college CD

Web design and development II (CW2)

Class 11

Interface slicing and integration

Methodology / workshop

Planning integration process

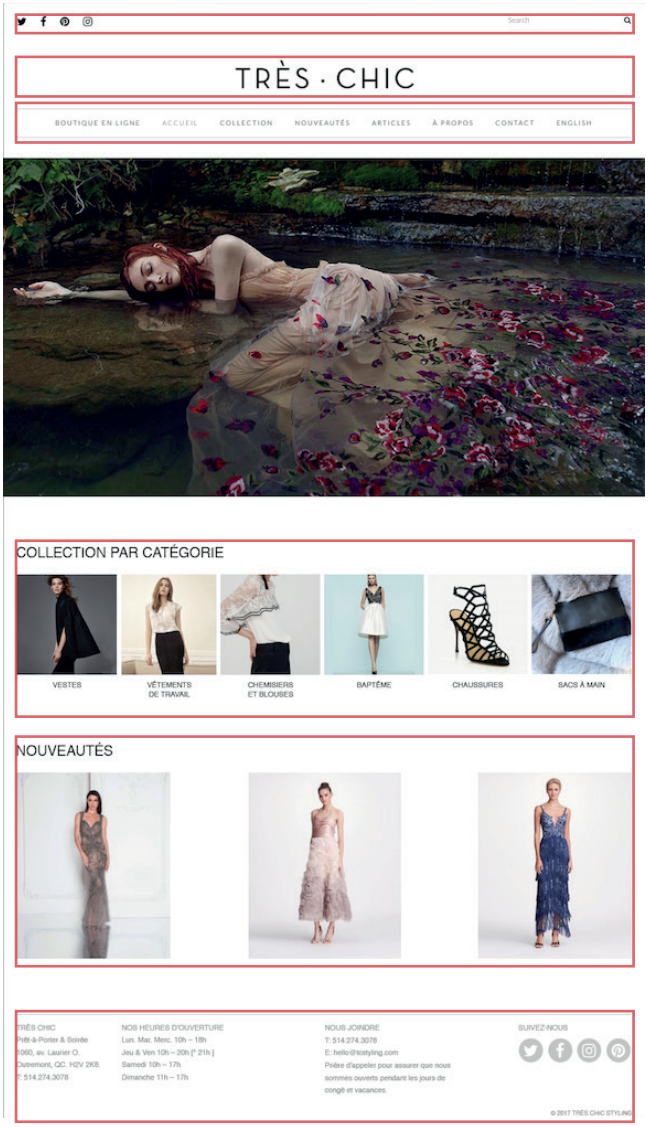
Before starting any kind of integration work, you must plan it. You need a strategy, to know how you will proceed.

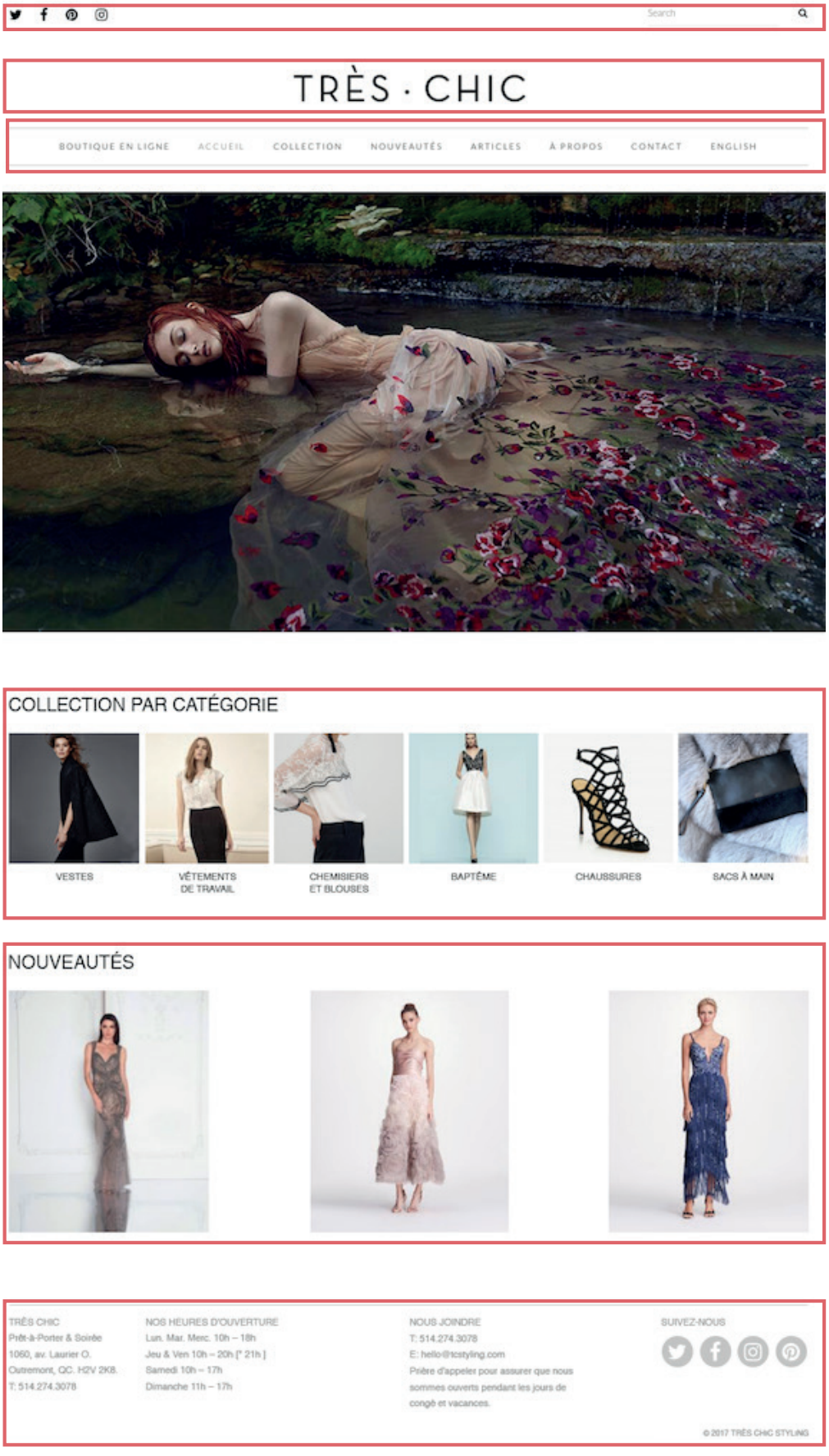
Unfortunately, you will need experience to plan your work efficiently, that is why it is important to practice as much as you can.

You need to answer many questions such as :

- Is there a background image? If there is, will it adapt to the browser's window or will it have a limited width and be centered?
- Will content be displayed full width or limited in width?
- The main structure will count how many rows?
- Will there be breaking point for different viewports? How will content adapt?

In the example beside, the red borders indicates content limited to 960px wide centered containers. However, the image following the navigation should be full width.




Search
TRÈS · CHIC
BOUTIQUE EN LIGNE
ACCUEIL
COLLECTION
NOUVEAUTÉS
ARTICLES
À PROPOS
CONTACT
ENGLISH
COLLECTION PAR CATÉGORIE
VESTES
VÊTEMENTS DE TRAVAIL
CHEMISIERS ET BLOUSES
BAPTÊME
CHAUSSURES
SACS À MAIN
NOUVEAUTÉS
TRÈS CHIC
Prêt-à-Porter & Soirée
1060, av. Laurier O.
Outremont, QC. H2V 2K8.
T: 514.274.3078
NOS HEURES D'OUVERTURE
Lun. Mar. Merc. 10h – 18h
Jeu & Ven 10h – 20h [* 21h]
Samedi 10h – 17h
Dimanche 11h – 17h
NOUS JOINDRE
T: 514.274.3078
E: hello@tcstyling.com
Prière d'appeler pour assurer que nous sommes ouverts pendant les jours de congé et vacances.
SUIVEZ-NOUS
© 2017 TRÈS CHIC STYLING

Header

In the example above, you can see the three containers (red borders) which are included in the full width <header> section.

Those three containers are limited to a width of 960px centered in the <header> section.

First line container

This container includes two groups of elements (yellow borders): social network on the left and a search field and button on the right.

Float could be used to position the two groups at the extreme left and right, or we could position them using *position: absolute*, or again *flex* could be used with *justify-content: space-between*.

Since it will be most probably useful to center our contents vertically, flex would probably be the best choice with the property *align-items: center*.

Second line container

This one is very easy as the text or the image can be centered using *text-align: center* or in flex using *justify-content: center*.

Here again, flex would allow to align the content vertically.

Third line container

In this container, spacing the button evenly would be quite difficult otherwise than using flex. Using *justify-content: space-around* will spread efficiently your navigation items in a split of a second.

HTML:

```
<header>
        <header>
                <div class="social_media"> </div>
                <div class="search"> </div>
        </header>
        <div>
                <img class="logo" src="logo.png" />
        </div>
        <nav>
                <button>Link</button>
                ...
        </nav>
</header>
```

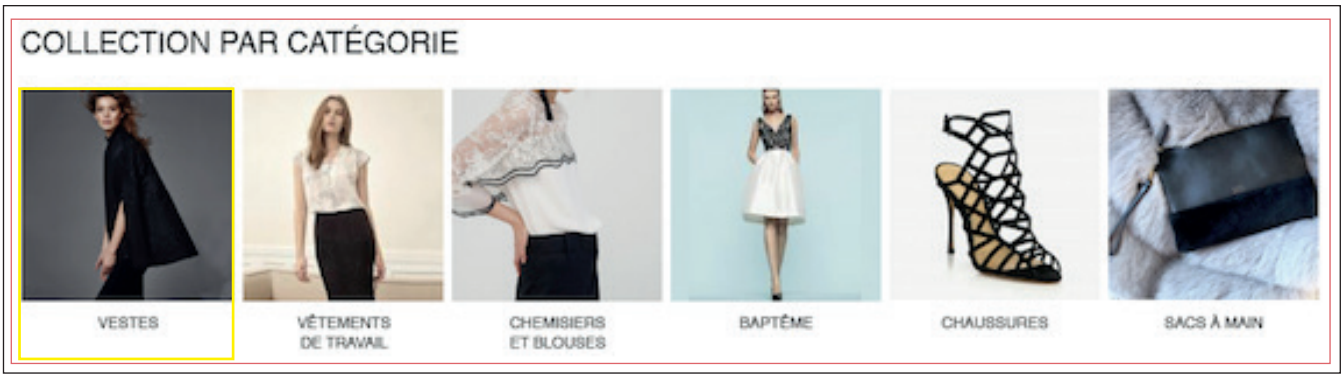
CSS:

```
header {
        width: 100%;
        height: auto;
}

header header {
        display: flex;
        justify-content: space-between;
        max-width: 960px;
        margin: auto;
}

header div {
        display: flex;
        justify-content: center;
        max-width: 960px;
        margin: auto;
}

header nav {
        display: flex;
        justify-content: space-around;
        max-width: 960px;
        margin: auto;
}
```



Main

The full width image would be placed by itself (not in a container) between <header> and <main> sections.

Using ***display: block*** allows to use ***margin: auto*** to center the image horizontally just like we would with a container.

<main> would use ***max-width: 960px*** and ***margin: auto*** to limit the horizontal spread of contents.

Fourth line container

This container, formed by <section> (red borders) is the first one of the main section and it includes six items we'll call *boxes* (yellow borders) consisting of an image with a text underneath.

In order to size and align them correctly, we'll use a six columns grid and the function ***repeat(6, 1fr)*** which means six columns using equal widths.

Using ***position: relative*** will allow us to *position: absolute* the <h1> later.

Using ***display: block*** on the image will keep us from having to use a
 between the image and the <h3>.

HTML:

```
<img class="banner" src="banner.jpg" />

<main>
<section>
        <h1>My section's title</h1>
        <div><img src="box1.jpg" /><h3>Box 1</h3></div>
        ...
</section>
</main>
```

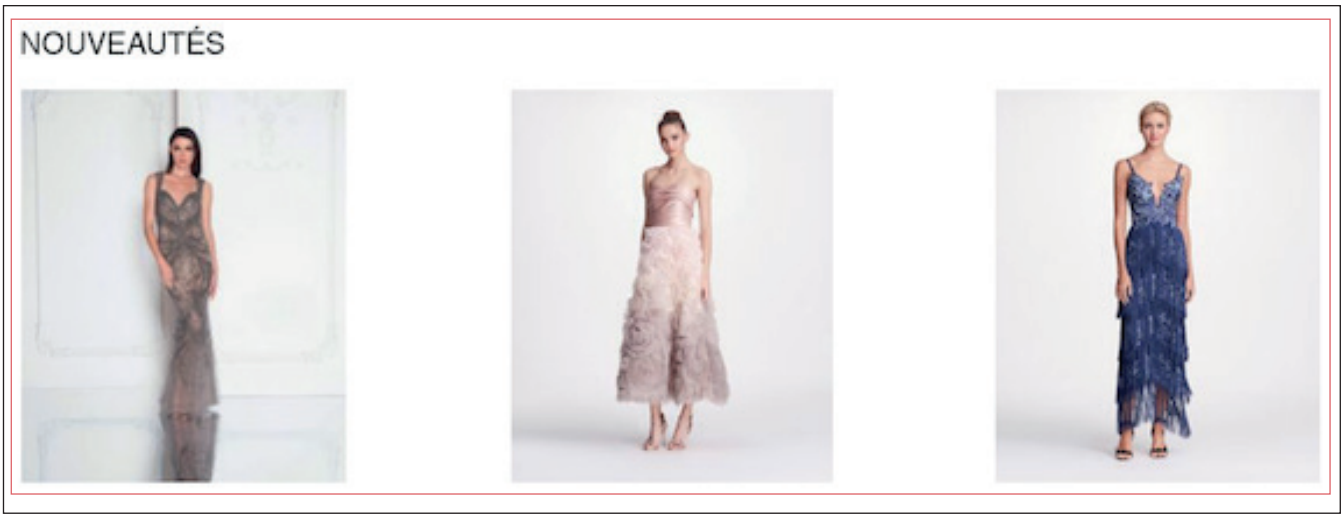
CSS:

```
img.banner {
        display: block;
        width: 100%;
        max-width: 960px;
        height: auto;
        margin: auto;
}

main section {
        position: relative;
        display: grid;
        grid-template-columns: repeat(6, 1fr);
        grid-gap: 10px;
}

main section h1 {
        position: absolute;
        top: 10px;
        left: 10px;
}


main section div img {
        display: block;
        max-width: 100%;
        height: auto;
}
```



Main

Fifth line container

This one is a lot like the preceding section, but even simpler. It only contains a title and three pictures.

The easiest way to manage this section is most probably using ***flex*** to spread the three elements using ***justify-content: space-between***.

We'll be using a class (***.fifth***) on this section as <section> for <main> has already been styled.

The images

Using ***position: relative*** allows us to use ***top: 30px*** so the title (<h1>) in *position: absolute* won't be superimposed.

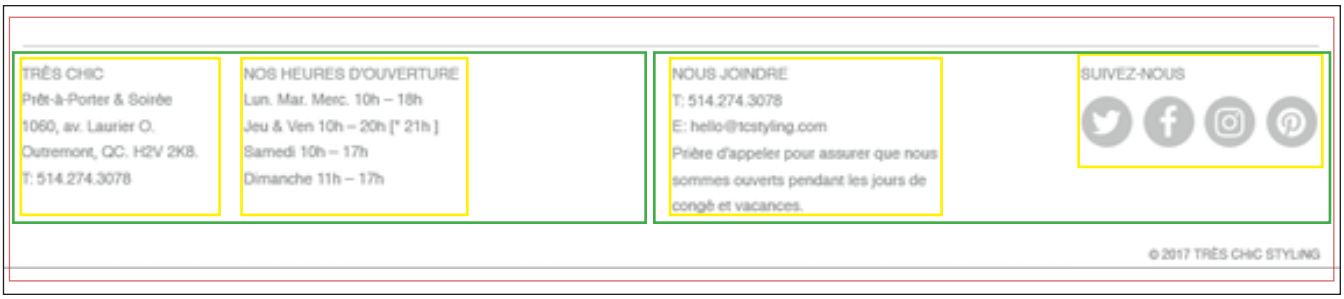
HTML:

```
<section class="fifth">
        <img src="image1.jpg" />
        <img src="image2.jpg" />
        <img src="image3.jpg" />
</section>
```

CSS:

```
main section.fifth {
        position: relative;
        display: flex;
        justity-content: space-between;
}


main section.fifth img {
        position: relative;
        top: 30px;
        display: block;
        max-width: 100%;
        height: auto;
}
```



Footer

Sixth line container

The sixth and final line of the web page (red borders) contains two groups (green borders) in which are placed the four content containers (yellow borders).

<footer> (***display: flex***) shows the two <section> (*.left* and *.right* / green borders) side by side. Both are using equal parts of the footer as they use ***flex-grow: 1***.

Using classes, the first <section> aligns its items left and the second <section> uses ***justify-content: space-between***.

<aside> are just using padding and will stretch to the volume of their contents.

The horizontal lines could have been produced with ***<hr />*** and the copyright could have been ***position: absolute***.

It would also be possible to make the header ***position: fixed*** so it always remain in place. Then, the use of ***z-index*** would be mandatory in order to have the header on top of the following elements.

HTML :

```
<footer>
        <section class="left">
                <aside>Some text</aside>
                <aside>Some text</aside>
        </section>
        <section class="right">
                <aside>Some text</aside>
                <aside>Social media icons</aside>
        </section>
</footer>
```

CSS :

```
footer {
        position: relative;
        display: flex;
        justity-content: space-between;
        max-width: 960px;
        margin: auto;
}

footer section.left {
        position: relative;
        display: flex;
        justity-content: flex-start;
        flex-grow: 1;
}

footer section.right {
        position: relative;
        display: flex;
        justity-content: space-between;
        flex-grow: 1;
}

footer section aside {
        padding 10px;
}
```