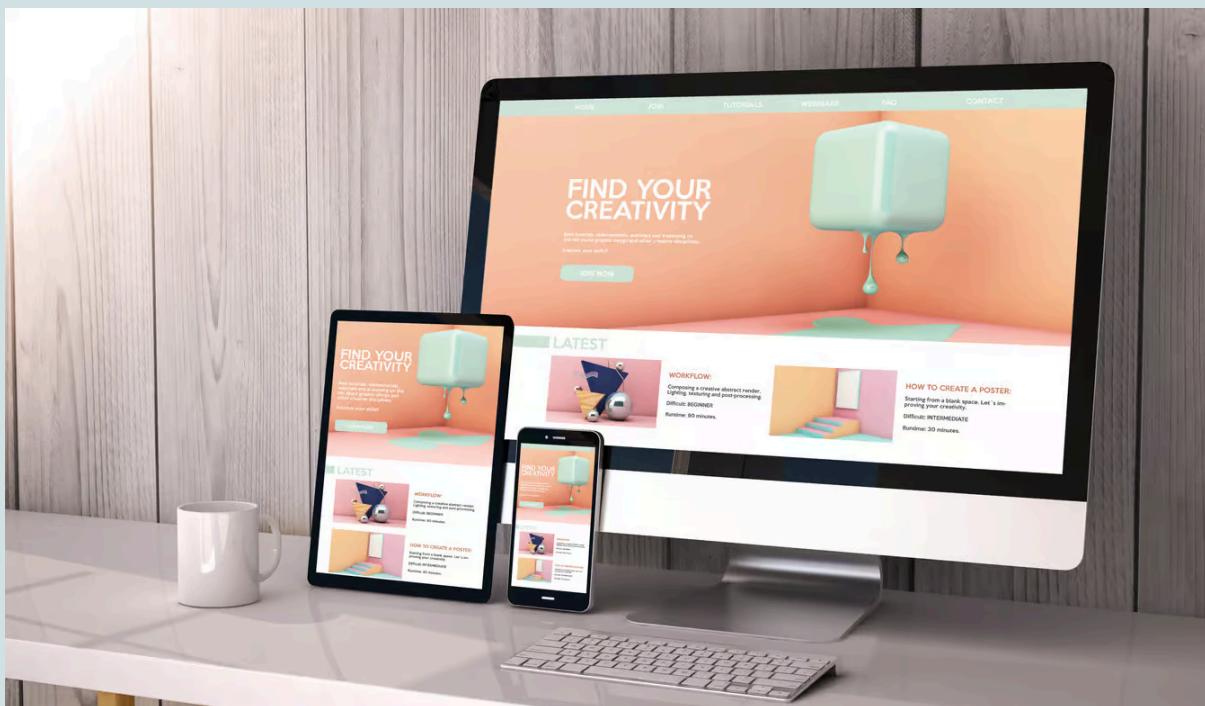


Responsive Design



[Exercice 1](#)

[Exercice 2](#)

[Exercice 3](#)

[Exercice 4](#)

[Exercice 5](#)

[Exercice 7](#)

[Exercice 8](#)

La problématique : Rendre le site accessible et confortable peu importe le format de l'écran qui sera choisi par l'utilisateur.

Exercice 1

Nous devons donner au body la largeur 100% mais aussi la valeur 67 et % au marge de article et aside.

```
article {  
    width: 60%;  
    margin-right: 67%;  
}  
  
aside {  
    width: 30%;  
    margin-right: 33%;  
}
```

Exercice 2

Nous allons devoir donner un limite de maximale et minimale à article et aside.
Article ne dépassera pas les 500px mais sera d'au minimum 200px et il en est de même pour l'aide qui ne dépassera pas les 250px mais sera au minimum de 150px.

```
article {  
    width: 60%;  
    margin-right: 67%;  
    max-width: 500px;  
    min-width: 200px;  
}  
  
aside {  
    width: 30%;  
    margin-right: 33%;  
    max-width: 250px;  
    min-width: 150px;  
}
```

Exercice 3

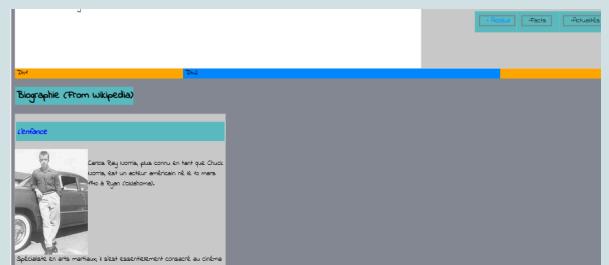
Les div pour un body de 750px se partage toute la longueur mais dès que la valeur du body dépasse les 800px alors la div 2 reste à 50% de l'écran à l'inverse de la div 1. (body de 1500px)

```
</header>
<div style="display:flex" class="div1">
  <div style="width:50%;max-width:400px;">
    |   Div1
  </div>
  <div style="width:50%;" class="div2">
    |   Div2
  </div>
</div>
```

body de 750px



body de 1500px



Exercice 4

Le flex-grow va être un élément crucial pour régler le problème de la div 2 et grâce à cela même avec un body au delà de 800px les div remplissent toute la largeur.

```
</header>
<div style="display:flex" class="div1">
  <div style="width:50%;max-width:400px;">
    |   Div1
  </div>
  <div style="width:50%;flex-grow: 1;" class="div2">
    |   Div2
  </div>
</div>
```

body 900



Nous devons augmenter la largeur de article et aside et de plus leur ajouter un flex-shrink et un flex-grow de tel sorte à ce que à ce que l'espace gagné par article lorsque que l'écran est trop soit 3 fois plus grand que celui gagné par aside.

```
article {  
    width: 300px;  
    margin-right: 67%;  
    max-width: 500px;  
    min-width: 200px;  
}  
  
aside {  
    width: 200px;  
    margin-right: 33%;  
    max-width: 250px;  
    min-width: 150px;  
}
```

```
article {  
    width: 300px;  
    flex-shrink: 1;  
    flex-grow: 3;  
}  
  
aside {  
    width: 200px;  
    flex-shrink: 1;  
    flex-grow: 1;  
}
```

Dans le cas opposé d'un écran trop petit il faut que article diminue de deux tiers de la largeur à supprimer et aside du reste. Pour cela il faudra donc utiliser flex-shrink.

```

article {
    width: 300px;
    flex-shrink: 2;
    flex-grow: 3;
}

```

Outil développeur :



La valeur du body est bien de 980px

```

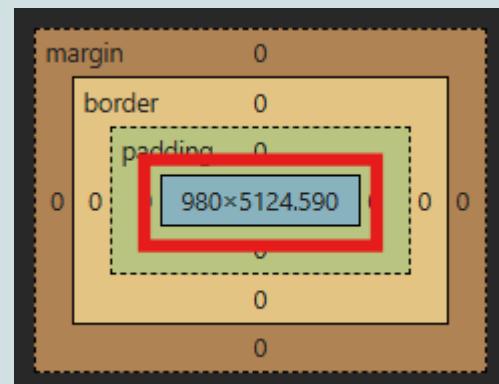
body{
    font-family: 'Indie Flower';
    background-color: #838892;
    width: 980px;
    line-height: 150%;
    text-indent: 5px;
    margin: 0 auto;
}

```

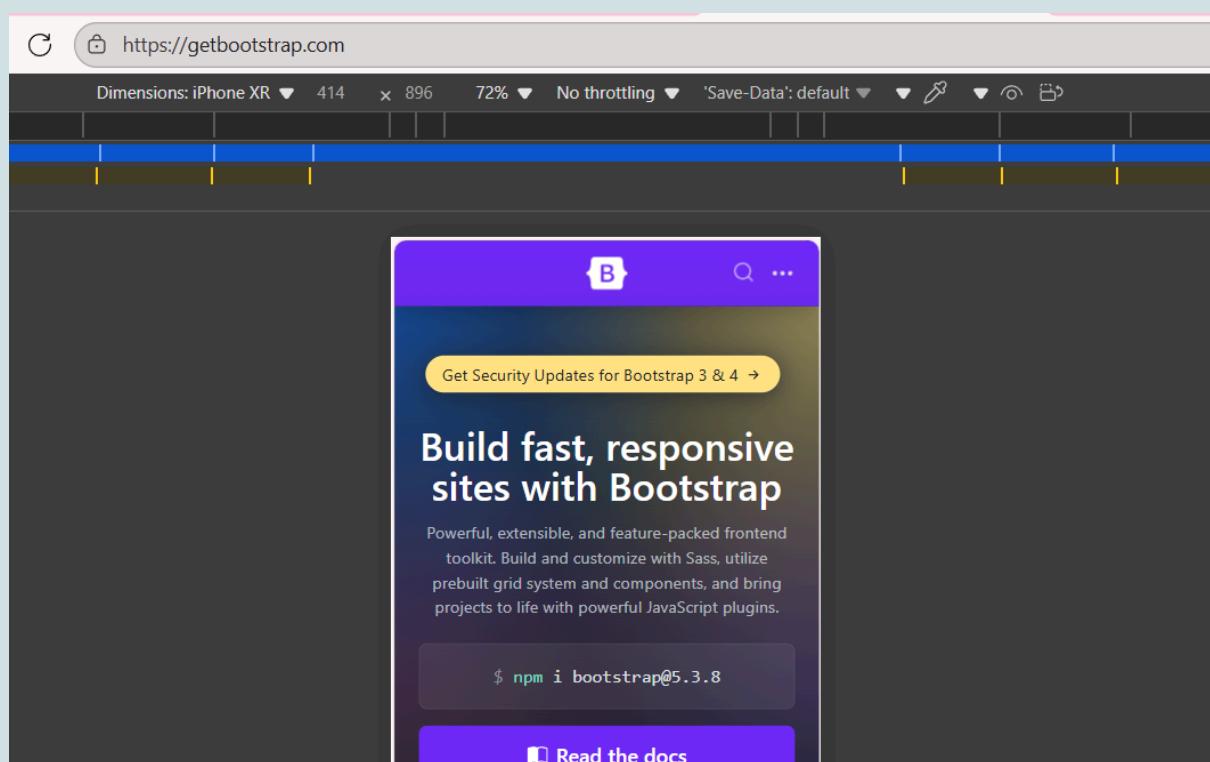


Exercice 6

Le body garde l'ancienne valeur de 980



Exercice 7



Au-dessus de 768px : le menu est visible horizontalement.

En dessous de 768px : le menu devient replié.

Exercice 8

En dessous de 768px la table de comparaison disparaît. Et en dessous de 480px article et aside s'affiche en ligne et non en colonne.

```
@media (max-width: 768px) {  
    table.comparaison {  
        display: none;  
    }  
}
```

```
@media (max-width: 480px) {  
    aside, article {  
        display: block;  
        width: 100%;  
    }  
}
```

Il va falloir rajouter le menu burger dans l'HTML

```
<div class="burger">  
    <div class="burger">  
          
        <div id="menu2">  
            <div><a href=".//index.html">Accueil</a></div>  
            <div><a href=".//facts.html">Facts</a></div>  
            <div><a href=".//news.html">Actualités</a></div>  
            <div><a href=".//contact.html">Contact</a></div>  
        </div></div>
```

Ensuite donner à la position du menu burger, la valeur fixed qui permet de positionner un élément par rapport à la fenêtre du navigateur.

```
position: fixed;
```

Z-index lui permettra lui de positionner le menu par dessus les autres éléments

```
z-index: 1001;
```

Conclusion : Ce TP m'a permis d'adapter mes méthodes de travail concernant le CSS pour qu'elles soient plus adaptées aux différentes interfaces utilisateurs.