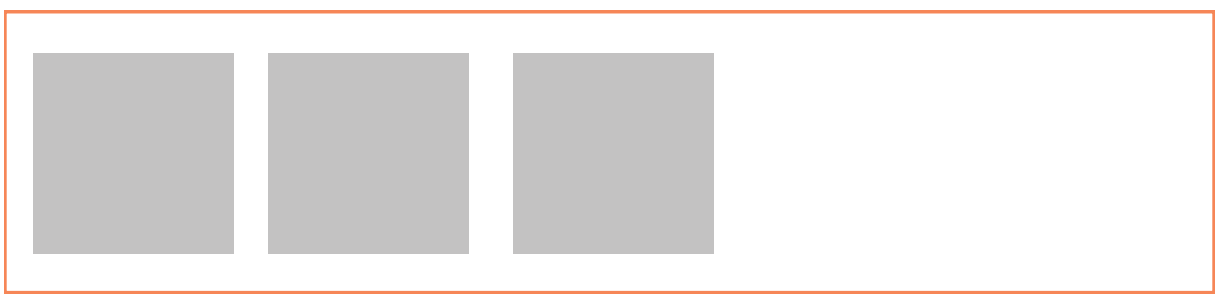
Core Module

row

La disposition se fait en rangées et en colonnes. On a de nouvelles façons de disposer nos éléments.

De base, dans une rangée, les éléments à l'intérieur sont alignés sur une ligne et les éléments se mettent tous en haut à gauche. De plus, la rangée a toujours 100% de la largeur de la page.

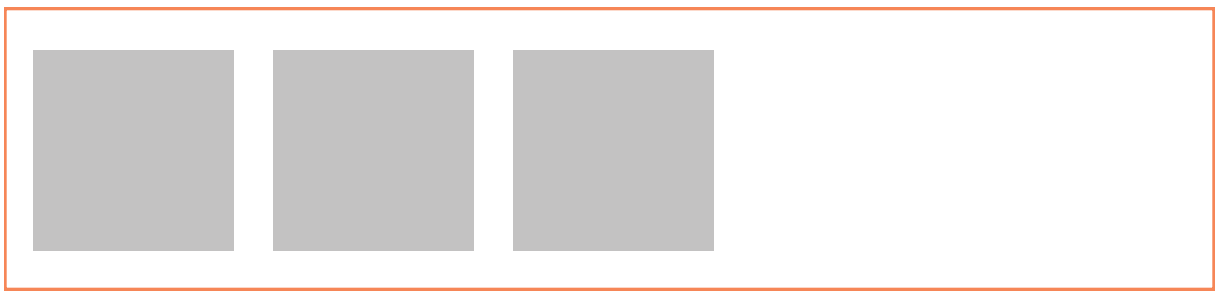
.row



```
<div class="row">…</div>
```

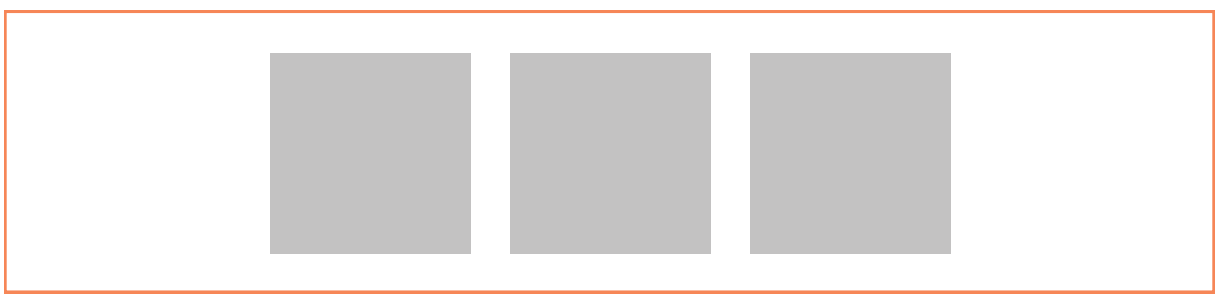
Cependant, selon les cas ont peut disposer les éléments au centre, à droite ou séparés entre eux.

.row.item-left



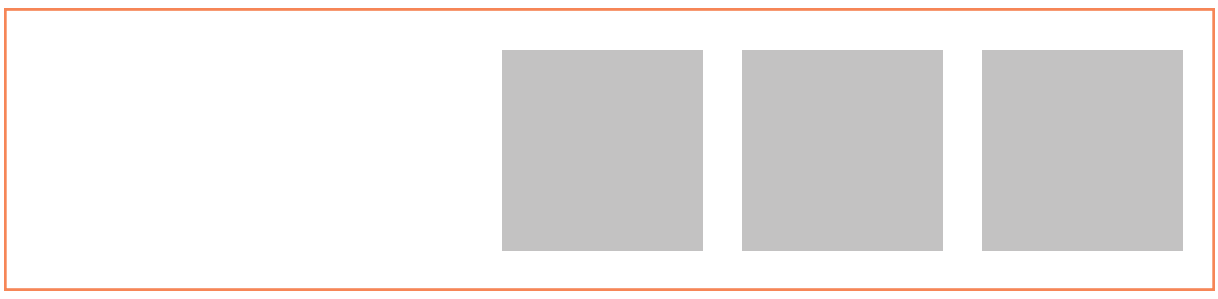
```
<div class="row item-left">…</div>
```

.row.item-center



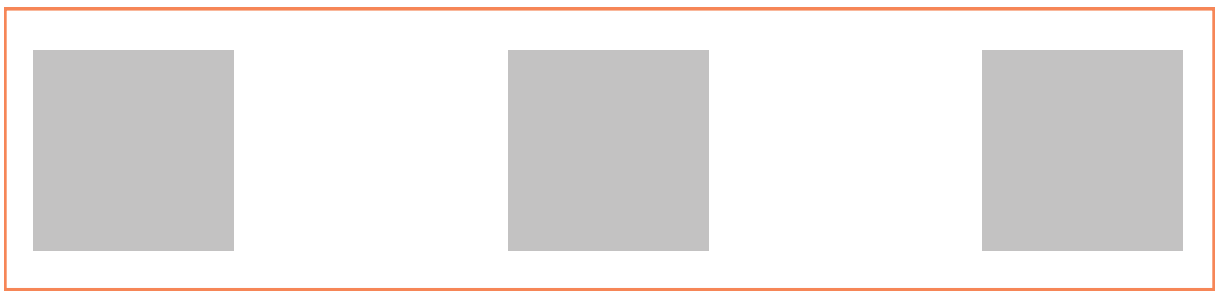
```
<div class="row item-center">…</div>
```

.row.item-right



```
<div class="row item-right">…</div>
```

.row.item-split



```
<div class="row item-split">…</div>
```

col

Les colonnes ont aussi évoluées. On a beacoup de possibilités.De base, elles utilisent toute la rangée à part égale.

.col

```
<div class="row">
    <div class="col">...</div>
    <div class="col">...</div>
    <div class="col">...</div>
</div>
```

On peut maintenant leurs donner une grandeur prédéterminée de 20% à 100% par tranche de 20. Aussi il y a full qui prend toute la largeur disponible sans auncune marge

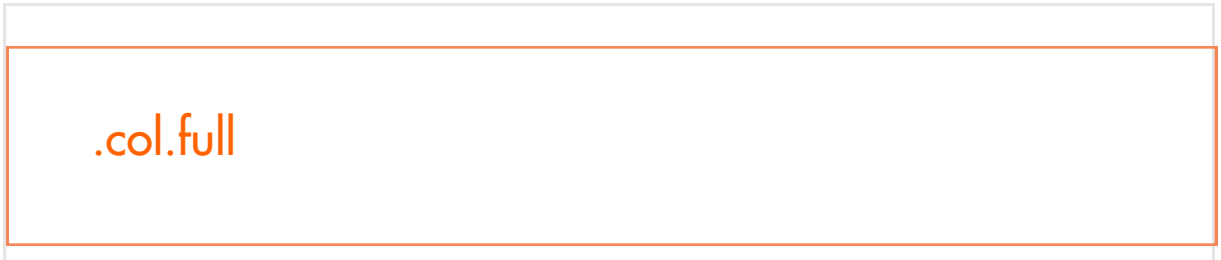
.xsmall -> 20%
.small -> 40%
.medium -> 60%
.large -> 80%
.xlarge -> 100%
.full -> 100% (sans marge)

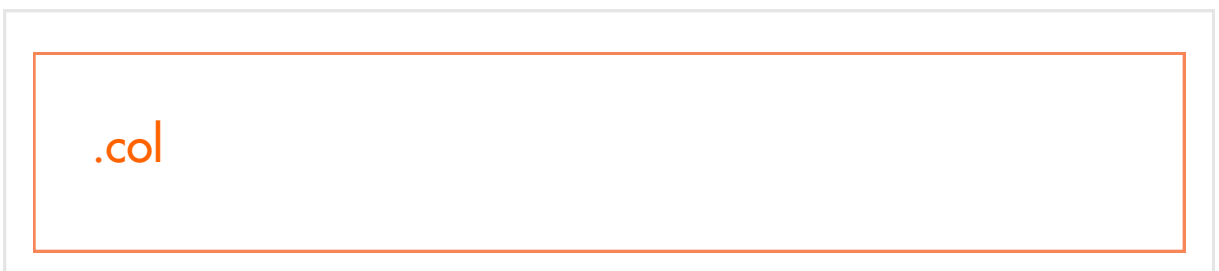
Le premier élément prend l'espace déterminé. Les autres s'ajustent avec ce qui reste. Ça ne peut pas dépasser 100%.


```
<div class="row">
    <div class="col large">...</div>
    <div class="col large">...</div>
</div>
```

Mais le total peut être plus petit que 100%.

Il est suggéré de finir la colone avec un .col (sans size). Ce dernier va remplir l'espace restant.



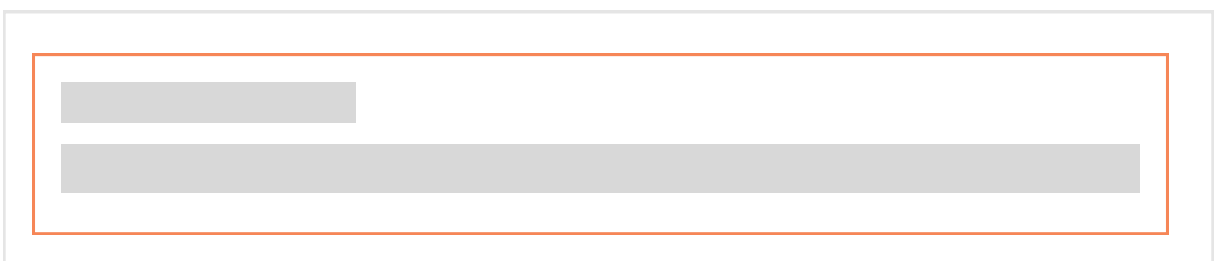


.wrap .nowarp

On peut maintenant choisir la disposition des éléments dans la colone. Par default, les éléments, labels et inputs par exemple, vont s'aligner les uns par dessus les autres.

Pratiquement, en UX, on suggère de mettre les labels au dessus des inputs pour être plus rapide à remplir par les utilisateurs.

.col.size.wrap



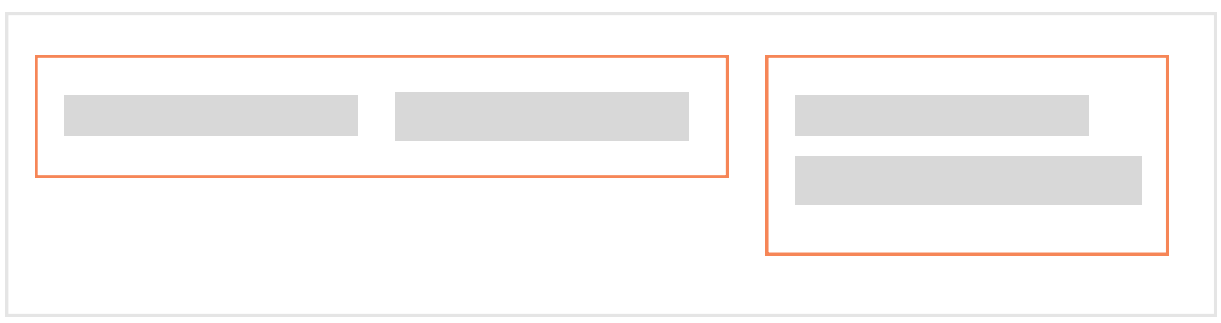
```
<div class="row">
    <div class="col xlarge wrap">
        ...
    </div>
</div>
```

Si on met nowrap les élément ce centre sur une même ligne. Selon les attributs des éléments le sizing devrait s'ajuster.

.col.size.nowrap

```
<div class="row">
    <div class="col xlarge nowrap">
        ...
    </div>
</div>
```

On peut mettre une colone en wrap et l'autre nowrap mais c'est vraiment pas à prélivégier. En mobile, tout wrap.



```
<div class="row">
    <div class="col nowrap">     </div>
    <div class="col wrap">   </div>
</div>
```

Titre

La grosseur des titres et des textes est déjà prédéterminée. On a de h1 à h6 comme titre. La typo est au choix du client mais de base on a une typo sans serif. Les class mobiles sont prédéterminées.

La grosseur des autres éléments de texte est aussi pédéterminée selon l'élément. De base, le texte dans un p serait de 1em soit environ16px.

h1
39.063 px

h2
31.25 px

h3
25 px

h4
20 px

h5
16 px

h6
12,8 px

h1
31.25 px

h2
25 px

h3
20 px

h4
18 px

h5
16 px

h6
12,8 px

Media Querie

Présentement on a pas de hack spécial pour les différents browser. Tout est gérer par Autoprefixer.

Il gère ie11 et plus et les 2 dernieres versions des navigateurs plus modernes.

Les classes sont déjà configurées pour les différentes tailles d'écran.

Voici les breakpoints:

640px	iphone lanscape
768px	Tablet portrait
1024px	tablet lanscape et desktop
1920px	big desktop

Certains media queries sont utilsés pour les image haute résolution, mais des plugins Gulp et des mixins règlent ces problèmes.

Couleur

On se limite a 5 couleurs bases pour nos styles.
On utilise les noms:

black
white
accent
primary
secondary

.bgcolor-primary
.textcolor-black

Et 3 couleurs pour les « validations »

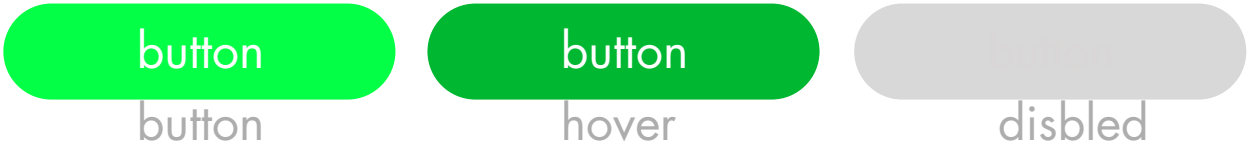
success warning alert

Il y a aussi des couleur pour les « brands » suivante:

twitter
googleplus
linkedin
vimeo
flickr
foursquare
facebook
pinterest
youtube
instagram
dribbble

Button

Que se soit un input ou un bouton avec un type submit les boutons ont 3 états. Leur taille est gérée par les données à l'intérieur du button.




```
<button type="submit">button</button>
— ou —
<input type="submit" value="button" />
```


```
<button type="submit" class="bg-color-secondary">button</button>
— ou —
<input type="submit" class="bg-color-secondary" value="button" />
```

Layout component

.accordion

Accordion cache les colonnes ou les sections sous lui. Il peut se placer dans une rangé ou une colonne. Il va prendre 100% de la largeur disponible. Par défault l'accordeon n'a pas d'icon. Le titre doit absolument etre un h2.

.accordion

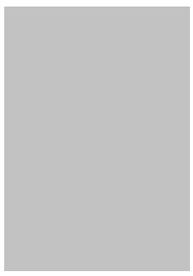
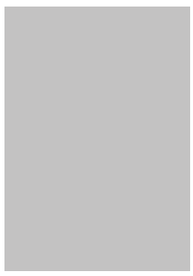
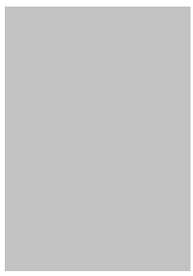
Titre

```
<div class="row accordion">
    <h2>titre</h2>
    <div class="row">
        ...
    </div>
</div>
```

en ajoutant .open les éléments apparaissent. Il faut créer un javascript qui ajoute la classe open.

.accordion .open

Titre

```
<div class="row accordion open">
    <h2>titre</h2>
    <div class="row ">
        ...
    </div>
</div>
```

si on utilise accordion-icon au lieu un icone se place à la droite du titre.

.accordion-icon

Titre

Hero image mixin

Le mixin hero image permet de gérér les images responsives dans les «heros» comme cover image. Comme dans le site d'Analystik ou Signder.

Cependant, vous devez aller dans modules/component/layout/hero-image et changer le fichier _element.scss

Vous devez simplement créer la nouvelle class avec le mixin

```
.nom_de_la_class{
    @cover_image('image_path',extention,cover )
}
```

nom_de_la_classe sera à votre discretion
image_path sera le chemin pour l'image par exemple
'/Content/images/nom_de_image'
extention sera le type d'extention (jpg, png ou gif)
cover est le type ou size peut être cover ou contain.

Par la suite, vous utilisez votre nouvelle classe dans un div.

Sticky header

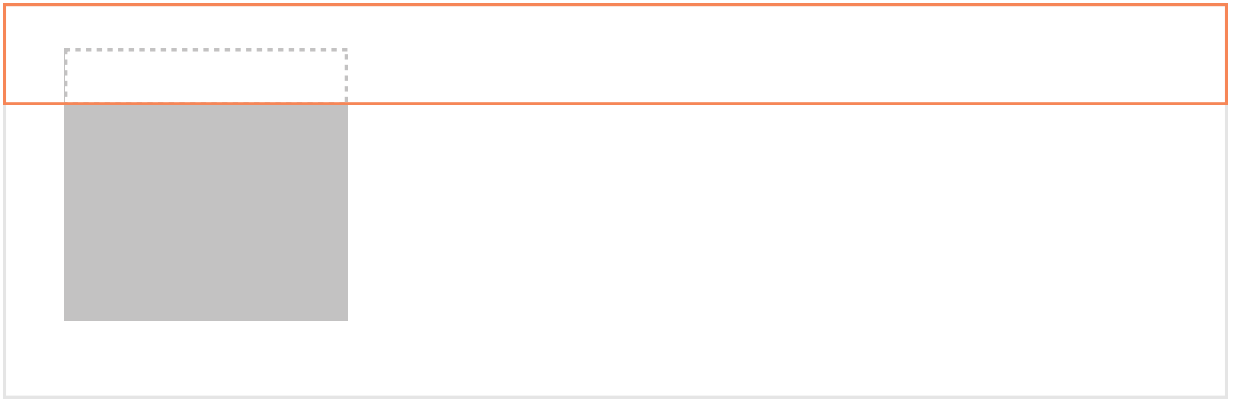
Le sticky header permet d'avoir un menu toujours accessible peu importe où on se trouve dans la page.

.sticky

```
<div class="row sticky">
    ...
</div>
```

Le menu s'accroche à une rangée. De base, le menu n'est pas visible. En ajoutant la class visible, à l'aide d'un javascript (positionnement de la fenêtre), le menu se fixe au haut de la page. Le menu est toujours au dessus des éléments de la page.

```
<div class="row sticky visible">
    ...
</div>
```



Element component

Blockquote

Tout ce qui est du blockquote est déjà prêt. L,élément de la citation et l'auteur; la couleur du texte, la grosseur et le style.

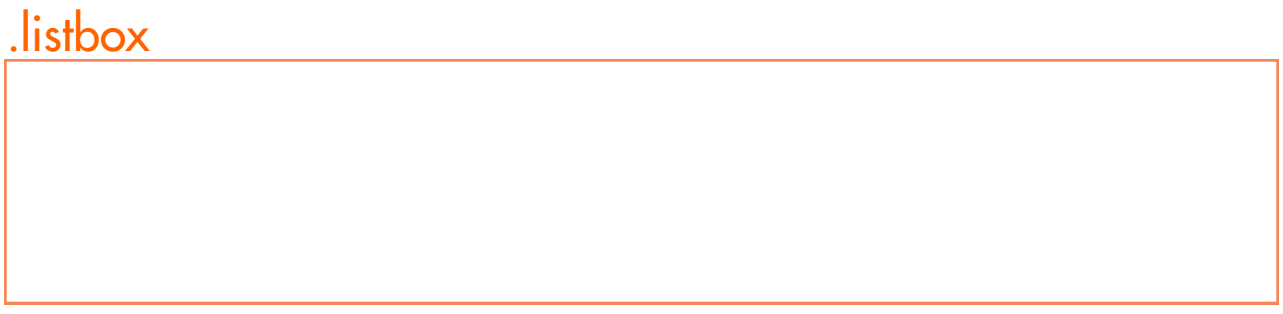
```
<blockquote>
    Less is more
    <cite>Ludwig Mies Van Der Rohe</cite>
</blockquote>
```

> Less is more
>
> *- Ludwig Mies Van Der Rohe*

Liste

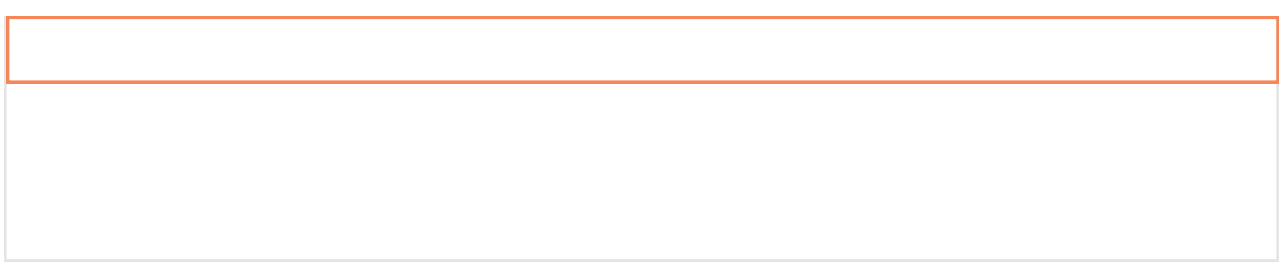
On découpe tous les éléments. Les cellules sont des div. independants.

.listbox



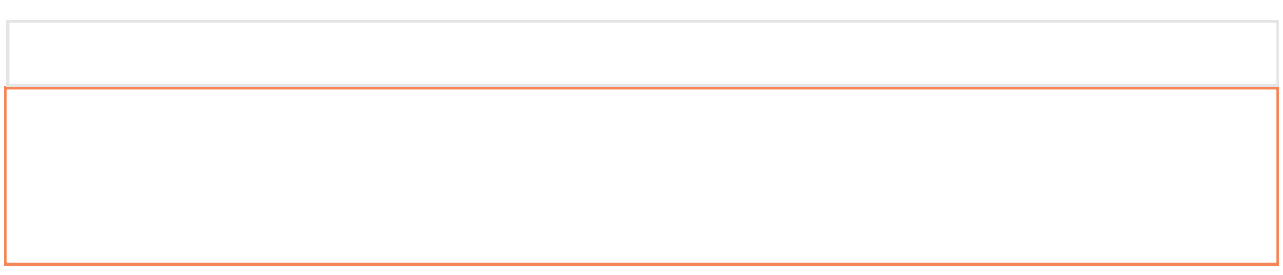
L'élement listbox découpe seulement le cadre de la liste.

.head



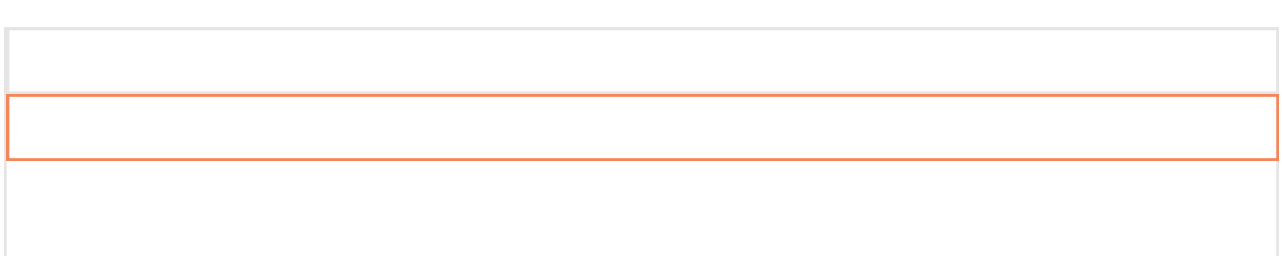
Le head est une rangée spéciale qui crée les cellules d'entêtes.

.listbody



Le listbody est le groupe de rangées de la liste sans le header.

a.row



Créer les autres rangées de la liste.

.col.size

Seulement les col.size sont utilisées dans les listes.

```
<section>
    <div class="listbox">
        <div class="head">
            <div class="col xsmall">Id</div>
            <div class="col small">Name</div>
            <div class="col medium">Note</div>
        </div>
        <div class="listbody hsmall">
        <a class="row">
            <div class="col xsmall left">6</div>
            <div class="col small">John Doe</div>
            <div class="col medium">Client modèle</div>
        </a>
        <div>
    </div>
</section>
```

Les sizes des colonnes du head doivent se répéter dans les autres colonnes des rangées au dessous.

Aux colonnes peuvent aussi être ajoutées la classe left ou right pour aligner les items à gauche où à droite.

Le listbody peut avoir des size de hauteur qui équivaut à un nombre de ligne

.hxsmall -> 4 lignes de haut

.hsmall -> 6 lignes de haut

.hmedium -> 10 lignes de haut

.hlarge -> 15 lignes de haut

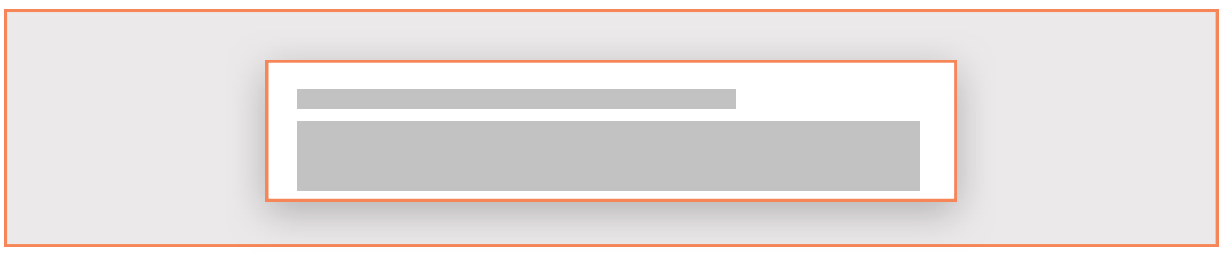
.hxlarge -> 20 lignes de haut

.modal-container

Cette class permet de créer des modals. Le modal n'a pas besoin d'être dans un row ou col.

Le background et l'animation sont créés, il manque juste un javascript qui rajoute open pour afficher le modal.

```
<div class="modal-container">
    <div class="modal">
        ...
    </div>
</div>
```



```
<div class="modal-container open ">
    <div class="modal">
        ...
    </div>
</div>
```

On peut afficher un x à droite du modal, en enlevant les commnetaire autour du fichier « x-on-modal » dans le fichier styles.scss

progress

Cet element permet de créer des carres de progression avec un tag html seulement.

Si le progress bar est dynamique, il faut créer un javascript qui permettera de mettre les données dans le progress bar.

```
<progress value="57" max="100"></progress>
```

Le dégrader a une couleur « de fin » dépandant de la valeur attribuée au progress bar.

```
<progress value="28" max="100">
</progress>
```

Ce pricinpe de couleur fonctionne seulement dans Chrome et Safari.

Dans Firefox, le progress bar est uniquement bleu avec 3d

Dans Edge et IE, le progress bar est uniquement bleu sans 3d

.tooltip

La classe tooltip s'accompagne de -left ou -right, pour la direction, si le tooltip est petit. Il permet de positioner le tooltip à gauche ou à droite. Cependant, quand le texte est long, le tooltip prend toute la largeur de la page.

Le tooltip se place simplement dans un div.


```
<div class="tooltip-left">...(texte)...</div>
```

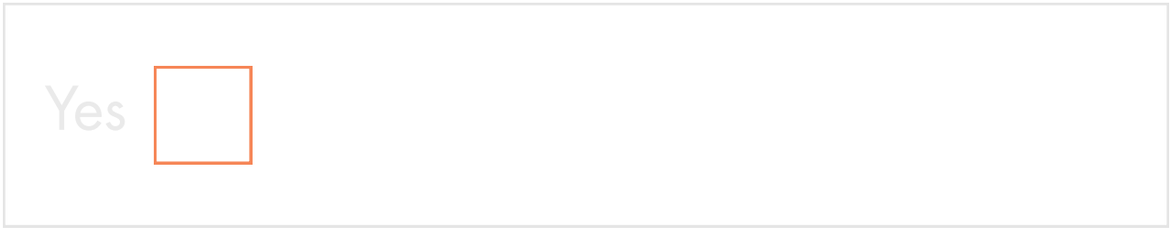
Lorsque l'on passe la souris sur le carré ce dernier disparait, et le texte, à l'intérieur du div, apparait.

Form component

Checkbox et radiobutton

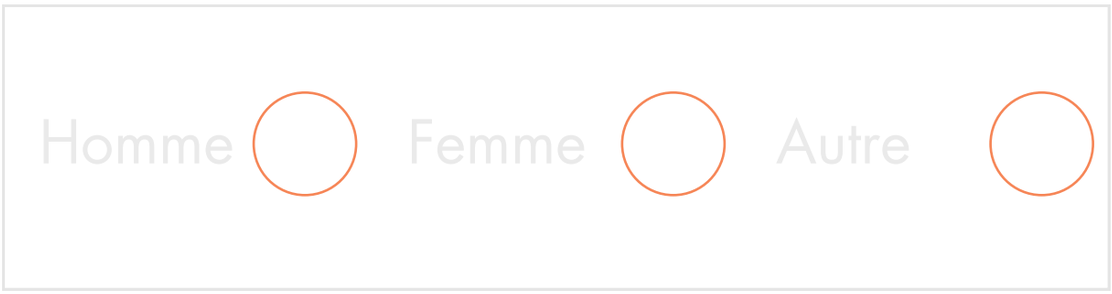
Pour avoir un element custom pour les checkboxs et radio buttons, on doit changer un peu le code. Les « for » et les « id » sont essentiels.

.checkbox




```
<div class="row">
    <div class="col checkbox"
        <label>Yes</label>
        <input type="checkbox" id="yes"  />
        <label for="yes"></label>
    </div>
</div>
```

.radiobutton




```
 <div class="row">
    <div class="col radio">
        <label>Homme</label>
        <input type="radio" name="gender' id="homme"/>
        <label for="homme"></label>

        <label>Femme</label>
        <input type="radio" name="gender" id="femme"/>
        <label for="femme"></label>

        <label>Autre</label>
        <input type="radio" name="gender" id="autre"/>
        <label for="autre"></label>
    </div>
</div>
```

Label

Il n'y a pas beaucoup de particularités avec eux. Ils ont toujours une largeur auto. Ils prendront toujours la place disponible pour être sur une ligne. Un label devrait avoir un nombre limité de mot, 2 ou 3 maximum.

label

Label

```
<div class="row">
    <div class="col"
        <label></label>
        <input type="text"  />
    </div>
</div>
```

On devrait écrire un label avec un « for » pour un meilleur UX. Lorsque l'on click sur le label avec un «for», le curseur va directement dans le input associé.

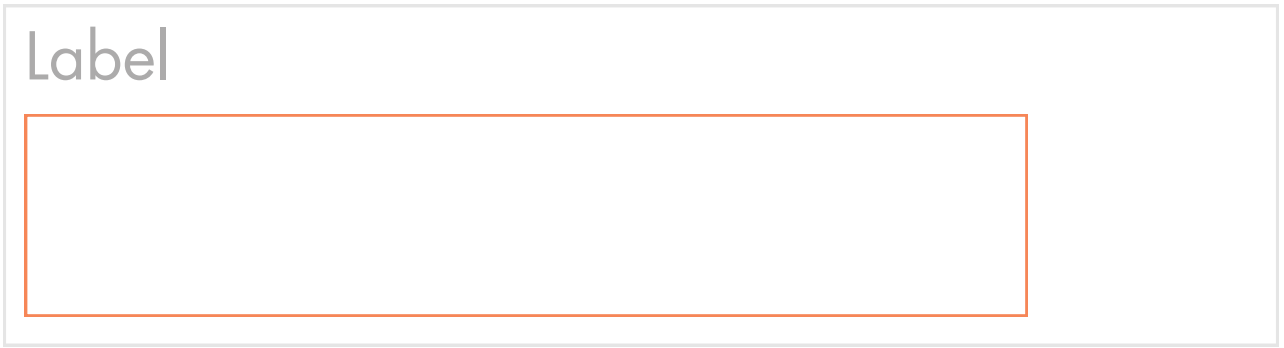
label

Label

```
<div class="row">
    <div class="col"
        <label for="name"></label>
        <input type="text" id="name"   />
    </div>
</div>
```

Textarea

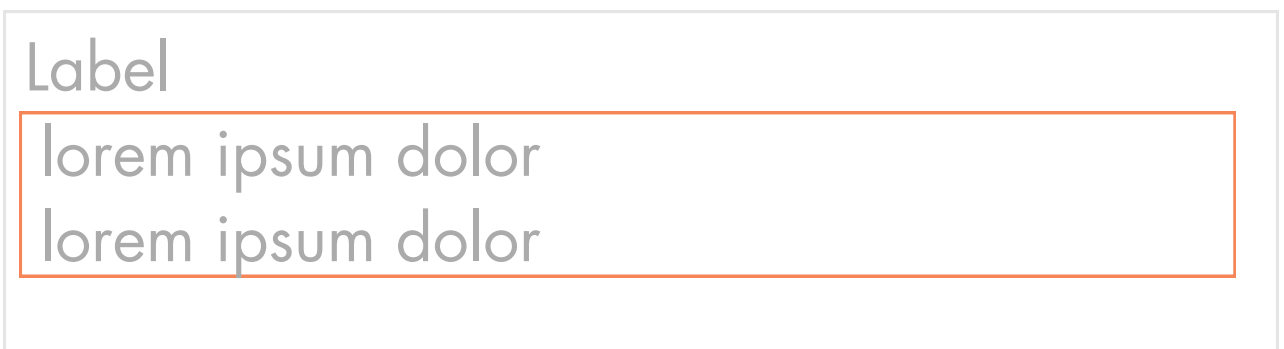
textarea



Le label devrait toujours être en haut du textearea.

```
<div class="row">
    <div class="col">
        <label>Label</label>
        <textarea></textarea>
    </div>
</div>
```

Le textarea prend toute la largeur disponible dans la colonne.
On peut aussi mettre la hauteur du textarea avec des classes.
On applique directement ces classes au textarea. Les classes sont line1 à line10




```
<div class="row">
    <div class="col">
        <label>Label</label>
        <textarea class="line2"></textarea>
    </div>
</div>
```

.curency et pourcent

Ces deux class sont déjà prêt il faut simplement mettre un label vide à la fin. On applique la class directement sur la row ou la colone. Sauf que la colone ne peut pas wrapper. Donc il est plus sécuritaire de l'utiliser dans un div séparé.

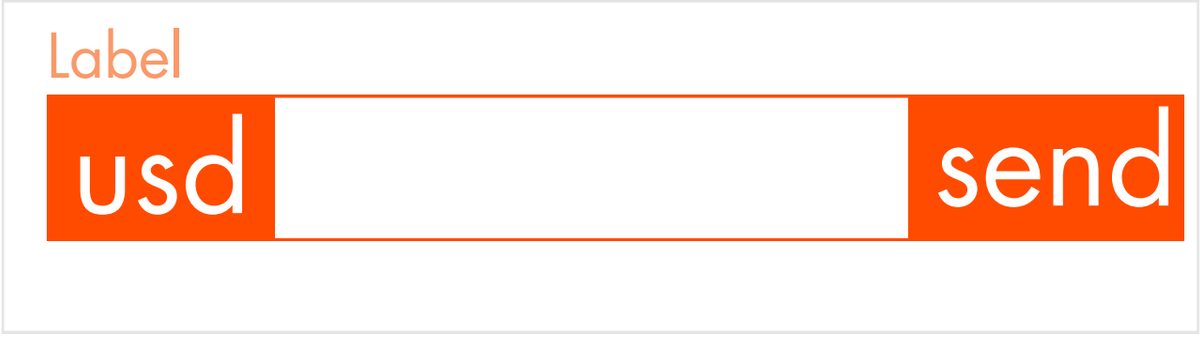

```
<div class="row">
    <div class="col wrap"
        <label>…</label>
        <div class="currency">
            <input type="number"  />
            <label></label>
        </div>
    </div>
</div>
```

le $ ou le % ce rajoute automatiquement suite au dernier label. le premier label doit etre a l'extérieur du div. Si on veut tout mettre en ranger on a juste a mettre no-wrap à la colonne.


```
<div class="row">
    <div class="col nowrap">
    <label>pourcentage</label>
        <div class="pourcent"
            <input type="number"/>
            <label></label>
        </div>
    </div>
</div>
```

.input-group

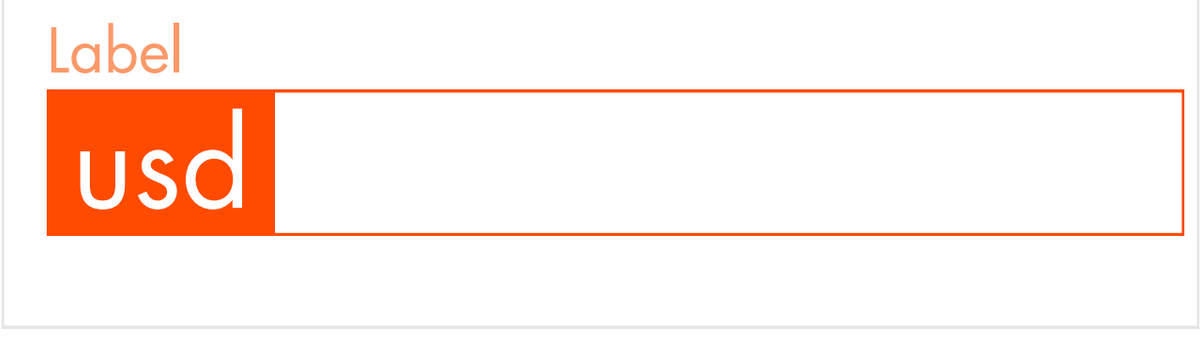
Cette class permet de recréer le même principle que pourcent ou currency, mais en créant des éléments personnalisés en avant et un bouton à la fin si nécessaire.




```
<div class="row">
    <div class="col wrap">
        <label>...</label>
        <div class="input-group">
            <label>usd</label>
            <input type="number"  />
            <input type="submit" value="send" />
        </div>
    </div>
</div>
```

Le premier label se style automatiquement. Le bouton à la fin, aussi, le texte choisit peut être modifié.

Il est possible aussi de faire le même principe sans bouton à la fin en utilisant la class input-group-noSubmit.
Il faut alors supprimer le bouton à la fin dans le code.




```
<div class="row">
    <div class="col wrap">
        <label>...</label>
        <div class="input-group-noSubmit">
            <label>usd</label>
            <input type="number"  />
        </div>
    </div>
</div>
```