Cours sur HTML et CSS

Initiation

HTML est le premier langage à apprendre pour réaliser des pages web. Il permet de structurer votre contenu (titres, paragraphes, images, liens...). CSS permet d'assurer le graphisme de ces éléments (disposition, couleur, taille...).

Faire Codecademy HTML et CSS

Un bon cours est disponible sur Open Classroom : faire parties 1/2/3.

Documentation

Balises HTML sur Mozilla et propriétés CSS sur OpenClassroom.

Vocabulaire

- balise ouvrante et fermante (ex: `<h1>` et `</h1>`)
- attribut (ex: `href` de `<a href="...">`)
- sélecteur (ex: `header h1`)
- propriété (ex: `display`)
- valeur (ex: `inline-block`)

Sélecteurs CSS

Voici un résumé des correspondances entre sélecteurs CSS et balises HTML :

```
p { color: red; }
```

```
<p>rouge</p>
```

```
p.titi { color: red; }
```

```
<p class="titi">rouge</p>
```

```
.titi { color: red; }
```

```
<nimportequoi class="titi">rouge</nimportequoi>
```

```
p#titi { color: red; }
```

```
<p id="titi">rouge</p>

#titi { color: red; }

<nimportequoi id="titi">rouge</nimportequoi>

p a { color: red; }

<p><a href="...">rouge</a></p>
<a href="..."><p>pas rouge</p></a>
<a href="...">pas rouge</a>
<p>pas rouge</i>

p, a { color: red; }

<p>rouge</p>
<a href="...">rouge</a>

ul#titi p.tata { color: red; }

<ul id="titi">
  <li><p class="tata">rouge</p></li>
  <li><p>pas rouge</p></li>
</ul>
<p class="tata">pas rouge</p>
```

Positionnement des éléments en CSS

Voir positionnement des éléments en css.

Exemple-résumé

Cet exemple utilise massivement les nouvelles balises HTML5 (header, nav, main, etc.). Elles sont tellement nouvelles que certains éditeurs de code n'ont pas la coloration syntaxique pour ces balises ! Aussi, si vous oubliez les fameux `display: inline-block`, certains navigateurs ne vont plus du tout fonctionner pour votre site. Cela étant, les `inline-block` fonctionnent tellement bien que nous n'allons quasiment plus qu'utiliser que ça pour gérer le positionnement des éléments en CSS. Adieux `float` !

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Titre de la page</title>
    <link rel="stylesheet" type="text/css" href="style.css">
  </head>
  <body>
    <div class="container">
      <header>

        <!-- En-tête du site -->
        <h1>Titre du site</h1>

      </header><nav>

        <!-- Zone de navigation -->
        <ul><!-- Liste d'éléments -->
          <li><a href="lien1.html">Lien 1</a></li>
          <li><a href="lien2.html">Lien 2</a></li>
```

```
          ...
        </ul>

      </nav><main>

        <!-- Contenu principal, texte de la page -->
        <h2>Titre de la page</h2>
        <p>Texte de la page</p>
        <p>Suite du texte de la page</p>

      </main><aside>

        <!-- Contenu annexe (pas souvent utilisé) -->

      </aside><footer>

        <!-- Pied de page -->

      </footer>
    </div>
    <script src="script.js"></script>
  </body>
</html>
```

```
* {
  margin: 0;
  padding: 0;
}

.container, header, nav, main, aside, footer {
  display: inline-block;
  vertical-align: top;
  box-sizing: border-box;
}

.container {
  width: 80%;
  margin-left: 10%;
  margin-right: 10%;
}

header {
  width: 100%;
}

nav {
  width: 20%;
}

main {
  width: 60%;
}

aside {
  width: 20%;
}

footer {
  width: 100%;
}
```

La trinité

Cette partie du code est **très** importante :

```
.container, header, nav, main, aside, footer {
  display: inline-block;
  vertical-align: top;
  box-sizing: border-box;
}
```

Nous avons besoin de display: inline-block dès lors que nous avons besoin de spécifier une largeur spécifique d'un block avec width. N'oubliez pas, si nous avons vu height en cours pour bien visualiser les blocks lors de nos tests, ils ne faut pas l'utiliser en vrai ! Tout au plus, éventuellement, min-height.

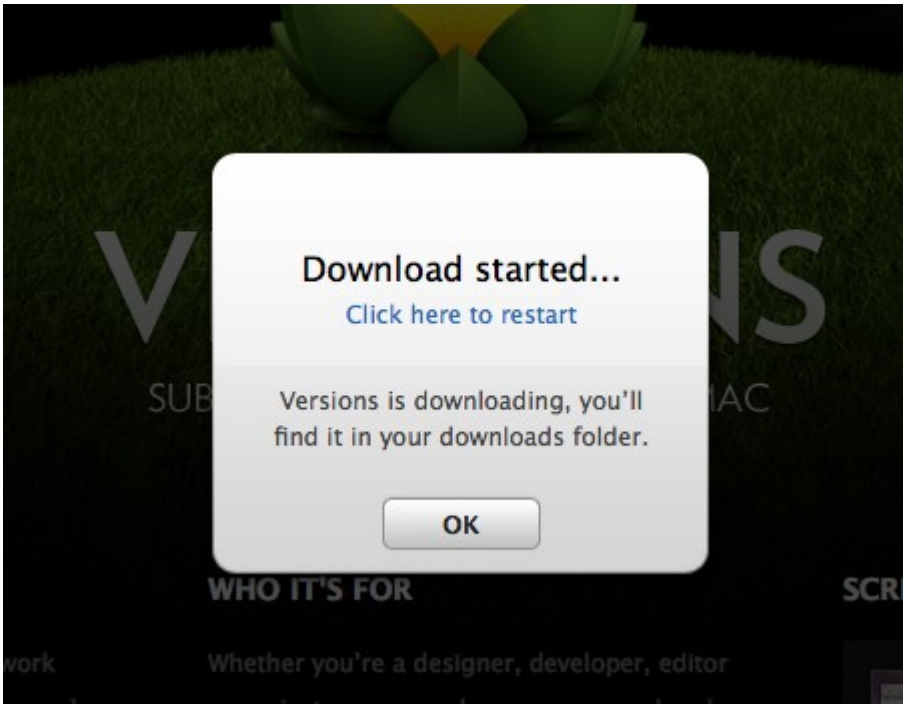
vertical-align: top vient corriger un bug d'alignement quand deux blocks (nav et main par exemple) sont cote-à-cote mais ne font pas la même taille.

box-sizing permet de mettre padding et border à nos blocks sans avoir à faire plein de calculs. Mais n'oubliez pas de tout de même soustraire vos margin de vos width.

N'oubliez jamais de coller vos fermetures et ouvertures, par exemple : …</nav><main>…

Exercice : Modal Box

Réalisez la plus belle modal box (sous-fenêtre) possible. Voici un exemple de modal :



Une structure et le bout de JS nécessaire ont été préparés ici : https://jsfiddle.net/m8x03wku/
Début de solution : https://jsfiddle.net/us83n1gg/

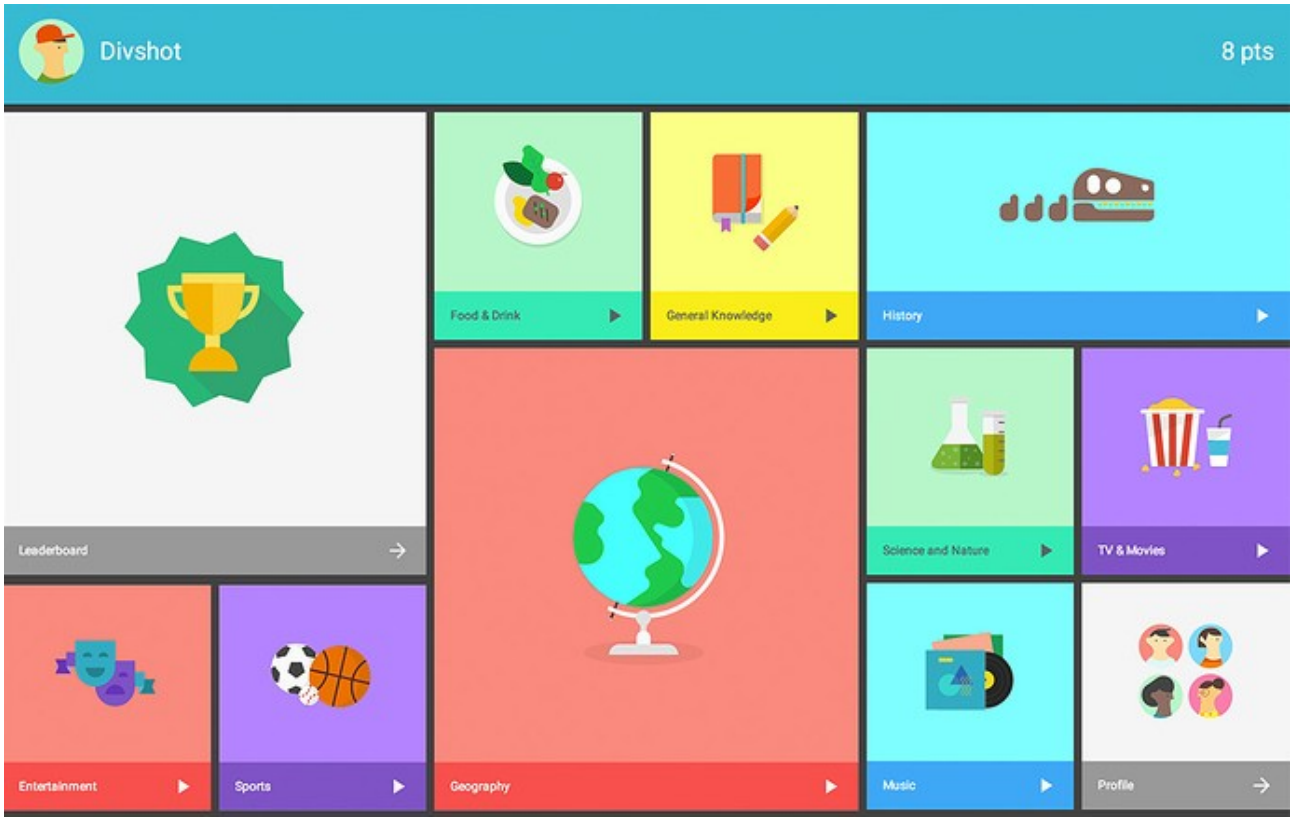
Exercice : des divs partout

Reproduire en HTML et CSS cette image. Vous pouvez utiliser Bootstrap, mais ce n'est pas obligatoire.

Le site doit s'afficher correctement sur les écrans de petite taille (reponsive design).

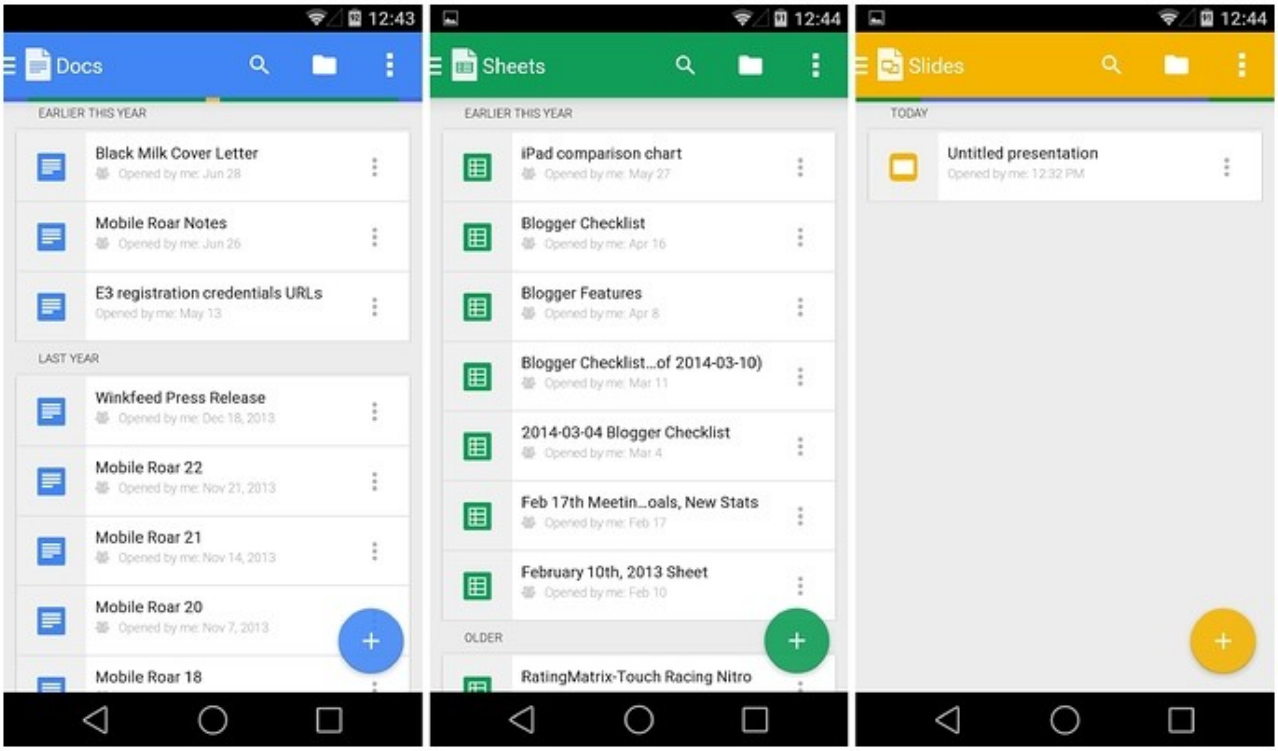
Pour les images, vous pouvez utiliser ce service : https://placehold.it/

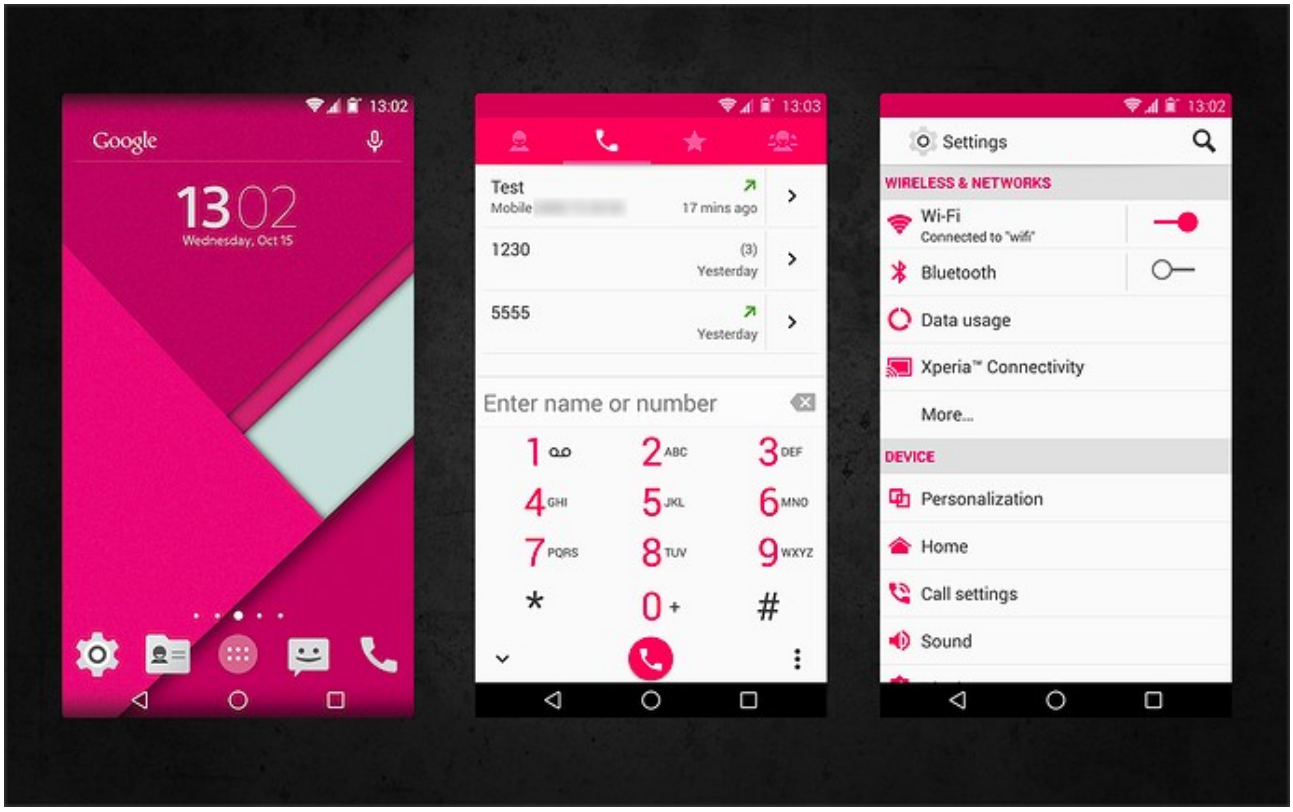
Ce n'est pas grave si les couleurs ne sont pas strictement identiques à celles-ci.



Exercices : écrans mobiles

Reproduire un ou plusieurs de ces écrans, sans les barres sur fond noir en haut et en bas :





La suite

Une fois le HTML assimilé, informez-vous sur ce qu'est le SEO.

Pour aller plus loin dans le CSS, Twitter Bootstrap