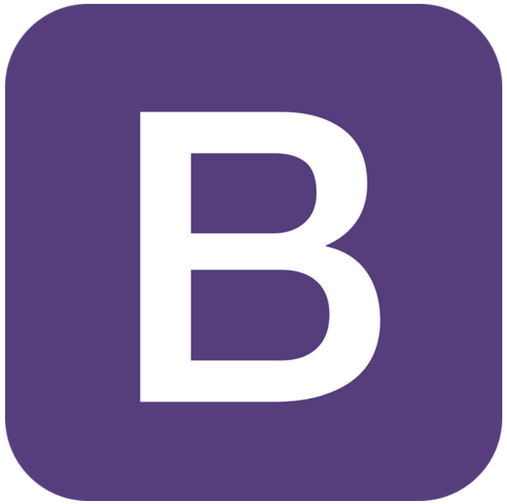
TALIS BUSINESS SCHOOL 2019-2020

BOOTSTRAP



Christopher Legrand - www.christopherlegrand.fr

Ce support est tiré du cours :
https://www.pierre-giraud.com/bootstrap-apprendre-cours/

Qu'est-ce que Bootstrap

Bootstrap a été crée par les développeurs de Twitter, et est devenu en peu de temps le framework CSS de référence !

Bootstrap est un framework, c'est à dire un ensemble de composants structurés qui sert à créer les bases et à organiser le code informatique pour facilité le travail des développeurs, gagner en productivité et simplifier la maintenance.

Pour aller encore un peu plus loin, on qualifie Bootstrap de framework CSS car il est tout simplement spécialisé dans le CSS.

Il va nous permettre de mettre en forme nos pages web : organisation, animation, responsive etc.. avec peu de lignes de code ajoutées.

Installation de Boostrap

L'installation de Bootstrap est très simple :

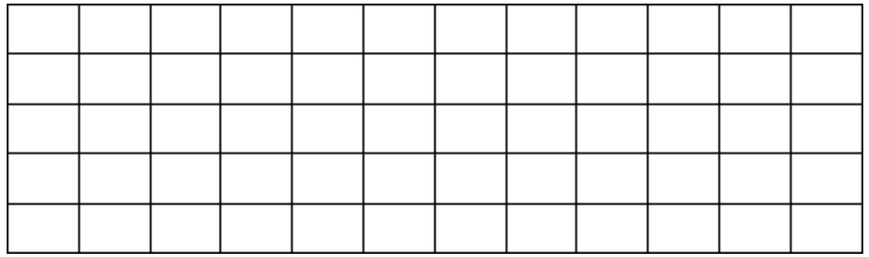
1. https://getbootstrap.com/
2. Télécharger Bootstrap version Compiled CSS and JS pour améliorer la vitesse de chargement du site
3. Déposer les fichiers dans le Workspace
4. Déclarer au minimum le fichier bootstrap.min.css (ou bootstrap.css) avec la balise <link>
5. Si vous souhaitez utilisez des composants JS, il faudra installer jQuery
6. Déclarer :
    a. jquery.js
    b. bootstrap.min.js
7. C'est parti !

Il existe d'autres façons plus simple d'installer Boostrap comme le CDN, mais dans une optique d'optimisation de site, je vous recommande d'installer directement Bootstrap dans votre projet.

La grille de Bootstrap

Bootstrap est un outil vraiment génial qui permet d'aménager nos pages à notre convenance.
Comment fait-il ? Et bien il intègre un système de grille pour obtenir le résultat souhaité.

Une grille est tout simplement un découpage en cellules :



La grille de Boostrap est composée de 12 colonne comme dans l'illustration précédente.

La grille de Bootstrap

Bootstrap utilise un système de « grille » à 12 colonnes de base reposant sur le modèle des boites flexibles (flexbox) comme système de disposition principal.

Le découpage en colonnes est tout simplement une division en pourcentage de la largeur de la fenêtre de visualisation

Le flexbox est ici utilisé par Bootstrap pour créer une sorte de grille mais nous n’allons pouvoir manipuler que les propriétés liées au flexbox (dépendant de display : flex) et non pas celles liées aux grilles (dépendant de display : grid).

Pour utiliser ce système de grilles Bootstrap, nous allons avant tout devoir définir un élément conteneur dans notre page HTML auquel on passera un attribut class="container" ou class="container-fluid".

La classe .container

La classe .container va permettre de définir un conteneur adaptable ou « responsive » de taille fixe, ce qui signifie que notre conteneur aura toujours la même taille pour un breakpoint donné et changera de taille à chaque breakpoint.

La largeur d'un conteneur défini avec .container va être la suivante entre chaque breakpoint :

La largeur par défaut d'un conteneur défini avec .container est égale à 100% de l'espace disponible.
Ensuite, pour les fenêtres d'au moins 1200px de large, le conteneur occupera une place de 1140px maximum.
Pour les fenêtres de largeur comprise entre 992px et 1200px, le conteneur occupera une place de 960px maximum etc..

```css
.container {
  width: 100%;
  padding-right: 15px;
  padding-left: 15px;
  margin-right: auto;
  margin-left: auto;
}

@media (min-width: 576px) {
  .container {
    max-width: 540px;
  }
}

@media (min-width: 768px) {
  .container {
    max-width: 720px;
  }
}

@media (min-width: 992px) {
  .container {
    max-width: 960px;
  }
}

@media (min-width: 1200px) {
  .container {
    max-width: 1140px;
  }
}
```

La classe .container-fluid

La classe .container-fluid permet de définir un conteneur de taille fluide, c'est-à-dire un conteneur dont la taille va changer en même temps que celle de la fenêtre de vos visiteurs.

Un conteneur défini avec .container-fluid occupera toujours 100% de la largeur disponible.

```
<div class="container bg-info mb-3">
    Conteneur responsive de taille fixe (change de taille à chaque breakpoint)
</div>

<div class="container-fluid bg-success">
    Conteneur responsive fluide (occupe toujours 100% de l'espace disponible)
</div>
```

Allons plus loin !

Lorsqu'on utilise Bootstrap, il est important de toujours définir un ou plusieurs conteneurs avec les 2 classes vues précédemment, dans lequel nous viendrons simplement ajouter notre contenu HTML grâce au système de grille Bootstrap.

Pour rappel, Bootstrap permet de créer des grilles de 12 colonnes en utilisant les propriétés du flexbox.

Comme vous le savez, le flexbox utilise un système d’axe principal et d’axe secondaire.

L’axe principal défini par Bootstrap est l’axe horizontal et les lignes vont ainsi servir de conteneurs aux colonnes.

La classe .row

La deuxième classe à connaître est .row, qui représente une rangée.

Dans le système de grille de Bootstrap, les lignes doivent absolument être placées dans des conteneurs et ne sont utilisées que dans le but de servir elles-mêmes de conteneurs pour les colonnes de la grille.

Il faut ensuite définir le nombre de colonnes pour chaque élément en sachant qu'il y en a au maximum 12.

Pour définir le nombre de colonnes utilisées pour chaque élément, on utilisera la classe .col-

La classe .col-

La troisième classe à connaître est .col-, qui représente une colonne.

Une fois la .row faite, Il faut définir le nombre de colonnes pour chaque élément en sachant qu'il y en a au maximum 12.
Pour définir le nombre de colonnes utilisées pour chaque élément, on dispose de quatre batteries de 12 classes :

- col-xs-1 ..... col-xs-12
- col-sm-1 ..... col-sm-12
- col-md-1 ..... col-md-12
- col-lg-1..... col-lg-12

Bootstrap considère 4 sortes de médias : les petits, genre smartphones (moins de 768 pixels), les moyens, genre tablettes (moins de 992 pixels), les écrans moyens (moins de 1200 pixels) et enfin les grands écrans (plus de 1200 pixels). Vous trouverez à la figure suivante un tableau pour illustrer les différences de réaction selon la catégorie.

Le nom des classes est intuitif :xs pour x-small, sm pour small, md pour medium et lg pour large.

Depuis Bootstrap 4 il y a désormais 5 tailles de grille possible ; extra-small, small, medium, large et extra large xl .

Un peu de pratique ?

Nous connaissons maintenant les .container, les .row et les .col. et donc, le fonctionnement principal de Boostrap !
Mettons un peu en pratique :

```
<div class="container-fluid">
  <div class="row">
      <div class="col-lg-3 bg-success">Colonne de 3/12</div>
      <div class="col-lg-5 bg-secondary">Colonne de 5/12</div>
      <div class="col-lg-4 bg-danger">Colonne de 4/12</div>
  </div>
</div>

<div class="container mt-3">
  <div class="row">
      <div class="col-lg-3 bg-success">Colonne de 3/12</div>
      <div class="col-lg-5 bg-secondary">Colonne de 5/12</div>
      <div class="col-lg-4 bg-danger">Colonne de 4/12</div>
  </div>
</div>
```

Le responsive

L’un des intérêts majeurs d’utiliser Bootstrap 4 est que cette version du framework a été développée en pensant à l’affichage sur mobile en premier et donc possède de nombreux outils permettant d’adapter nos designs en fonction de la taille de l’écran (ou plus exactement de la fenêtre) de nos visiteurs.

Avec les grilles, on va notamment pouvoir spécifier qu’une ligne doit posséder tel nombre de colonnes personnalisées qui vont chacune occuper tel nombre de colonnes de base pour une taille de fenêtre donnée puis que ces colonnes personnalisées doivent être réorganisées et occuper plutôt la place de tel autre nombre de colonnes de base pour cette colonne, telle autre nombre pour telle autre, etc. pour une autre taille de fenêtre.

Ainsi, les colonnes personnalisées d’une ligne vont pouvoir occuper un nombre différent de colonnes de base selon la taille de la fenêtre de chaque visiteur.Cette fonctionnalité va être très utile pour proposer un affichage optimisé pour différentes tailles de fenêtre. Par exemple, on pourra vouloir qu’un élément occupe une largeur égale à 3 colonnes de base pour un grand écran puis à 6 colonnes pour une taille d’écran 2 fois plus petite.

Pour faire cela, on va pouvoir utiliser les classes .col-sm, col-s, .col-md, .col-lg et .col-xl.

Le responsive

Ces classes sont liées aux breakpoints définis par Bootstrap. L'idée ici est très simple : selon la taille de la fenêtre, les styles d'une classe vont être appliqués prioritairement par rapport à ceux des autres. Ces classes vont s'appliquer pour les tailles de fenêtre suivantes :

	Extra small (< 576px)	**Small (>= 576px)**	**Medium (>= 768px)**	**Large (>= 992px)**	**Extra large (>= 1200px)**
Largeur maximale du conteneur	Aucune (auto)	540px	720px	960px	1140px
Nombre de colonnes	12				
Largeur de gouttière	30px (15px à gauche + 15px à droite)				

Le responsive

Comme Bootstrap 4 a été construit sur le principe du « mobile first », c'est la classe par défaut qui va s'appliquer pour toutes les tailles de conteneur si nous ne précisons pas de règle plus précise.

Par ailleurs, vous pouvez noter qu'il n'est pas strictement obligatoire d'utiliser une classe .col avec les éléments de ligne tant qu'au moins une autre classe .col-sm, .col-s .col-md, .col-lg ou .col-xl a été définie.

Dans ce cas-là, les éléments occuperont tout l'espace disponible dans la ligne pour les tailles de fenêtres où aucune instruction n'a été donnée. Regardez plutôt l'exemple suivant :

```
<div class="container">
    <div class="row">
        <div class="col-md bg-info">1er élément</div>
        <div class="col-md-6 bg-warning">2è élément</div>
        <div class="col-md bg-success">3è élément</div>
    </div>
</div>
```

L’alignement vertical des colonnes

Par défaut, les colonnes vont occuper toute la hauteur d’une ligne. On va cependant pouvoir demander aux colonnes de n’occuper que la place nécessaire à leur contenu et de s’aligner soit au début, soit au milieu, soit en fin de ligne selon l’axe vertical.

Cela va pouvoir être intéressant dans le cas où nos différentes colonnes possèdent des contenus qui utilisent des espaces en hauteur différents ou dans le cas où une hauteur a été explicitement définie pour la ligne.

Pour aligner toutes les colonnes en même temps par rapport à une ligne, nous allons appliquer les classes .align-items-* à nos lignes. Nous pouvons choisir parmi trois classes qui représentent trois positions différentes :

- .align-items-start : les colonnes seront alignées en début (en haut) de la ligne ;
- .align-items-center : les colonnes vont être alignées au centre de la ligne ;
- .align-items-end : les colonnes seront alignées en fin (en bas) de la ligne.

Pour aligner chaque colonne individuellement, nous allons cette fois-ci plutôt utiliser les classes .align-self-* qu’on va utiliser avec chaque élément cette fois-ci.

L’alignement horizontal des colonnes

On va principalement vouloir aligner horizontalement des colonnes dans une ligne dans le cas où colonnes créées n’occupent pas tout l’espace de la ligne, c’est-à-dire dans le cas où il reste de l’espace à distribuer entre les colonnes.

Cela va être le cas si on utilise des classes .col-{nombre} pour chaque colonne de la ligne et que la somme des nombres fait moins de 12.

On va pouvoir aligner horizontalement nos colonne dans la ligne grâce aux classes .justify-content-* qu’on va devoir appliquer à la ligne en soi. Nous allons pouvoir utiliser les classes suivantes :

- .justify-content-start : les colonnes vont se positionner en début de ligne (à gauche par défaut) ;
- .justify-content-center : les colonnes vont se positionner au milieu de la ligne ;
- .justify-content-end : les colonnes vont se positionner en fin de ligne (à droite par défaut) ;
- .justify-content-around : les colonnes vont être réparties équitablement dans la ligne. Chaque colonne va posséder le même écart à droite et à gauche, même celles situées contre les bords de la ligne (l’espacement entre le bord de la ligne et la première / dernière colonnes sera donc deux fois plus petit que l’espacement entre deux colonne) ;
- .justify-content-between : les colonnes vont être réparties équitablement dans la ligne. La première colonne va être collée contre le début de la ligne et la dernière va être collée contre la fin de celle-ci.

Les paddings entre les colonnes

Par défaut, Boostrap génère des espaces situés de part et d'autre du contenu de chacune de nos colonnes. De manière effective, elles sont représentées par une propriété padding. Elles servent à espacer les contenus des différentes colonnes.

Par défaut, un padding gauche et droit de 15px est affecté à chaque colonne créée. Cette marge intérieure est contrebalancée au niveau de la ligne en affectant automatiquement un margin gauche et droit de -15px (négatif donc) sur chaque ligne. Cela permet d'obtenir une consistance dans l'alignement des éléments.

On peut néanmoins choisir de supprimer les gouttières entre les colonnes en appliquant la classe .no-gutters à l'élément représentant la ligne en soi. Cela va supprimer le padding gauche et droit des éléments enfants directs de l'élément représentant la ligne. Les marges négatives appliquées par défaut sur la ligne vont également être supprimées.

Gérer l’ordre d’affichage des colonnes

Les classes Bootstrap .order-* vont nous permettre de modifier l’ordre visuel de notre contenu.

Nous allons ainsi pouvoir choisir dans quel ordre doit apparaitre notre contenu en passant une classe .order-1, .order-2... .order-12 à chacun de nos éléments représentant nos colonnes et qui va déterminer l’ordre d’affichage visuel de celles-ci dans la ligne.

Le principe est simple : une colonne avec une classe .order-* possédant un chiffre plus petit s’affichera avant une colonne avec une classe .order-* possédant un chiffre plus grand.

De plus, vous devez savoir que les colonnes ont par défaut un order : 0 (pour les colonnes dont l’ordre n’est pas spécifiquement défini avec une classe .order-).

Notez également que les classes .order-* sont responsive et supportent donc les breakpoints. Concrètement, cela signifie que nous allons donc pouvoir choisir des ordres d’affichage différents selon la taille de l’écran d’un visiteur en ajoutant -sm-, -md-, etc. à ces classes.

Imbriquer des lignes

Pour créer des designs complexes, on va également pouvoir imbriquer des lignes dans d'autres lignes ou plus exactement imbriquer des lignes dans des colonnes.

Cela va nous permettre d'avoir en quelques sortes plusieurs « niveaux de grilles ».

Pour faire cela, il va suffire d'ajouter un ou plusieurs éléments avec des classes .row comme enfant direct d'un élément portant une classe de type .col-*.

Les lignes imbriquées dans d'autres colonnes vont agir comme de nouvelles sous-grilles et vont suivre les mêmes règles que précédemment et également être implicitement découpées en 12 colonnes de base.

Intégrer une vidéo Responsive

Intégrer des médias vidéos est de plus en plus commun avec l'essor de plate-formes telles que Youtube ou Viméo. Aujourd'hui, rien de plus simple avec Bootstrap pour intégrer une vidéo sur votre blog ou votre site web.

Bootstrap propose par défaut une liste de ratios permettant d'afficher votre vidéo avec le ratio qui vous semble le plus approprié:

- 21:9
- 16:9 (Le plus utilisé aujourd'hui sur le web)
- 4:3
- 1:1

Pour intégrer une vidéo en responsive sur votre site web, il faut commencer par utiliser une div principale portant la classe embed-responsive qui englobera votre contenu.

Après avoir défini votre div principale qui englobera votre vidéo, nous devons intégrer la balise <iframe> qui contiendra la source de la vidéo, ses options et autres paramètres.

Les classes et propriété CSS

Bootstrap offre une quantité absolument monstrueuse de possibilité ! Vous pouvez l'utiliser pour remplacer énormément de propriété CSS par de simple classe Bootstrap comme les couleurs, les fonds de couleurs, les marges, les bordures, les positionnements, les ombres etc.. etc..

Cette formation n'est pas accès uniquement sur Bootstrap, nous n'allons donc pas les détailler, mais sachez simplement que cela existe.

Pour les plus curieux d'entre vous, rendez-vous sur la documentation officielle de Boostrap ou se trouve toutes les possibilités