## Implémentez votre structure de pages



Dans le chapitre précédent, vous avez vu comment une page peut être décomposée en sections d'en-tête (header), de pied de page (footer) et de contenu.

Maintenant que vous avez un bon aperçu de la manière dont vous allez implémenter la structure de vos pages, il est temps de les créer !

### Découvrez les conteneurs

Commençons par l'élément de mise en page le plus basique dans Bootstrap, le **conteneur**(container). Envelopper vos contenus dans un élément   <div>  avec la classe   .container  permet de centrer votre contenu à l’écran en y ajoutant des marges horizontales automatiques.

De ce fait, la classe   .container  permet d’aérer la lecture horizontale du contenu de votre site web.

<div class="container">

<!-- Ici notre contenu -->

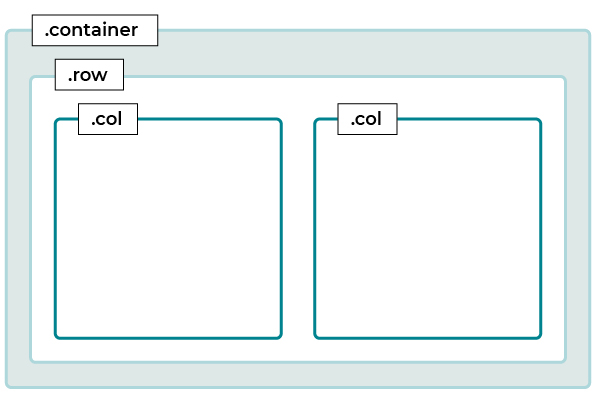
</div>

Si vous voulez que le contenu occupe toujours 100 % de la largeur de l’écran, vous pouvez utiliser la classe   .container-fluid    à la place. Cela permet d’enlever les marges horizontales appliquées par défaut par le framework.

Depuis Bootstrap 5, de nouveaux containers sont arrivés :  .container-{breakpoint}  (exemple :   .container-lg). Ils permettent d’adapter votre “container” en fonction de la taille d’écran de l’utilisateur. Rendez-vous sur la [**documentation Bootstrap 5**](https://getbootstrap.com/docs/5.0/layout/containers/) pour en savoir plus.

### Maîtrisez les grilles de Bootstrap

Bootstrap 5 possède un système de grille puissant et flexible qui permet de créer tous les types de mise en page. Cette grille utilise une série de **lignes** (en anglais, row) et de **colonnes** (en anglais, column) pour mettre en page le contenu. Une ligne permet d’envelopper une ou plusieurs colonnes qui intègrent du contenu.

Imbrication des éléments de la grille de Bootstrap 5

OK, mais qu’est-ce que ça veut dire, en pratique ?

Un élément   <div>  de ligne est déclaré en lui attribuant une classe   .row. Pour chaque colonne de la ligne, un élément   <div>  enfant du   <div>  de ligne doit se voir attribuer l'une des classes de colonnes de Bootstrap 5. La première de ces classes de colonnes est la classe   .col.

Examinons cet extrait de code :

<div class="container">

<div class="row">

<div class="col">

Première colonne

</div>

<div class="col">

Deuxième colonne

</div>

</div>

</div>

Vous remarquez qu'il y a un élément  <div>  parent  .row  enveloppant deux  <div>  enfants  .col. Bootstrap 5 possède une grille à 12 colonnes. Dans ce cas précis, Bootstrap attribuera **automatiquement** la moitié des colonnes à chacun des éléments   <div>  de classe   .col, donnant alors une largeur de 6 colonnes par  <div>.



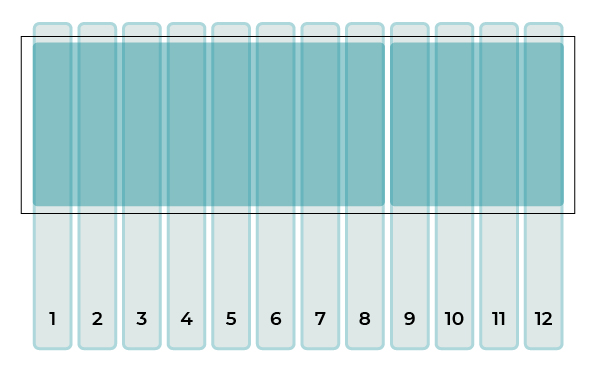
Le contenu sera organisé sur une grille de 12 colonnes

En d'autres termes, cela représente la moitié de la largeur de la ligne. De ce fait, Bootstrap divisera systématiquement les 12 colonnes de manière égale entre les éléments   <div>  qui ont la classe   .col.

Mais, et si je veux indiquer explicitement le nombre de colonnes à la place de Bootstrap ?

Par exemple, vous pouvez avoir une mise en page qui ne possède pas de colonnes de taille égale. Supposons que vous disposiez d'une ligne, que vous vouliez qu'une colonne de cette ligne occupe deux tiers de sa largeur, et qu'une autre colonne occupe le tiers restant.

Dans ce cas, en tenant compte du fait qu'une ligne de la grille de Bootstrap peut être divisée en un maximum de 12 colonnes, vous voulez que l'une des   <div>    .col  occupe huit colonnes, et que l'autre occupe quatre colonnes.

Une div qui occupe 8 colonnes, une autre div qui occupe 4 colonnes sur les 12

Pour ce faire, ajoutez chacun de ces nombres à la classe   .col  correspondante, comme indiqué dans cet extrait :

<div class="container">

<div class="row">

<div class="col-8">

Première colonne

</div>

<div class="col-4">

Deuxième colonne

</div>

</div>

</div>

Pour en savoir plus sur les grilles, vous pouvez consulter [la documentation Bootstrap](https://getbootstrap.com/docs/5.0/layout/grid/).

### Créez des aplats de couleurs grâce aux classes utilitaires de Bootstrap 5

Vous allez utiliser les classes utilitaires (en anglais, utilities) de Bootstrap 5 pour mettre des aplats de couleurs et nos marges internes (padding) et externes (margin) entre nos éléments.

Les classes utilitaires sont des classes créées par Bootstrap 5, qui permettent d’ajouter des propriétés CSS à vos éléments, telles que des :

* background-color  (comme  .bg-light) ;
* margin  (comme  .m-5) ;
* padding  (comme  .p-5) ;
* border  (comme  .border-top).

Entre autres ! Vous pouvez consulter toutes les classes sous la liste “Utilities” dans la [documentation de Bootstrap](https://getbootstrap.com/docs/5.0/utilities/).

En plus de la classe **.bg-light**, Bootstrap comprend un certain nombre d'autres classes **.bg-\***  que vous pouvez utiliser pour fournir aux éléments des couleurs d'arrière-plan prédéfinies. La liste complète des couleurs est disponible dans la [**documentation de Bootstrap**](https://getbootstrap.com/docs/4.3/utilities/colors/#background-color).

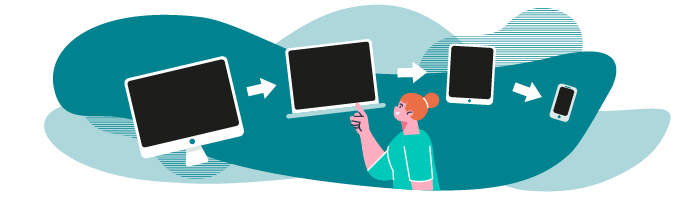
Si on prend l’exemple du menu :

À ce stade, un menu sans liens de navigation

Vous voyez qu’il y a une première barre gris clair qui est mise en page grâce à la classe utilitaire  .bg-light. La barre gris foncé est quant à elle mise en page grâce à la classe  .bg-secondary  pour la couleur et  .py-4  pour le padding (marge interne).

Nous verrons d’autres classes utilitaires tout au long de ce cours, donc n’hésitez pas à les parcourir dans la documentation et à les tester pendant les différentes mises en pratique.

## Rendez votre mise en page responsive



### Découvrez les breakpoints

Nous vivons à une époque où le progrès est de plus en plus rapide. On a l'impression qu'un nouveau périphérique sort tous les jours sur le marché ! Bien que cette cadence puisse être excitante, il est également épuisant de suivre toutes les variations des tailles d'écran que vous devez prendre en compte pour vous assurer que l'aspect et le fonctionnement des sites web que vous créez sont corrects.

Souvenez-vous, **une page responsive**est une page qui réagit à différentes tailles d'écran de l'utilisateur, en modifiant la mise en page des composants et du contenu à ajuster.

La bonne nouvelle, c'est qu'il n'est pas nécessaire de créer une mise en page différente pour chaque taille d'écran. Vous pouvez vous faciliter la vie, tout en créant une bonne expérience utilisateur. Pour cela, il est possible de regrouper toutes les tailles d'écran dans des compartiments différents, c'est-à-dire en utilisant des plages de tailles, et en générant des mises en page pour chacune d’elles.

Prenons un exemple type : un site web standard a besoin d'une mise en page pour des périphériques avec de grands écrans (ordinateurs de bureau), une autre pour des périphériques avec des écrans de taille moyenne (tablettes et petits ordinateurs portables) et enfin pour des périphériques plus petits (smartphones).

Quel doit être le nombre exact de pixels de chacune de ces tailles ?

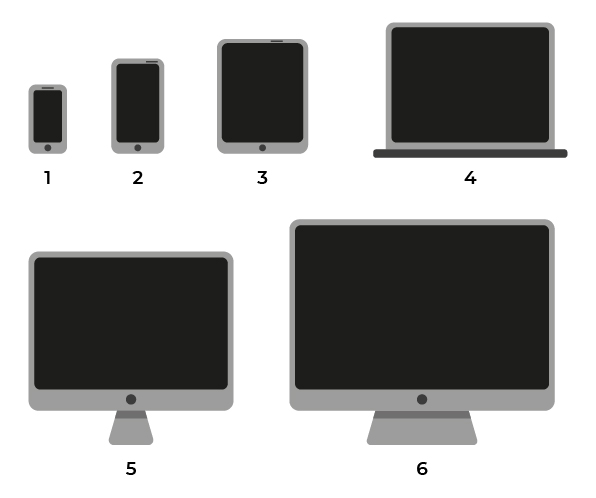
En tant que développeur front-end, je suis (comme vous) très à cheval sur les détails, et il me faut des définitions plus précises que grandes, moyennes, petites, etc. Je dois connaître les mesures exactes de chacune d'entre elles.

Conformément aux recommandations standard, les développeurs du framework ont défini des ***breakpoints* (points d’arrêt)** afin de compartimenter les différentes tailles d’affichage :

|  |  |  |  |
| --- | --- | --- | --- |
| **Taille** | **Breakpoint (Point d'arrêt)** | **Modificateur de classe** | **Équivalent écrans** |
| Très petite | < 576 px | aucun | Petit Smartphone |
| Petite | ≥ 576 px | -sm | Smartphone |
| Moyenne | ≥ 768 px | -md | Tablette |
| Grande | ≥ 992 px | -lg | Grande tablette et PC 13’’ |
| Très grande | ≥ 1 200 px | -xl | Écran 15’’ |
| Très très grande | ≥1400 px | -xxl | Écran > 15’’ |

**Un breakpoint (point d'arrêt)**est un point de discontinuité, modification ou interruption. Dans le contexte de conception et de développement frontend, un point d'arrêt est le point à partir duquel la mise en page change. Généralement, un nombre de pixels représente la largeur de la taille d’écran affichée à laquelle le point d'arrêt se produit.

Comme vous pouvez le voir, il y a six tailles définies dans Bootstrap 5, ce qui vous offre bon nombre d’options.



De nombreuses classes Bootstrap pour la mise en page et les composants vous permettent d'ajouter un modificateur lié à la taille. Cela permet de contrôler la façon dont ils sont restitués, selon la taille d’écran affichée.

### Mettez le responsive en pratique

Nous avons beaucoup parlé théorie. Vous êtes sûrement impatient de mettre en pratique tout ce que vous avez appris !

Commençons par examiner la manière dont la réactivité fonctionne pour la grille de Bootstrap. La vidéo suivante montre comment une mise en page s’adapte en cas de changement de taille d’écran marquée par un point d’arrêt :

Même si Bootstrap comporte plusieurs tailles prédéfinies, vous pouvez choisir d’adapter le nombre utilisé en fonction de vos besoins ou de la structure de la page web. N'oubliez pas qu'en plus de l'effort supplémentaire que vous devrez fournir pour développer et tester chaque mise en page, un concepteur de site web devra probablement apporter sa contribution à chacune d'entre elles.

Pour appliquer les modificateurs de classes appropriés pour ces options de mise en page, il est important de comprendre qu'ils fonctionnent **par le haut**, c'est-à-dire de la plus petite taille d’écran vers la plus grande.

C'est l'approche **mobile-first**, vous vous rappelez ?

Cela signifie que **si vous n'ajoutez pas de modificateur de classe relatif à une taille d’écran supérieure, Bootstrap affichera automatiquement** les éléments et composants de la grille dans les plus petits écrans, et conservera la même mise en page, même si la taille de l'écran augmente.

Par exemple, si une   <div>  possède une seule classe ayant le modificateur   -md, celui-ci s’appliquera à tous les écrans d’une taille minimum de 768 px et supérieure.

Si cette même    <div>  utilise un second modificateur de classe d’une taille supérieure à   -md  comme   -lg  (largeur ≥ à 992 px) ou  -xl  (largeur ≥ à 1 200 px), ces derniers prendront effet sur le modificateur   -md  lorsque l’écran affiché aura atteint les points d’arrêt respectifs.

Chaque modificateur de classe s’ajoute au même niveau de la   <div>. L’adaptation responsive de la page web s’effectue automatiquement, dès lors qu’un point d’arrêt est franchi par la taille d’écran affichée.

<div class=”row”>

<div class=”col-6 col-md-4 col-lg-3”></div>

<div class=”col-6 col-md-8 col-lg-9”></div>

</div>

Ci-dessus, nous avons une ligne avec deux   <div>  enfants qui occupent des largeurs différentes selon la taille de l’écran. Le mode d'interprétation des classes pour la première des   <div>  enfants de la ligne est le suivant :

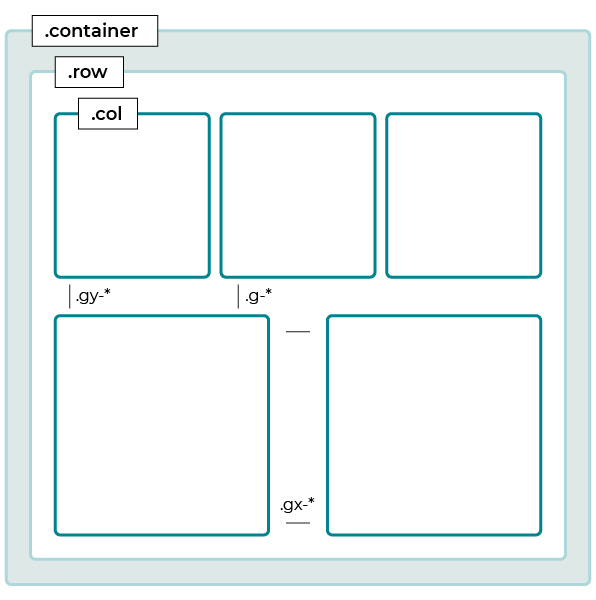
* la classe  .col-6  signifie que pour tous les écrans sans restriction de taille minimum ou maximum, attribuez six colonnes à cette   <div>  ;
* la classe  .col-md-4  signifie que pour tous les écrans avec une largeur minimum supérieure ou égale à 768 px, attribuez quatre colonnes à cette   <div>  ;
* la classe  .col-lg-3  signifie que pour tous les écrans avec une largeur minimum supérieure ou égale à 992 px, attribuez trois colonnes à cette   <div>.

Une logique similaire s'applique à la seconde   div  enfant.

Notez que le nombre total de colonnes attribuées à chaque modificateur de classe s'ajoute encore aux 12 colonnes de la grille de Bootstrap. Dans l'exemple précédent, cela signifie que lorsque vous utilisez un modificateur de classe sur une   <div>, n’oubliez pas de le répercuter sur la seconde   <div>  du même niveau pour un total de 12 colonnes :   .col-md-4   +  .col-md-8    =    .col-md-12.

### Espacez des colonnes avec des gouttières

Les espacements entre les différentes colonnes de la grille sont appelés **gouttières** (gutters, en anglais).

Ajoutez de l'espace entre vos éléments avec les gouttières

Ces dernières sont ajustables grâce à la classe utilitaire  .g-{1-5}. L’espacement vertical par défaut des  .col  de Bootstrap 5 est équivalent à l’espacement donné par la classe  .g-4 ; ainsi dans l’exemple précédent, en ajoutant la classe  .g-2  à la  .row, cela a pour effet de réduire l’espacement entre les colonnes.

Voici un exemple de gouttières dans le code :

<div class="container px-4">

<div class="row gx-5">

<div class="col">

<div class="p-3 border bg-light">Custom column padding</div>

</div>

<div class="col">

<div class="p-3 border bg-light">Custom column padding</div>

</div>

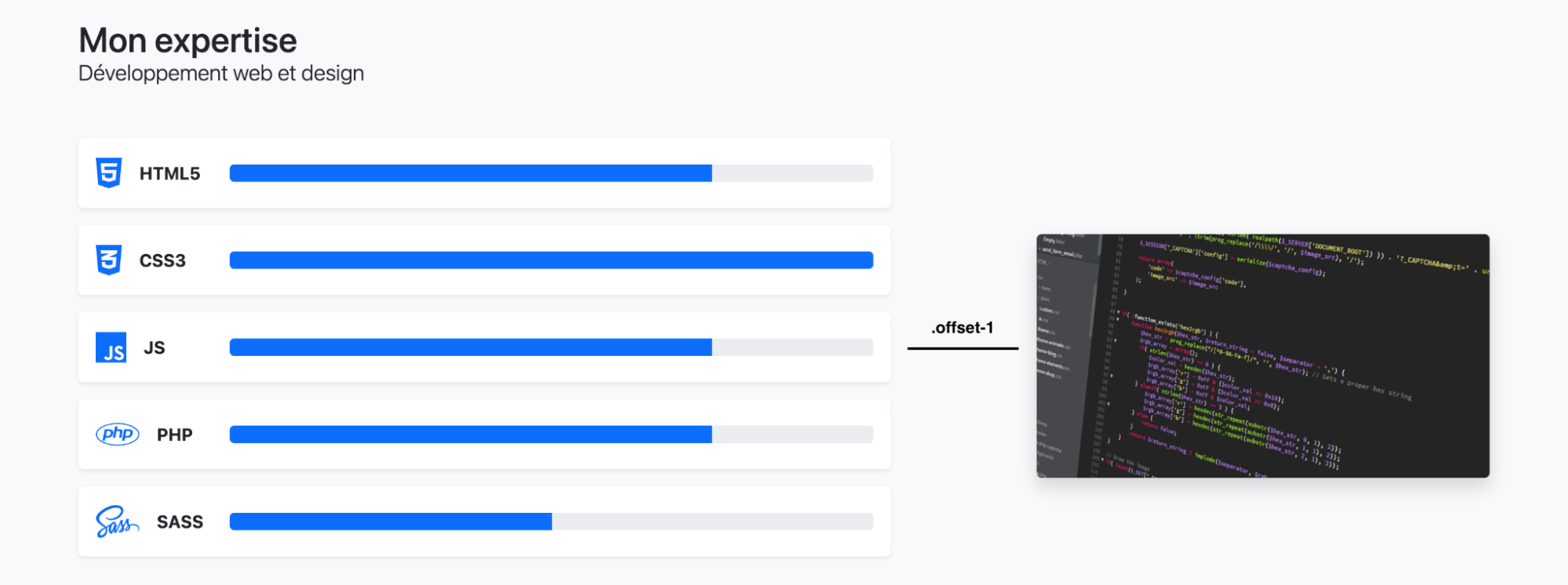
</div>

</div>

Pour en savoir plus sur les gouttières, je vous invite à consulter [la documentation Bootstrap](https://getbootstrap.com/docs/5.0/layout/gutters/).

### Laissez des colonnes vides grâce aux  offset

Pour certains designs, vous aurez besoin de laisser l’équivalent d’une colonne (ou plus) vide entre 2 colonnes. Ce qui est le cas pour la partie Expertise de notre portfolio :

Entre les deux éléments qui composent la section Expertise, il faudra ajouter de l'espace avec offset

Pour ce faire, Bootstrap met à votre disposition la classe  .offset-{0-11}  qui va permettre de pousser de 0 à 11 colonne(s) sur la droite le contenu de la colonne sur laquelle vous ajoutez cette classe.

Dans la documentation, Bootstrap présente un exemple des  offset  :

<div class="container">

<div class="row">

<div class="col-md-4">.col-md-4</div>

<div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>

</div>

<div class="row">

<div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>

<div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>

</div>

<div class="row">

<div class="col-md-6 offset-md-3">.col-md-6 .offset-md-3</div>

</div>

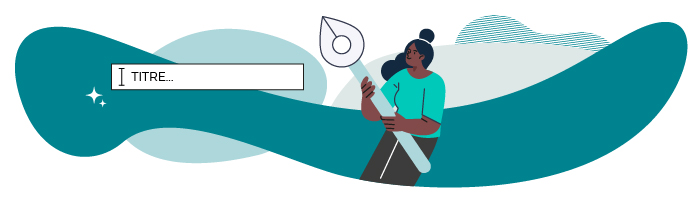
</div>

Ce code a pour rendu :

Les différentes colonnes espacées grâce aux offsets

Dans le cas de votre portfolio, il faudra utiliser la classe  .offset-1  sur la deuxième colonne pour pouvoir recréer la structure attendue. Vous pouvez en savoir plus sur comment utiliser l’offset avec des colonnes dans [la documentation Bootstrap](https://getbootstrap.com/docs/5.0/layout/columns/#offsetting-columns).

## Personnalisez votre page



Dans la partie 2, nous avions placé nos titres de section en même temps que la structuration de notre page. Nous allons maintenant voir comment les personnaliser grâce aux classes utilitaires de Bootstrap 5.

Quand on observe les classes utilitaires de Bootstrap, on remarque qu’elles ne sont pas construites n’importe comment. Bien souvent, la classe utilitaire est composée des initiales de la propriété CSS qu’elle représente, **concaténées** (collées) à la valeur qu’elle ajoute.

Pour les propriétés liées au texte, il existe plusieurs classes que vous pouvez utiliser pour modifier l’épaisseur (  font-weight  ) ou le style (  font-style  ) du texte :

|  |  |  |
| --- | --- | --- |
| **Valeur** | **C’est-à-dire...** | **Classe utilitaire** |
| bold | texte en gras | fw-{breakpoint}-bold |
| bolder | texte encore plus gras | fw-{breakpoint}-bolder |
| normal | texte normal | fw-{breakpoint}-normal |
| light | texte léger | fw-{breakpoint}-light |
| lighter | texte encore plus léger | fw-{breakpoint}-lighter |
| italic | texte en italique | fs-{breakpoint}-italic |
| normal | texte normal | fs-{breakpoint}-normal |

Vous remarquez que comme pour toute autre classe utilitaire, nous pouvons ajouter des breakpoints (points d’arrêt) pour rendre le texte responsive.

Dans le cas de notre   h1, nous allons lui ajouter la classe   .fw-bold, ce qui veut dire  font-weight: bold;. En suivant cette logique, on peut ajouter la classe suivante,  .fw-light, à notre  h2, ce qui a pour effet de réduire l’épaisseur de la police d’écriture.

<h1 class="fw-bold">Je suis John Doe Développeur et Designer Web</h1>

<h2 class="fw-light">Bienvenue dans mon univers créatif</h2>

Vous pouvez voir toutes les classes utilitaires liées aux propriétés du texte dans [la documentation Bootstrap](https://getbootstrap.com/docs/5.0/utilities/text/#font-weight-and-italics).

Vous vous demandez s'il existe de bonnes pratiques pour guider la mise en page de votre site ? Rendez-vous au cours [**Codez un site web accessible avec HTML & CSS**](https://openclassrooms.com/fr/courses/6691451-codez-un-site-web-accessible-avec-html-css?status=published) !

### Ajoutez de la logique liée au sens de lecture

Depuis Bootstrap 5, le sens de lecture d’une page a pris une importance capitale dans la logique de développement du framework. Même s’il est vrai qu’une grande majorité des sites que nous consultons en France se lisent de gauche à droite, ce n’est pas nécessairement le cas partout !

Dans un souci d’accessibilité, les développeurs de Bootstrap 5 ont décidé de supprimer les termes   left  et  right  au profit de  start  et  end.

L’avantage est que si l’on change le sens de lecture de notre site pour l’adapter à un public dont le sens de lecture est de droite à gauche, les alignements et autres classes utilitaires utilisés avec la sémantique  start  passent à droite, et ceux avec  end  à gauche.

En plus concret, si vous souhaitez aligner vos textes à gauche, vous pouvez utiliser la classe utilitaire  .text-start, et pour un alignement à droite vous devez utiliser  text-end. Pour la propriété CSS  text-align, on retrouve donc 3 valeurs :

|  |  |  |
| --- | --- | --- |
| **Valeur** | **C’est-à-dire...** | **Classe utilitaire** |
| center | texte centré | text-{breakpoint}-center |
| start | texte aligné à gauche (pour un site avec le sens de lecture de gauche à droite) | text-{breakpoint}-start |
| end | texte aligné à droite (pour un site avec le sens de lecture de gauche à droite) | text-{breakpoint}-end |

Les classes utilitaires  text-align  ne suivent pas la logique de construction des autres classes utilitaires car elles existent depuis les premières versions de Bootstrap, bien avant la création du système de classes utilitaires actuel.

Vous pouvez voir toutes les classes utilitaires liées à la propriété  text-align  dans [la documentation Bootstrap](https://getbootstrap.com/docs/5.0/utilities/text/#text-alignment).

### Découvrez comment fonctionnent les boutons Bootstrap 5

Les **boutons** (buttons, en anglais) constituent l'une des formes les plus courantes d'interaction sur les sites web. Ils indiquent à vos utilisateurs qu'une action est possible sur la page, comme l’envoi d’un formulaire, l’ajout au panier d’un objet, etc.

Pour afficher un bouton dans Bootstrap, utilisez un élément   <button>  ou    <a>  avec une  classe  .btn, ainsi qu'une deuxième classe pour définir la couleur d'arrière-plan.

Les classes relatives aux couleurs disponibles sont affichées ci-dessous :

Exemples de classes de couleurs btn-\*

Vous pouvez afficher un bouton utilisant l’élément   <button>  (pour vos formulaires)  ou   <a>  (pour vos liens) avec la couleur d'arrière-plan primaire (bleu par défaut), avec le code HTML suivant :

<button type="button" class="btn btn-primary">Bouton</button>

<a class="btn btn-primary" href="#" role="button">Lien</a>

À présent, nous allons mettre en forme notre premier bouton “Mon expertise web”. Ajoutez-lui les classes  .btn  et  .btn-primary, cela aura pour effet d’appliquer le style des boutons Bootstrap 5.



Vous pouvez consulter [la documentation Bootstrap](https://getbootstrap.com/docs/5.0/components/buttons/) pour en savoir plus sur les boutons.

### Ajoutez des icônes

Boostrap 5 a son propre système de font d'icônes que l’on peut intégrer dans notre interface utilisateur pour, par exemple, lier nos réseaux sociaux au site web.

Pour les besoins de notre projet, nous allons également ajouter le jeu d'icônes de Font Awesome en ajoutant cette ligne de code à la fin de votre    <head>  :

<!-- Font Awesome -->

<script src="https://kit.fontawesome.com/14273d579a.js" crossorigin="anonymous"></script>

Pour utiliser les icônes de Bootstrap, il suffit d’ajouter une balise  <link>  dans le  <head>  de notre page.

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.5.0/font/bootstrap-icons.css">

Ensuite, rendez-vous dans le catalogue d'icônes de Bootstrap via la documentation dans la rubrique “Extend” puis “Icons”, dans leur [catalogue d'icônes Bootstrap 5](https://icons.getbootstrap.com/).

Pour afficher une icône de LinkedIn, vous pouvez ajouter cette ligne de code dans la div  qui contiendra vos icônes :

<a href=”mon lien linkedin” target=”\_blank”>

<i class="bi bi-linkedin"></i>

</a>

### Allez plus loin avec la documentation Bootstrap 5

Il existe de nombreuses autres propriétés CSS qui ont été couvertes par le système de classes utilitaires de Bootstrap 5. Le meilleur moyen de les découvrir est de parcourir la documentation Bootstrap 5 dans les rubriques “[Utilities](https://getbootstrap.com/docs/5.0/utilities/api/)” et “[Helpers](https://getbootstrap.com/docs/5.0/helpers/clearfix/" \t "_blank)”.

On y trouve différentes classes pour les couleurs de typographie, les backgrounds, les bordures, les tailles de texte, etc.

## Créez de la navigation responsive



Maintenant que vous avez appris à créer la structure de vos mises en page à l'aide de la grille de Bootstrap 5, il est temps d'ajouter des composants à cette structure !

La **navigation** est un des points majeurs de tout site web. C’est un excellent point de départ pour utiliser des composants Bootstrap.

### Ajoutez de la navigation à votre site

Le framework comprend un composant de barre de navigation puissant et réactif. Il est implémenté à l'aide d'un élément   nav  possédant la classe  .navbar.

<nav class="navbar">

...

</nav>

Les barres de navigation nécessitent également une classe   .navbar-expand  pour la réduction responsive du menu sur Smartphones et petits écrans. La classe   .navbar-expand  affecte la largeur de la zone de visualisation pour laquelle la barre de navigation est développée ou réduite. Il est possible d’adapter le breakpoint de la barre de navigation à l’aide d’un modificateur de classe  {  -sm  ,  -md ,  -lg ,  -xl, -xxl  }  venant s'ajouter à la classe   .navbar-expand  comme suit :   .navbar-expand-md.

Pour l'instant, ajoutez la classe   navbar-expand  sans modificateur de classe relatif à la taille. Nous examinerons ultérieurement la création d'une barre de navigation réduite pour des écrans plus petits.

<nav class="navbar navbar-expand">

...

</nav>

Pour la plupart des sites web, la marque est affichée le long de la barre de navigation principale. Bootstrap propose une classe   .navbar-brand  prévue à cet effet dans le composant de la barre de navigation. Ajoutez le nom du site dans l'élément prévu à cet effet, comme ci-dessous :

<nav class="navbar navbar-expand">

<a class="navbar-brand" href="...">Prénom Nom</a>

</nav>

La mise en page de base de la marque n’est pas très originale. C’est pourquoi nous allons utiliser les classes utilitaires de Bootstrap 5 pour la rendre un peu plus stylée !

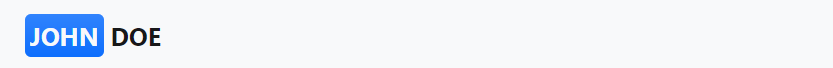
Pour rappel, toutes les [**classes utilitaires sont dans la documentation Bootstrap 5**](https://getbootstrap.com/docs/5.0/utilities/api/), donc n’hésitez pas à vous y référer en cas de besoin.

Dans notre cas, nous allons ajouter les classes  .text-uppercase .fw-bold  (texte en majuscules + texte en gras) à notre  a.navbar-brand. Ensuite, nous allons entourer notre prénom d’une balise  span  afin de lui donner d'autres propriétés de mise en page. Pour ce faire, nous ajoutons ces classes à notre élément  span,  .bg-primary .bg-gradient .p-1 .rounded-3 .text-light  :

<a class="navbar-brand text-uppercase fw-bold" href="/index.html">

<span class="bg-primary bg-gradient p-1 rounded-3 text-light">John</span> Doe

</a>

Le logo du navbar

Et voilà, nous avons ajouté un peu de personnalisation à notre logo sans taper une seule ligne de CSS !

Passons maintenant aux liens de navigation. Les éléments de navigation doivent être placés dans une liste non ordonnée contenant les éléments  <ul>,   <li>  et   <a>, selon les classes indiquées dans cet extrait :

<nav class="navbar navbar-expand">

<div class="container">

<a class="navbar-brand text-uppercase fw-bold" href="/index.html">

<span class="bg-primary bg-gradient p-1 rounded-3 text-light">John</span> Doe

</a>

<ul class="navbar-nav">

<li class="nav-item active">

<a class="nav-link" href="#johndoe">Qui suis-je ?</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#expertise">Expertises</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#portfolio">Portfolio</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#contact">Contact</a>

</li>

</ul>

</div>

</nav>

Vous remarquerez que le premier élément   <li>  dans l'extrait de code comprend une classe   .active. Cela permet d'indiquer l'un des éléments de navigation comme l'élément “actif”, actuellement sélectionné. Vous pouvez créer le code HTML de votre barre de navigation afin de vous assurer qu’un seul élément utilise cette classe.

Il y a un autre élément que vous devez bien connaître maintenant, il s’agit du  div.container. L’effet est exactement le même que partout ailleurs dans la page, il sert à contenir les éléments de la  navbar.

Désormais, votre barre de navigation est personnalisée avec le nom du site et les liens de navigation. Ensuite, ajoutons de la couleur !

Pour modifier les couleurs de l'arrière-plan et le texte dans la barre de navigation, vous pouvez ajouter des styles CSS créés de toutes pièces. Cependant, vous avez également la possibilité d’appliquer les classes utilitaires de couleurs d'arrière-plan de Bootstrap à vos différents éléments. Celles-ci sont particulièrement utiles lors du prototypage, car elles vous permettent d'ajuster la couleur d'arrière-plan d'un élément (pas uniquement les barres de navigation !) à l'aide des [préréglages de Bootstrap.](https://getbootstrap.com/docs/5.0/utilities/background/)

Comme vous l'avez vu dans le chapitre précédent, l'arrière-plan de la barre de navigation est gris clair. Vous utiliserez donc la classe utilitaire  .bg-light  pour modifier sa couleur :

<nav class="navbar navbar-expand bg-light">

...

</nav>

Les classes   .navbar-light  et   .navbar-dark  constituent un autre ensemble pratique dans la gestion des couleurs Bootstrap. Ces classes ajustent la couleur des éléments de la barre de navigation, comme la marque et les liens de navigation, pour offrir un contraste à l'arrière-plan choisi. Lorsque vous utilisez une barre de navigation de couleur foncée, il vous suffit d’ajouter la classe   .navbar-dark. À l’inverse, comme dans le cas de notre site, ajoutez la classe   .navbar-light  à une barre de navigation de couleur claire.

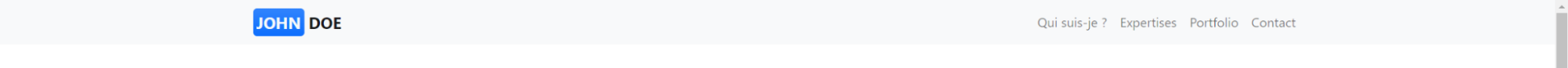
Voici comment faire :

<nav class="navbar navbar-expand bg-light navbar-light">

...

</nav>

Maintenant, avec toutes les modifications apportées à la section d'en-tête et à la barre de navigation, votre prototype de page d'accueil doit ressembler à ceci :

La barre de navigation telle qu’elle doit être au chapitre 2 de la partie 3

Les choses commencent à prendre forme ! Bravo !

### Créez une navigation responsive

Maintenant que vous avez une configuration de barre de navigation basique, il est temps de vérifier qu'elle soit **responsive**. Étant donné que Bootstrap est un framework mobile-first et que les classes de la barre de navigation dans le code utilisées jusqu'à présent ne comprennent pas de modificateurs de classes relatifs à la taille, l'aspect de la barre de navigation sera identique pour toutes les tailles d'écran.

Lorsque vous ajoutez du contenu à la barre de navigation, tel que des logos et des liens supplémentaires, elle finit par être trop grande pour des écrans plus petits. Pour la rendre responsive à des tailles d'écran différentes, modifiez-la de sorte qu'elle soit **réduite** pour des écrans plus petits et **développée** pour des écrans plus grands.

Pour le site portfolio, nous voudrons que la navigation soit comme dans l'image ci-dessous :

Pour la version mobile, la navigation se réduit

Selon le comportement de la navigation responsive pour notre exemple de site ci-dessus, nous pouvons en tirer quelques conclusions :

* lorsque la largeur de la zone de visualisation est grande au minimum, le menu de navigation est développé et affiche l’intégralité du contenu ;
* lorsque la largeur de la zone de visualisation est inférieure au point d'arrêt utilisé ( {-sm|-md|-lg|-xl|-xxl}), le menu de navigation est masqué et un bouton “bascule de navigation” s’affiche. Ce dernier permet de réduire ou développer le menu.

Cela fait beaucoup à apprendre ; mais heureusement, Bootstrap simplifie extrêmement bien les choses.

La première étape dans l'implémentation de la navigation responsive consiste à utiliser le **modificateur** de classe relatif à la taille dans la classe   .navbar-expand  de l'élément   <nav>. N'oubliez pas que les modificateurs de classe dans Bootstrap fonctionnent **vers le haut**. Utiliser la classe   .navbar-expand  sans modificateur de classe appliquera la classe à toutes les tailles en partant de la taille extra small au minimum.

Pour rendre la navigation responsive de la manière spécifiée ci-dessus, vous devez appliquer le modificateur de classe qui développe le menu de la barre de navigation pour de grands écrans minimum. Vous devrez donc le remplacer par .navbar-expand-md.

<nav class="navbar navbar-expand-md bg-light navbar-light fixed-top">

...contenu de la barre de navigation...

</nav>

Vous devez ajouter la classe  .fixed-top  pour faire en sorte que votre navigation soit fixée en haut de l’écran pendant le défilement de la page. Cette classe utilitaire fait partie des “Helpers”. Pour en savoir plus, rendez-vous à [**la documentation Bootstrap**](https://getbootstrap.com/docs/5.0/helpers/position/).

Ensuite, vous devez ajouter le **bouton bascule (*toggle*) de navigation**qui permettra d’afficher ou masquer le menu de navigation de votre site web. Pour une navigation responsive, intégrez le bouton juste après l’élément  <a>  classe  .navbar-brand, avec les attributs et le contenu suivants :

<button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

Comme vous pouvez le constater, le bouton comprend une classe   .navbar-toggler, ainsi que des attributs data tels que   data-bs-toggle="collapse"  et   data-bs-target="#navbarNav". Ces attributs permettent de cibler le bon élément   <div>  utilisant la classe   .collapse  et l’identifiant  #navbarNav, sans écrire une seule ligne de JavaScript.

Pour ce faire, ajoutez les classes   .collapse  et   .navbar-collapse  à la <div>  qui enveloppe la liste des liens de navigation.

Enfin, ajoutez l’attribut identifiant ayant la même valeur que celle de l'attribut  data-bs-target  du bouton Bascule, dans ce cas   navbarNav  :

<div class="collapse navbar-collapse justify-content-end" id="navbarNav">

<ul class="navbar-nav">

<li class="nav-item">

<a class="nav-link" href="#johndoe">Qui suis-je ?</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#expertise">Expertises</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#portfolio">Portfolio</a>

</li>

<li class="nav-item">

<a class="nav-link" href="#contact">Contact</a>

</li>

</ul>

</div>

Vous remarquerez peut-être la classe  justify-content-end  : encore une fois, c’est une [classe utilitaire de Bootstrap 5](https://getbootstrap.com/docs/5.0/utilities/flex/) permettant d’ajouter la propriété  flex justify-content: flex-end.

En effet, la navigation Boostrap 5 est par défaut en display  flex, ce qui fait que nous pouvons utiliser les propriétés  flex  sur les éléments enfants de la  nav  pour les placer à droite, à gauche, au centre...

### Appréhendez le composant progress bar

Dans ce nouveau chapitre, vous allez découvrir un nouveau composant Bootstrap 5 : les **barres de progression** (progress bar, en anglais).

Comme à chaque fois que vous allez avoir envie d’intégrer un composant Bootstrap 5 dans votre site, rendez-vous d’abord dans la documentation Bootstrap du composant en question. Pour ce chapitre, vous allez chercher le [composant “Progress” dans le menu latéral “Components”](https://getbootstrap.com/docs/5.0/components/progress/).

Sur la documentation Bootstrap, on peut voir qu’il existe **plusieurs variantes** des barres de progression, libre à vous de les tester de votre côté, mais dans le cas de votre portfolio, vous allez utiliser la première option :

Les progress bars de Bootstrap

<div class="progress">

<div class="progress-bar" role="progressbar" aria-valuenow="0" aria-valuemin="0" aria-valuemax="100"></div>

</div>

<div class="progress">

<div class="progress-bar" role="progressbar" style="width: 25%" aria-valuenow="25" aria-valuemin="0" aria-valuemax="100"></div>

</div>

<div class="progress">

<div class="progress-bar" role="progressbar" style="width: 50%" aria-valuenow="50" aria-valuemin="0" aria-valuemax="100"></div>

</div>

<div class="progress">

<div class="progress-bar" role="progressbar" style="width: 75%" aria-valuenow="75" aria-valuemin="0" aria-valuemax="100"></div>

</div>

<div class="progress">

<div class="progress-bar" role="progressbar" style="width: 100%" aria-valuenow="100" aria-valuemin="0" aria-valuemax="100"></div>

</div>

Vous voyez que la structure est construite comme ceci :

* une div  avec la classe  progress (la barre grise) ;
* une div  enfant avec la classe  progress-bar  (la barre bleue), une  width  pour donner la valeur de la progression.

Maintenant que vous avez vu comment se construit la barre de progression de Bootstrap 5, vous pouvez les combiner avec des **classes utilitaires** pour les personnaliser un peu plus (par exemple, ajouter un  box-shadow, arrondir complètement les angles...).

### Découvrez le composant card

Les cards en anglais (“cartes” en français) sont un type d’élément d’**interface utilisateur** très populaire. Presque tous les frameworks CSS ont leur propre système de cards pour permettre aux développeurs de proposer un **affichage structuré** des informations de leur site.

Le composant card

Dans notre cas, nous allons les utiliser pour afficher nos projets à la suite de notre expertise.

Je ne fais que me répéter, mais commençons par aller voir la documentation Bootstrap 5 concernant les cartes. Rendez-vous dans [“components” puis “card”.](https://getbootstrap.com/docs/5.0/components/card/)

En parcourant rapidement la page, nous pouvons voir qu’il existe plusieurs types de cartes sur Bootstrap 5. La diversité des affichages permet de répondre à une multitude de **besoins** et de **cas d’usage spécifiques** (galerie photo, résumé de produits e-commerce, etc). Et comme d’habitude, il est toujours possible de pousser les composants encore plus loin grâce aux classes utilitaires.

Dans notre cas, nous allons utiliser l’affichage le plus populaire, c'est-à-dire le premier exemple de la doc Bootstrap : image, titre, texte, bouton.

<div class="card">

<img src="..." class="card-img-top" alt="...">

<div class="card-body">

<h5 class="card-title">Card title</h5>

<p class="card-text">Some quick example text to build on the card title and make up the bulk of the card's content.</p>

<a href="#" class="btn btn-primary">Go somewhere</a>

</div>

</div>

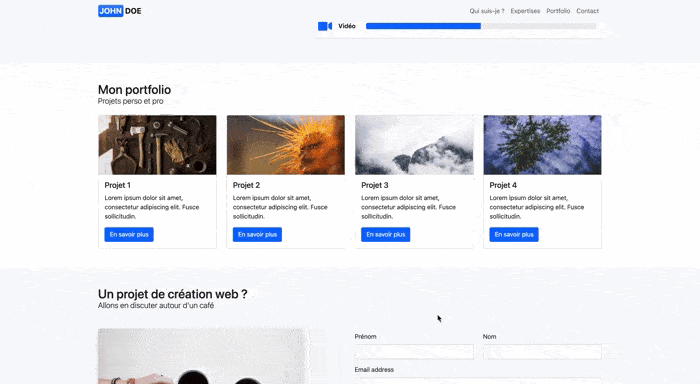
Nous pouvons voir que le composant suit une architecture donnée :

* Une div  globale avec une classe du nom du composant (l’attribut style de l’exemple Bootstrap n’est pas à garder).
* À l’intérieur de cette div, nous trouvons une  img  ainsi qu’une autre  div  avec la classe  card-body.
* Ce dernier div.card-body  indique l’emplacement du corps de la carte, dans lequel on retrouve le  card-title, le  card-text  et notre bouton.

### Affichez des informations complémentaires au clic sur un bouton grâce au Offcanvas

En fin de partie 3, nous avons vu comment afficher nos projets grâce au système de cartes de Bootstrap 5. Voyons maintenant comment le  offcanvas  nous permet de donner plus d'informations sur nos projets, sans venir alourdir notre page.

Le nom “Offcanvas” vient de canvas en anglais, qui veut dire “toile” (de peinture, pas d'araignée), et off, qui veut dire “désactivé”, car le contenu de   offcanvas  n’est visible qu’après une interaction de l’utilisateur.

Un exemple d’Offcanvas dans votre portfolio final

Comme d’habitude, rendez-vous sur la [documentation de Bootstrap 5](https://getbootstrap.com/docs/5.0/components/offcanvas/) pour voir comment fonctionne notre composant.

<div class="offcanvas offcanvas-start" tabindex="-1" id="offcanvas" aria-labelledby="offcanvasLabel">

<div class="offcanvas-header">

<h5 class="offcanvas-title" id="offcanvasLabel">Offcanvas</h5>

<button type="button" class="btn-close text-reset" data-bs-dismiss="offcanvas" aria-label="Close"></button>

</div>

<div class="offcanvas-body">

Content for the offcanvas goes here. You can place just about any Bootstrap component or custom elements here.

</div>

</div>

En observant la structure du composant, on remarque qu’il fonctionne un peu de la même manière que les  card  :

* Une div  globale avec une classe qui porte le nom du composant.
* Cette div porte aussi une classe  offcanvas-start  ; on se souvient que  -start  réfère à l’alignement dans le sens de lecture du site, ici de gauche à droite.
* On observe également un  id  qui nous permettra de cibler notre élément.
* Ensuite, nous avons deux  divsœurs, l’une portant la classe  .offcanvas-header  et l’autre  .offcanvas-body.

Pour notre cas précis, nous souhaitons mettre en place un  offcanvas  qui sorte du bas de page. En parcourant un peu la documentation, on voit qu’il existe bien une variante qui part d’en bas.

<button class="btn btn-primary" type="button" data-bs-toggle="offcanvas" data-bs-target="#offcanvasBottom" aria-controls="offcanvasBottom">Toggle bottom offcanvas</button>

<div class="offcanvas offcanvas-bottom" tabindex="-1" id="offcanvasBottom" aria-labelledby="offcanvasBottomLabel">

<div class="offcanvas-header">

<h5 class="offcanvas-title" id="offcanvasBottomLabel">Offcanvas bottom</h5>

<button type="button" class="btn-close text-reset" data-bs-dismiss="offcanvas" aria-label="Close"></button>

</div>

<div class="offcanvas-body small">

...

</div>

</div>

Dans l’exemple ci-dessus, nous retrouvons la structure que nous venons de détailler plus haut, mais nous voyons aussi l’architecture du bouton déclencheur. Et normalement, il y a des attributs qui doivent vous poser problème : les attributs  data.

### Revenons rapidement sur l’attribut  data

L’attribut data permet essentiellement l’échange de données entre l’HTML et le DOM qui, lui, est manipulable en JS.

Besoin d’un petit rappel sur le DOM ? Rendez-vous au chapitre “[**Comprenez ce qu'est le DOM**](https://openclassrooms.com/fr/courses/5543061-ecrivez-du-javascript-pour-le-web/5543068-comprenez-ce-quest-le-dom)” du cours Écrivez du JavaScript pour le Web.

Concrètement, l’attribut data peut être personnalisé comme on le souhaite suivant cette structure : data-{mon-text-perso} . Bootstrap 5 a choisi comme sémantique  data-bs-{utilité},  bs  pour Bootstrap, et l’utilité représente l’usage auquel il est lié en JS.

Dans notre exemple,  data-bs-toggle  permet au JS de Bootstrap d’identifier le bouton comme étant le déclencheur du composant  offcanvas  identifié par son  id  que l’on renseigne dans le deuxième attribut data  data-bs-target.

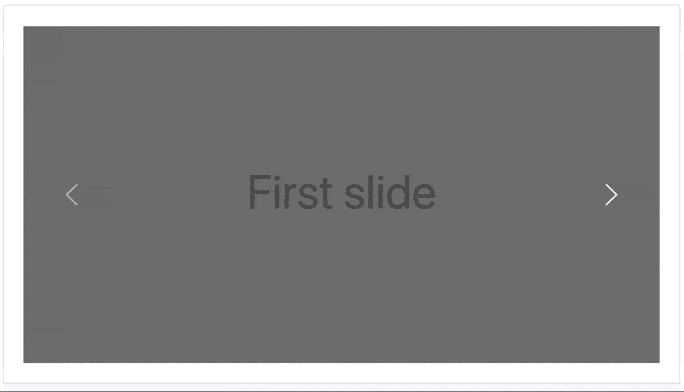
L’intérêt de ce système, c’est qu’il nous permet d’utiliser des composants qui fonctionnent grâce au JavaScript, en restant uniquement dans le HTML.

### Ajoutez un carrousel d’images pour illustrer vos projets

Ce que l’on appelle communément un "carrousel", c’est un slider d’images ou de vidéos qui défilent sur un intervalle donné. On le retrouve souvent dans les headers de pages pour mettre en avant des images accompagnées d’un message.

Il en existe plusieurs variantes, comme par exemple les “stories” sur Instagram, ou encore la navigation sur Netflix.

Celui de Bootstrap est le format le plus populaire, un encart fixe affichant les images les unes après les autres :

Un exemple du carrousel

Comme à chaque fois, commençons par nous rendre sur [la documentation Bootstrap 5](https://getbootstrap.com/docs/5.0/components/carousel/) pour voir comment fonctionne le carrousel : on clique sur “components” puis “carousel”.

La première chose que l’on voit, c’est que le carrousel fonctionne en CSS et JavaScript, mais Bootstrap 5 a fait en sorte qu’il soit utilisable **sans** écrire une seule ligne de JS !

La deuxième chose à savoir, c’est que le carrousel **ne normalise pas** la taille des images. Si on veut avoir un carrousel tout beau tout propre, il faudra soit ajouter un peu de CSS, soit travailler nos images en amont pour qu’elles aient la même taille (je préconise cette deuxième option qui a plus de sens d’un point de vue optimisation de votre site).

Attention, la traduction française du terme carousel est "carrousel", avec 2 "r". Veillez à bien utiliser **carousel** dans votre code !

Étudions un peu l’exemple de base du carrousel de Bootstrap 5 :

<div id="carouselExampleSlidesOnly" class="carousel slide" data-bs-ride="carousel">

<div class="carousel-inner">

<div class="carousel-item active">

<img src="..." class="d-block w-100" alt="...">

</div>

<div class="carousel-item">

<img src="..." class="d-block w-100" alt="...">

</div>

<div class="carousel-item">

<img src="..." class="d-block w-100" alt="...">

</div>

</div>

</div>

On observe plusieurs choses :

* Il y a une architecture d’éléments et de classes à respecter pour que ça fonctionne.
* Ils utilisent des classes utilitaires sur leurs images, c’est donc compatible ensemble (comme les barres de progression).
* Il y a un  id  sur l’élément parent principal.
* On retrouve également un attribut  data  pour faire le lien avec le JS.

Qu’est-ce que tout ça veut dire ?

Concernant l’architecture, ce n’est pas bien sorcier :

* unediv  parent qui prend les classes  .carousel .slide  ;
* une div  enfant avec la classe  .carousel-inner  (littéralement "intérieur du carrousel", qui sert à délimiter les slides) ;
* enfin, autant de  div  enfants avec la classe  carousel-item  que nous souhaitons de slides dans notre carousel.

L’  id  sert à identifier le carrousel pour lui apporter des modifications en JS, par exemple.

### Découvrez les formulaires

Les formulaires sont utilisés pour tout ce qui implique une saisie de données de la part des utilisateurs : inscription, connexion, recherche, modification d’informations. Pour permettre à vos visiteurs de vous contacter sur votre portfolio, vous allez à présent créer un prototype de formulaire de contact.

Le formulaire de base de Bootstrap

Familiarisons-nous avec la manière de créer des formulaires à l'aide de Bootstrap, afin que vous puissiez réaliser ce prototype.

Commençons par définir les champs dont nous avons besoin :

* un nom ;
* un prénom ;
* un e-mail ;
* un message ;
* un bouton “Envoyer”.

Comme à notre habitude maintenant, commençons par nous rendre sur la [documentation de Bootstrap 5,](https://getbootstrap.com/docs/5.0/forms/overview/) dans la rubrique “Forms”.

En cliquant sur “Overview”, nous pouvons voir la structure d’un formulaire classique.

<form>

<div class="mb-3">

<label for="exampleInputEmail1" class="form-label">Email address</label>

<input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp">

<div id="emailHelp" class="form-text">We'll never share your email with anyone else.</div>

</div>

<div class="mb-3">

<label for="exampleInputPassword1" class="form-label">Password</label>

<input type="password" class="form-control" id="exampleInputPassword1">

</div>

<div class="mb-3 form-check">

<input type="checkbox" class="form-check-input" id="exampleCheck1">

<label class="form-check-label" for="exampleCheck1">Check me out</label>

</div>

<button type="submit" class="btn btn-primary">Submit</button>

</form>

En laissant volontairement de côté la  checkbox  , nous observons deux éléments récurrents dans cet exemple : les classes  .form-control  et  .form-label. Ce sont les deux classes qui permettront à nos  input  d’être mis en page et d'accéder aux affichages alternatifs s’ils sont  disabled  ou en  readonly.

Une autre particularité de la structure du formulaire, c’est la div  qui englobe le champ (input + label). Dans l’exemple, on remarque également la classe utilitaire  .mb-3  (margin-bottom).

Pour récapituler, notre champ de formulaire :

* doit être entouré d’une div  ;
* son input doit avoir la classe  .form-control  ;
* son label doit avoir la classe  .form-label.

En respectant cette structure, nous nous assurons du bon affichage de notre champ.

En parcourant la documentation, nous pouvons voir qu’en fonction du type d’input (select, checkbox, range, etc.), la classe associée ainsi que la structure peuvent varier.

Nous pouvons résumer ça avec le tableau que vous trouverez dans ce cours :

|  |  |
| --- | --- |
| Classes | Input |
| input.form-control  label.form-label | text (et ses dérivés, email, password, etc.)  textarea |
| select.form-select | select |
| div.form-check  input.form-check-input  label.form-check-label | checkbox  radio |
| input.form-range  label.form-label | range |

Il existe d’autres variantes de mise en page de formulaire, ainsi que différents “états” (disabled,   readonly) que vous pouvez découvrir en parcourant la documentation de Bootstrap 5.

## Améliorez l’expérience utilisateur avec une modal et des tooltips



### Ajoutez les mentions légales avec le composant modal

L’affichage des mentions légales d’un site est une obligation en France. Ces mentions peuvent être plus ou moins longues et sur un portfolio, ce n’est pas forcément du contenu que l’on souhaite mettre en avant. Pour en savoir plus sur les problématiques juridiques liées au numérique, rendez-vous au cours [**Maîtrisez les risques juridiques liés au numérique**](https://openclassrooms.com/fr/courses/5162341-maitrisez-les-risques-juridiques-lies-au-numerique).

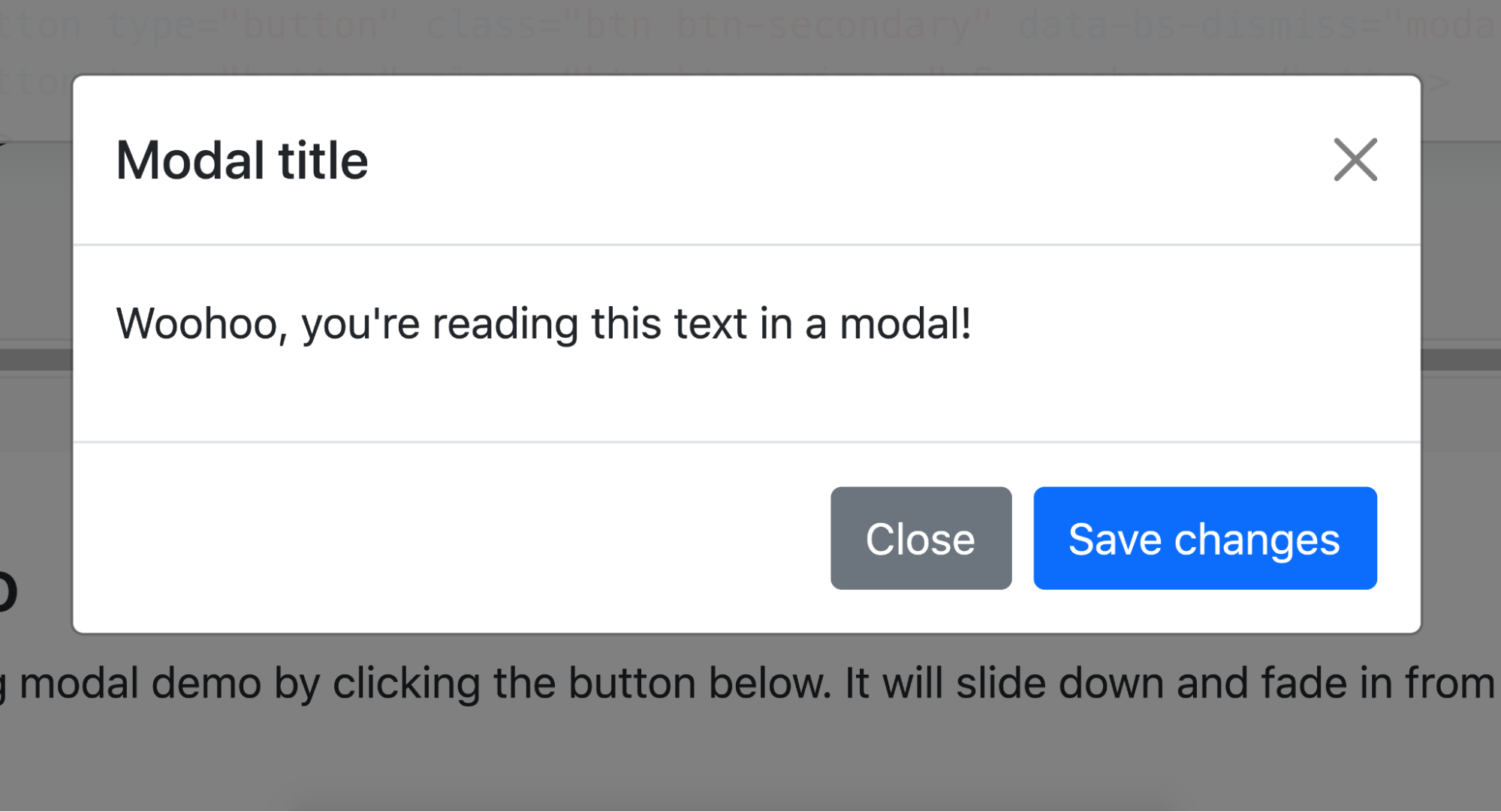
Dans le même genre que le composant  offcanvas, Bootstrap 5 met à notre disposition le composant  modal.

En quoi la  modal  est différente du  offcanvas  ?

En termes de fonctionnement, les deux composants sont similaires, c’est dans la forme qu’ils se différencient.

Concrètement, c’est une fenêtre “popup” qui s’affiche par-dessus la page affichée et qui est souvent utilisée dans le cadre de validation d’une action (ex. : “Êtes-vous sûr de vouloir supprimer XX ?”) ou pour afficher un contenu annexe (notre cas).

En nous rendant sur la [documentation Bootstrap 5](https://getbootstrap.com/docs/5.0/components/modal/) associée aux modals, nous allons pouvoir commencer par étudier la structure de ces dernières.

La démonstration d’une modal dans la documentation Bootstrap

<!-- Button trigger modal -->

<button type="button" class="btn btn-primary" data-bs-toggle="modal" data-bs-target="#exampleModal">

Launch demo modal

</button>

<!-- Modal -->

<div class="modal fade" id="exampleModal" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="exampleModalLabel">Modal title</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body">

...

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

<button type="button" class="btn btn-primary">Save changes</button>

</div>

</div>

</div>

</div>

Encore une fois ici, Bootstrap garde la même logique dans la structure de ses composants. On retrouve dans un premier temps le bouton sur lequel est liée l’activation de la modal. Les attributs  data-\*  servent à faire le lien avec le JS de Bootstrap.

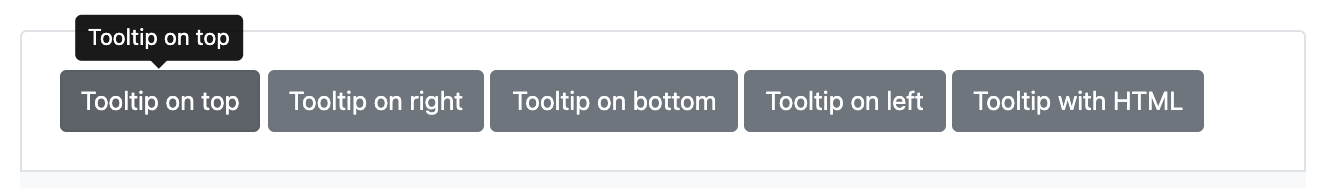
Ensuite, nous avons la modal en elle-même qui est composée de :

* Une  div  globale avec les classes  .modal  et  .fade  (“.fade” concerne l’animation de l’apparition) et un id  unique.
* Une autre  div  avec la classe  .modal-dialog  qui englobe les trois  div  qui vont structurer le contenu de la modal.
* Trois  div  sœurs (au même niveau hiérarchique) avec les classes  .modal-header,  .modal-body  et  .modal-footer.
* Enfin du contenu divers dont un élément avec la classe titre  .modal-title.

Dans notre cas, nous avons besoin d’une modal classique, avec une  div.modal-header  et une  div.modal-body. Vous noterez que le plus important pour utiliser les modals de Bootstrap 5, ce sont la  div.modal  et la  div.modal-dialog, car la structure interne est interchangeable au besoin.

### Rendez votre site plus accessible grâce aux “tooltips”

Un petit composant bien utile et que vous avez sûrement déjà vu sur des sites web, c’est le   tooltip.

Dans l'exemple, nous voyons le tooltip à gauche en haut du bouton

C’est ce petit texte qui s’affiche au survol de la souris, souvent pour apporter plus d’information à un élément qui en a besoin – une icône, par exemple.

Le tooltip a aussi une utilité en termes d’accessibilité et Search Engine Optimization (SEO), car il permet d’associer du texte à un élément qui n’en est pas (icône, image, etc.) !

Ce composant fonctionne avec la bibliothèque JS tiers “Popper JS” dont on a déjà parlé au tout début de ce cours. Ne vous inquiétez pas, cette bibliothèque est déjà ajoutée à Bootstrap, car nous avons choisi d’utiliser la solution “bundle”.

Le  tooltip  nécessite une initialisation en JS que l’on doit coder nous-même. Pas de panique, tout est expliqué dans [la documentation Bootstrap](https://getbootstrap.com/docs/5.0/components/tooltips/#example-enable-tooltips-everywhere) (comme d’habitude), et voici le code d’initialisation :

*var* tooltipTriggerList = [].slice.call(document.querySelectorAll('[data-bs-toggle="tooltip"]'))

*var* tooltipList = tooltipTriggerList.map(*function* (tooltipTriggerEl) {

return new bootstrap.Tooltip(tooltipTriggerEl)

})

L’objet du cours n’étant pas de comprendre le JS, nous allons nous concentrer sur les points liés au composant.

L’idée générale du script, c’est de récupérer dans la page HTML (le DOM) tous les éléments qui ont un attribut  [data-bs-toggle=”tooltip”]. Une fois ces éléments récupérés, le script va juste lier ces derniers avec le reste du JS fourni par Bootstrap 5.

Ce bout de code nous permet donc d’utiliser les tooltips dans toutes les pages où il est présent.

Voici à quoi va ressembler une icône avec un tooltip :

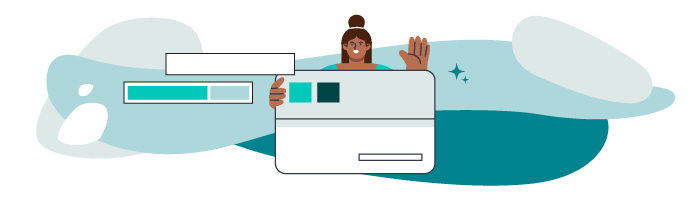
<a href="#" class="text-decoration-none text-dark" data-bs-toggle="tooltip"

title="LinkedIn">

<i class="fab fa-linkedin fa-2x"></i>

</a>

## Créez votre propre thème de conception



Vous avez fait beaucoup de progrès au cours des précédentes parties de ce cours. Vous avez pu créer un prototype fonctionnel du site web de portfolio, en utilisant uniquement les classes de Bootstrap, et sans écrire ni modifier une seule ligne de CSS !

Aussi intéressants que soient le thème par défaut et les exemples mis à disposition par Bootstrap pour créer rapidement et efficacement un site web, il est essentiel de personnaliser le thème aux couleurs de votre projet afin de le rendre unique. Bon nombre de ressources en ligne existent à ce sujet. Entre autres, il est possible d’acquérir des [thèmes Bootstrap](https://themes.getbootstrap.com/) sur le site web officiel, qui vous permettront d’apprécier le niveau de détail qu’offre le framework en termes de personnalisation.

### Configurez votre projet pour pouvoir le personnaliser

Avant de pouvoir créer votre propre thème, vous devez vous assurer que votre projet soit correctement configuré. Jusqu'à présent, vous avez utilisé les fichiers compilés de [BootstrapCDN](https://www.bootstrapcdn.com/" \t "_blank). La seule manière de remplacer le fichier compilé est de créer des classes Bootstrap de substitution, mais cela prendrait du temps et serait potentiellement source d’erreurs.

À présent, nous utiliserons **Sass** pour la génération de variables.

Vous pouvez consulter le cours [**Simplifiez-vous le CSS avec Sass**](https://openclassrooms.com/fr/courses/6106181-simplifiez-vous-le-css-avec-sass) sur OpenClassrooms en guise de révision, ou pour en savoir plus sur Sass.

Pour utiliser Sass avec Bootstrap 5, ce dernier doit être installé dans votre projet en utilisant **npm**.

Si vous ne l'avez pas déjà sur votre ordinateur, veillez à l'avoir avant de passer à l'étape suivante (vous pouvez suivre la documentation de npm, [**Instructions d'installation de npm**](https://www.npmjs.com/get-npm)).

OK, maintenant que vous avez les outils appropriés, il est temps de commencer la personnalisation !

Ouvrez un terminal de ligne de commande, placez-vous à la racine du répertoire qui contient les fichiers de votre projet, puis exécutez la commande suivante :

$ npm init

Besoin de vous rafraîchir la mémoire sur l'utilisation du terminal ? Vous pouvez consulter le cours "[**Apprenez à utiliser la ligne de commande dans un terminal**](https://openclassrooms.com/fr/courses/6173491-apprenez-a-utiliser-la-ligne-de-commande-dans-un-terminal)".

Vous pouvez accepter les valeurs par défaut ou ajouter vos propres réponses aux suggestions. Une fois la commande terminée, vous aurez un nouveau fichier appelé **package.json** dans la racine de votre dossier de projet.

Par la suite, utilisez **npm** pour télécharger la dernière version de Bootstrap dans votre projet.

Exécutez la commande suivante depuis le terminal :

$ npm install bootstrap

Une fois exécutée, cette commande créera un répertoire nommé **node\_modules**, qui contiendra un sous-répertoire nommé **bootstrap**. Ce répertoire contient les fichiers Bootstrap compilés dans le répertoire **dist**, ainsi que les fichiers sources de Bootstrap (non compilés) dans le répertoire **scss**.

Maintenant, créez deux répertoires à la racine de votre projet : l'un nommé **scss** dans lequel iront vos fichiers Sass, et l’autre nommé **css** dans lequel iront les fichiers CSS.

Dans votre répertoire **scss**, créez un fichier nommé **style.scss**.

Étant donné que vous allez remplacer les variables de Bootstrap, vous devez importer le ou les fichiers pertinents du répertoire **node\_modules/bootstrap/scss** dans votre projet.

Pour ce faire, dans le fichier **style.scss**, ajoutez la ligne :

@import "../node\_modules/bootstrap/scss/bootstrap.scss"

Cela inclura la totalité de Bootstrap dans votre projet.

Bien entendu, vous avez la possibilité d’optimiser la taille de votre fichier CSS généré, en n’important que les fichiers nécessaires à votre personnalisation. Cependant, afin de simplifier notre démarche, vous importerez la totalité de Bootstrap pour ce cours.

Pour vérifier si Sass est déjà installé sur votre machine, vous pouvez exécuter la commande   $ sass --version    sur votre terminal. Si votre terminal vous renvoie une erreur, cela veut dire que Sass n’est pas installé.

Pour générer le fichier CSS que votre projet utilisera, nous devons tout d’abord installer Sass (si ce n’est pas déjà fait) en exécutant la commande suivante :

$ npm install -g sass

Une fois Sass installé, vous devriez pouvoir lancer la compilation automatique de votre fichier SCSS grâce à la commande suivante :

$ sass --watch scss:css

Si vous rencontrez des problèmes pour installer Sass sur votre machine, vous pouvez suivre le cours d’OpenClassrooms [**Installer Sass sur votre machine**](https://openclassrooms.com/fr/courses/6106181-simplifiez-vous-le-css-avec-sass/6599386-installez-sass-sur-votre-machine).

Cela générera un fichier appelé **style.css** dans le répertoire **css**. L'option  **--watch**permet de garder la commande en cours d'exécution, et de recompiler le ou les fichiers CSS dans le répertoire **css**chaque fois que des modifications sont apportées au contenu du répertoire **scss**. C'est très utile lorsque vous travaillez sur vos fichiers.

L'étape finale consiste à remplacer la référence à la version de CDN de Bootstrap dans l'en-tête de vos fichiers HTML, par une référence au fichier style.css généré.

Vous devez donc remplacer cette ligne dans la section  <head>  de vos fichiers HTML :

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css" integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh" crossorigin="anonymous">

par cette ligne :

<link rel="stylesheet" href="css/style.css" />

Testez vos fichiers en les ouvrant dans un navigateur. Ils doivent toujours afficher correctement les composants de Bootstrap.

À l’heure où j’écris ces lignes, une erreur est présente dans la version Sass de Bootstrap 5. Sur le composant   offcanvas, les   z-index  de la classe   .offcanvas  et celui de la classe   .modal-backdrop  sont **inversés**, ce qui cache le contenu de votre carrousel derrière un voile noir en   opacity: 0.5 .

Pour y remédier, cherchez le fichier **\_offcanvas.scss** dans votre dossier Bootstrap, puis récupérez le nom de la variable du   z-index  de la classe   .offcanvas. Si ça ne change pas, cette variable devrait être   $zindex-offcanvas. Ajoutez-la dans votre fichier **\_variables\_overrides.scss** et donnez-lui une valeur supérieure au   z-index  de la classe   .modal-backdrop, soit   1051  si ça n’a pas changé au moment où vous lisez ces lignes.

Et voilà, vous avez corrigé un bug en seulement une ligne de code ! Félicitations !

### Personnalisez vos pages

Il est temps maintenant de commencer la personnalisation de votre site web !

Bootstrap 5 utilise des variables et des cartes Sass pour définir de nombreux paramètres et composants du site.

Ouvrez et examinez le fichier **node\_modules/bootstrap/scss/\_variables.scss**. Il s’agit de l'emplacement où, par exemple, les variables des couleurs utilisées pour certaines classes que vous avez rencontrées dans ce cours sont définies (par exemple, les couleurs des classes   .bg- et   button-\*).

Il vous est possible de modifier n'importe quelle valeur par défaut de Bootstrap en remplaçant les variables qui ont été définies dans ce fichier. Comme vous le remarquez, elles sont définies à l'aide de l'[indicateur](https://sass-lang.com/documentation/variables" \l "default-values" \t "_blank)**[!default](https://sass-lang.com/documentation/variables" \l "default-values" \t "_blank)**[Saas](https://sass-lang.com/documentation/variables" \l "default-values" \t "_blank).

Pour les remplacer, elles doivent être définies avant l'importation de la bibliothèque et sans l’indicateur**!default**.

Pour ce faire, créez un fichier appelé **\_variables\_overrides.scss** dans le dossier **scss** de votre projet, et importez-le. Une fois le fichier créé, il vous faut définir son importation tout en haut de **style.scss**.

Le contenu du fichier **style.scss** sera désormais le suivant :

@import "variables\_overrides.scss"; // <-- ajouter cette ligne tout en haut du fichier

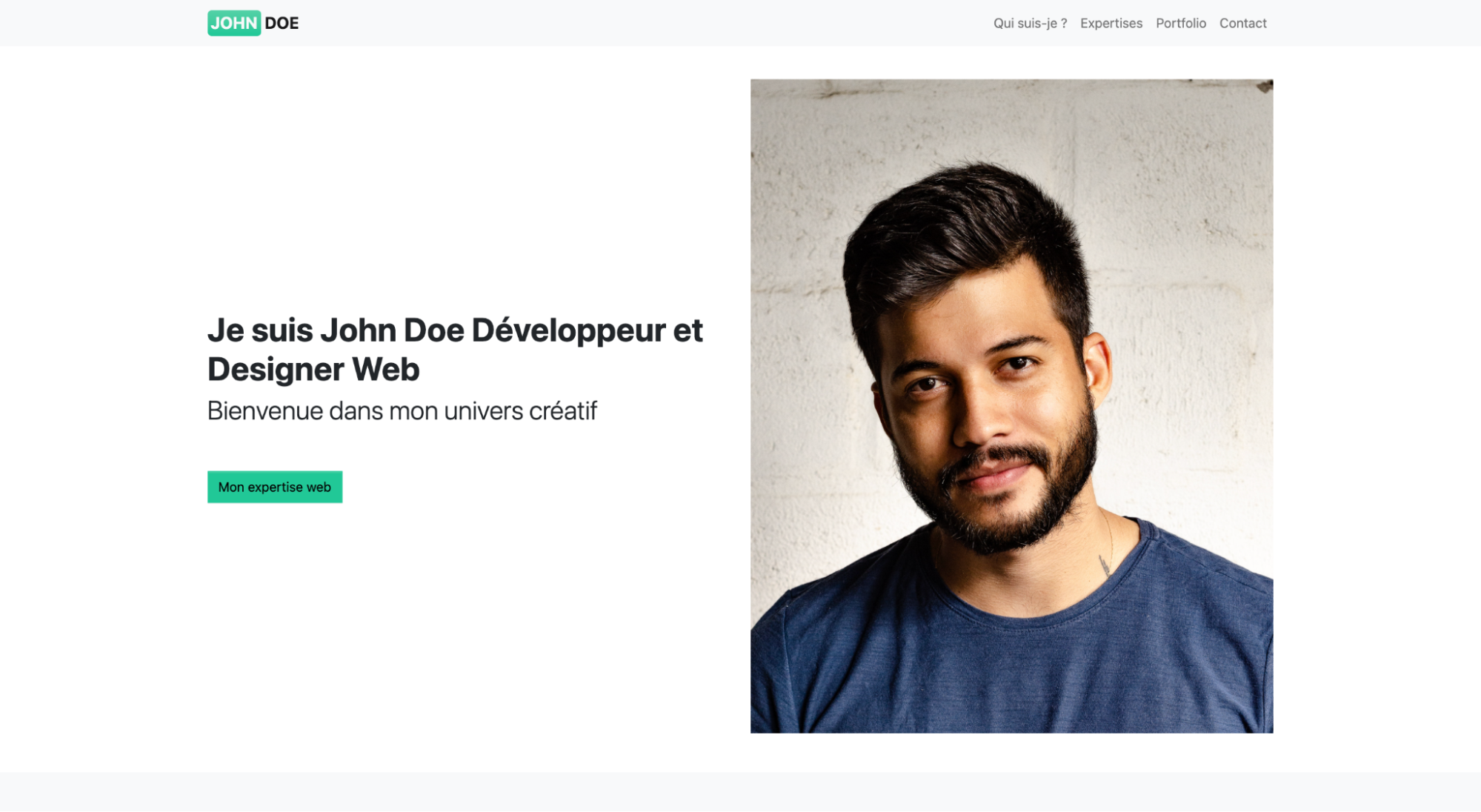
@import "../node\_modules/bootstrap/scss/bootstrap.scss";

Maintenant que vous avez mis en place l’architecture Sass de votre projet, vous êtes en mesure de modifier les variables par défaut de Bootstrap 5 pour personnaliser votre thème.

Pour ce faire, ouvrez dans votre IDE la page **node\_moduls/bootstrap/scss/\_variables.scss**, puis copiez-collez les variables que vous souhaitez modifier dans votre fichier **\_variables\_overrides.scss**.

Si vous voulez changer la couleur “primary” de toutes les classes qui l’utilisent, il vous suffit de copier-coller la variable Sass “$primary” par la valeur de la nouvelle couleur voulue dans **\_variables\_overrides.scss**.

$primary: #20c997;

Les éléments qui utilisent la couleur “primary” ont changé de couleur

## Créez vos propres classes utilitaires grâce à l’API de Bootstrap 5



Dans sa dernière mise à jour, Bootstrap 5 a ajouté une API en Sass permettant de générer des classes utilitaires en quelques lignes de code.

Le but de ce chapitre est de vous expliquer comment elle fonctionne et comment la mettre en pratique dans vos futurs projets.

Rendez-vous dans la [documentation Bootstrap 5,](https://getbootstrap.com/docs/5.0/utilities/api/) dans la rubrique “Utility” puis “API”. On retrouve le code suivant :

$utilities: (

"opacity": (

property: *opacity*,

values: (

0: 0,

25: .25,

50: .5,

75: .75,

100: 1,

)

)

);

Qui a pour résultat :

.opacity-0 { *opacity*: 0; }

.opacity-25 { *opacity*: .25; }

.opacity-50 { *opacity*: .5; }

.opacity-75 { *opacity*: .75; }

.opacity-100 { *opacity*: 1; }

Dans l’exemple ci-dessus, nous voyons la structure Sass de l’API.

Quand on voit cet exemple, on peut légitimement se demander à quoi elle sert, car il y a presque autant d’un côté que de l’autre !

Tout l'intérêt de cette API se dévoile quand on ajoute certaines **options** dans le code Sass (partie 1 de l’exemple ci-dessus).

Voici les options utilisables, dans une traduction du tableau de la documentation de Bootstrap 5 :

|  |  |  |
| --- | --- | --- |
| **Option** | **Type** | **Description** |
| property | Obligatoire | Nom de la propriété, il peut s'agir d'une chaîne ou d'un tableau de chaînes (par exemple, des marges ou des marges horizontales). |
| values | Obligatoire | Liste de valeurs, ou un tableau de valeurs si vous ne voulez pas que le nom de la classe soit le même que la valeur. Si   null  est utilisé comme clé de tableau, il n'est pas compilé. |
| class | Optionnel | Variable pour le nom de la classe si vous ne voulez pas qu'il soit le même que la propriété. Si vous ne fournissez pas la   class  clé et que la   property  clé est un tableau de chaînes de caractères, le nom de la classe sera le premier élément du   property  tableau. |
| state | Optionnel | Liste des variantes de pseudo-classe comme   :hover  ou   :focus  à générer pour l'utilitaire. Aucune valeur par défaut. |
| responsive | Optionnel | Booléen indiquant si des classes responsives doivent être générées.   false  par défaut. |
| rfs | Optionnel | Booléen pour activer le redimensionnement fluide. Jetez un œil à la page [RFS](https://getbootstrap.com/docs/5.0/getting-started/rfs/) pour savoir comment cela fonctionne.   false  par défaut. |
| print | Optionnel | Booléen indiquant si les classes d'impression doivent être générées.   false  par défaut. |
| rtl | Optionnel | Booléen indiquant si l'utilitaire doit être conservé en RTL.   true  par défaut. |

Voici l’exemple d’une classe utilitaire générée par l’API de Bootstrap 5  avec l’option   responsive  égale à true  :

$utilities: (

"opacity": (

property: *opacity*,

responsive: true,

values: (

0: 0,

25: .25,

50: .5,

75: .75,

100: 1,

)

)

);

qui génère :

.opacity-0 { *opacity*: 0 !important; }

.opacity-25 { *opacity*: .25 !important; }

.opacity-50 { *opacity*: .5 !important; }

.opacity-75 { *opacity*: .75 !important; }

.opacity-100 { *opacity*: 1 !important; }

@media (*min-width*: 576px) {

.opacity-sm-0 { *opacity*: 0 !important; }

.opacity-sm-25 { *opacity*: .25 !important; }

.opacity-sm-50 { *opacity*: .5 !important; }

.opacity-sm-75 { *opacity*: .75 !important; }

.opacity-sm-100 { *opacity*: 1 !important; }

}

@media (*min-width*: 768px) {

.opacity-md-0 { *opacity*: 0 !important; }

.opacity-md-25 { *opacity*: .25 !important; }

.opacity-md-50 { *opacity*: .5 !important; }

.opacity-md-75 { *opacity*: .75 !important; }

.opacity-md-100 { *opacity*: 1 !important; }

}

@media (*min-width*: 992px) {

.opacity-lg-0 { *opacity*: 0 !important; }

.opacity-lg-25 { *opacity*: .25 !important; }

.opacity-lg-50 { *opacity*: .5 !important; }

.opacity-lg-75 { *opacity*: .75 !important; }

.opacity-lg-100 { *opacity*: 1 !important; }

}

@media (*min-width*: 1200px) {

.opacity-xl-0 { *opacity*: 0 !important; }

.opacity-xl-25 { *opacity*: .25 !important; }

.opacity-xl-50 { *opacity*: .5 !important; }

.opacity-xl-75 { *opacity*: .75 !important; }

.opacity-xl-100 { *opacity*: 1 !important; }

}

@media (*min-width*: 1400px) {

.opacity-xxl-0 { *opacity*: 0 !important; }

.opacity-xxl-25 { *opacity*: .25 !important; }

.opacity-xxl-50 { *opacity*: .5 !important; }

.opacity-xxl-75 { *opacity*: .75 !important; }

.opacity-xxl-100 { *opacity*: 1 !important; }

}

Avec cet exemple, nous voyons plus concrètement la puissance de l’API utilitaire de Bootstrap 5, et le gain de temps qu'elle nous permet d’avoir sur notre développement.

En effet, elle nous permet de générer automatiquement 30 classes CSS liées aux “media queries” de chacun des breakpoints (points d’arrêt) de Bootstrap 5. C’est un gain de temps indéniable pour la création de vos propres classes utilitaires.

### Implémentez l’API utilitaire dans votre projet

Pour utiliser l’API dans notre projet, nous allons créer une nouvelle page sous **sass/\_utilities\_custom.scss**, que nous allons importer juste après nos  **sass/\_variables\_overrides.scss**et avant  **bootstrap.scss**de notre fichier **style.scss**.

Tout en haut de ce nouveau fichier, il faut commencer par déclarer la variable  **$utilities**à vide par défaut.

Pour ce faire, il suffit d’écrire   **$utilities: ()!default;**.

Une fois la page créée, il nous faut utiliser la fonction  **map-merge**de Sass pour ne pas remplacer les classes déjà existantes de Bootstrap 5. Voyons comment nous y prendre :

$utilities: map-merge(

$utilities,

(

"cursor": (

property: *cursor*,

class: *cursor*,

responsive: true,

values: auto pointer grab,

)

)

);

Nous voyons que la structure est sensiblement la même, à la différence près que tout se passe dans la fonction **map-merge**.

Concrètement, nous disons à notre code de fusionner le contenu de la variable  **$utilities**déjà existante (premier argument de la fonction **map-merge**), avec le reste du code que nous ajoutons (deuxième argument de la fonction **map-merge**).

Et bien sûr, pour que le CSS soit généré, il ne faut pas oublier de compiler notre Sass avec la commande (si elle n’est pas déjà en cours d'exécution) :

$ sass --watch scss:css