS 3
- Canvas : en utilisant la balise `<canvas>` il est possible de dessiner des formes, ajouter des images et les manipuler, ..
- SVG : permet de faire des dessins et les agrandir à l'infini
- Drag & Drop : permet d'utiliser la fonctionnalité « glisser-déposer »
- File Api : souvent utilisée avec les formulaires (input de type file) et le Drag & Drop, elle permet l'accès aux informations d'un fichier stocké dans la machine de l'utilisateur



# Technologies liées à Html 5

- Géolocalisation : fournit des services de localisation de l'utilisateur
- Web Storage : un espace de stockage sur la machine de l'utilisateur
- Appcache : utilisation du cache navigateur pour recharger des pages
- Web Sockets : utilisée pour l'amélioration des échanges entre le navigateur et le serveur

# La compatibilité entre les navigateurs

- Les anciennes version de Internet Explorer (< IE 9) sont souvent un obstacle pour un développeur web. Il faut toujours vérifier et faire en sorte que votre site fonctionne sur ces versions qui sont très utilisées
- L'une des manières de gérer ces incompatibilités est d'utiliser les « commentaires traditionnels ». Ce sont des commentaires Html (<!-- -->) qui ne sont lus que par IE :

```
<!--[if IE] >  
    Si le navigateur est IE alors :  
<![ endif ] -->
```

# La compatibilité entre les navigateurs

- Il est possible d'utiliser des opérateurs de comparaisons (lt, lte, gt, gte) pour vérifier la version de IE

```
<!--[if lte IE8] >  
    Si la version de IE est inférieur ou égale à 8 alors :  
<![ endif ] -->
```

- Cibler des navigateurs autres que IE :

```
<!--[if ! IE]><!-->  
    Code Html vu que par tous les navigateurs sauf IE  
<!--<![endif]-->
```

# Exemples d'utilisation des commentaires conditionnels

- Charger un fichier de feuilles de style CSS si ce n'est pas IE8

```
<!--[if ! IE8]>
<link rel="stylesheet" href="style.css"/>
<![endif]-->
```

- Appliquer un style sur une classe d'éléments pour chaque version de IE

page.html

```
<!--[if IE6] > <body class ="ie6"> <![endif]-->
<!--[if IE7] > <body class ="ie7"> <![endif]-->
<!--[if IE8] > <body class ="ie8"> <![ endif ] -->
<!--[if !IE] > <!-- --><body class="other"><!-- <![ endif ] -->
```

Style.css

```
.ie6{/*Style IE6*/}
.ie7{/*Style IE7*/}
.ie8{/*Style IE8*/}
.other{/*Style autres navigateurs*/}
```

# L'accessibilité

- Permettre au public ciblé, quelque soit son matériel/logiciel, sa localisation et son handicap de profiter de son passage sur le site
- Il faut toujours faire en sorte que notre application web soit accessible par tous les internautes y compris ceux qui ont des handicaps visuels (daltoniens, malvoyants, ..)
- Assurer un chargement rapide de la page
- Vérifier la compatibilité des navigateurs

# L'accessibilité : les couleurs et les images

- Pour la gestion des couleurs de la page il faut toujours utiliser un contraste élevé entre les couleurs utilisées et l'arrière plan de la page
- Eviter de différencier les éléments et les actions uniquement avec la couleur
- Eviter de proposer des fonctionnalités selon les couleurs (cliquer sur le bouton rouge pour faire quelque chose)
- Utiliser l'attribut « alt » qui représente une alternative textuelle à l'image
- Mettre la valeur de « alt » à vide si l'image n'apporte pas une information qui devrait être lue par un lecteur d'écran (NVDA)

# L'accessibilité et le outline

- L'outline est généralement l'encadrement sur un élément d'une page web qui s'affiche lorsque cet élément est sélectionné (active, focus, ..)
- Il permet à l'utilisateur d'avoir un repère dans la page lorsqu'il fait une action
- Supprimer ce comportement sur l'élément supprimera le mécanisme de l'accessibilité sur cet élément

# L'accessibilité et le texte

- Il est possible de changer la dimension du texte sur un navigateur (jusqu'à zoom x 4)
- Cette fonctionnalité est utile pour les personnes qui souffrent de la presbytie (difficultés à voir net)
- Il faut toujours penser que lorsque le zoom est fait, il ne faut surtout pas perdre l'information même au détriment du design
- Le fait d'avoir un scroll lors du zoom n'est pas un problème par rapport à l'accessibilité (l'information est toujours là)
- Tronquer/cacher des mots suite à un zoom est une perte d'information !



# HTML 5 et ARIA

- ARIA permet au développeur de décrire plus précisément ses widgets (programmes dans une page web) et fournir une sémantique essentielle au bon fonctionnement des lecteurs d'écrans
- ARIA distingue trois types d'attributs:
  - Rôles : utilisé avec l'attribut « role », il décrit à quoi sert l'élément (navigation, bouton, ..)
  - États : utilisé avec l'attribut « aria-[etat], il décrit l'état de l'élément (activé, sélectionné, ..)
  - Propriétés : utilisé avec l'attribut « aria-[propriété], il décrit les caractéristiques de l'élément (label, description, ..)



# Bootstrap



# Utilisation de bootstrap

- Bootstrap est un framework CSS permettant de faire des pages responsives plus facilement
- Il se base sur une représentation de la page web en grille d'éléments sur 12 colonnes
- L'une des manières les plus simples pour l'utiliser est de récupérer le lien (CDN) vers le CSS Bootstrap minifié et de le mettre dans le head du document Html (accès à internet requis)

## BootstrapCDN

Skip the download with [BootstrapCDN](#) to deliver cached version of Bootstrap's compiled CSS and JS to your project.

```
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">  
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-
```

# Utilisation de bootstrap

- Définir notre propre feuille de style CSS et de l'ajouter **après** le link vers bootstrap
- Utiliser les différentes classes utilisées par Bootstrap (important) :
  - container-fluid : une classe permettant à un élément de prendre toute la largeur de la page (équivalent à width : 100%)
  - container : une classe permettant de prendre une grande partie de la largeur de l'écran en laissant des marges à gauche et à droite

Exemple d'utilisation de la classe « container-fluid »

```
<body>
  <div class="container-fluid div-rouge">
    <div>
  </body>
```

```
.div-rouge {
  height :400px;
  background-color : grey;
}
```

# Utilisation de bootstrap

- Pour chaque ligne ajoutée dans la grille, utiliser une `<div>` avec la classe bootstrap « row »
- Pour les colonnes, Bootstrap considère que l'écran contient 12 colonnes. Exemple d'utilisation de la classe « col-md-12 »:

```
<body>
  <div class="container-fluid div-rouge">
    <div class="container">
      <div class="row">
        <section class="col-md-12">
          <h1>Titre principal</h1>
          <p>
            Lorem ipsum dolor sit amet, consectetur
          </p>
        </section>
      </div>
    </div>
  </div>
</body>
```

## Titre principal

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

- Exemple d'utilisation de 3 colonnes

« col-md-4 » :

```
<body>
  <div class="container-fluid div-rouge">
    <div class="container">
      <div class="row">
        <section class="col-md-4">
          <h1>Titre principal 1</h1>
          <p>
            Lorem ipsum dolor sit amet
          </p>
        </section>
        <section class="col-md-4">
          <h1>Titre principal 2</h1>
          <p>
            Lorem ipsum dolor sit amet
          </p>
        </section>
        <section class="col-md-4">
          <h1>Titre principal 3</h1>
          <p>
            Lorem ipsum dolor sit amet
          </p>
        </section>
      </div>
    </div>
  </div>
</body>
```

## Titre principal 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Titre principal 2

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Titre principal 3

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Utilisation de bootstrap

- Bootstrap propose l'utilisation de 5 types de colonnes pour 5 différents types d'écrans :
  - col-xs-[nbr-col] : l'élément prendra « nbr-col » de colonnes pour un écran très petit (extra small)
  - col-sm-[nbr-col]: l'élément prendra « nbr-col » de colonnes pour un écran petit ( small)
  - col-md-[nbr-col]: l'élément prendra « nbr-col » de colonnes pour un écran moyen (middle)
  - col-lg-[nbr-col]: l'élément prendra « nbr-col » de colonnes pour un grand écran (large)
  - col-xl-[nbr-col]: l'élément prendra « nbr-col » de colonnes pour un très grand écran (extra large)
- Il est possible (recommandé) d'utiliser plusieurs classes pour les éléments afin d'assurer un rendu responsive sur les différents types d'écrans