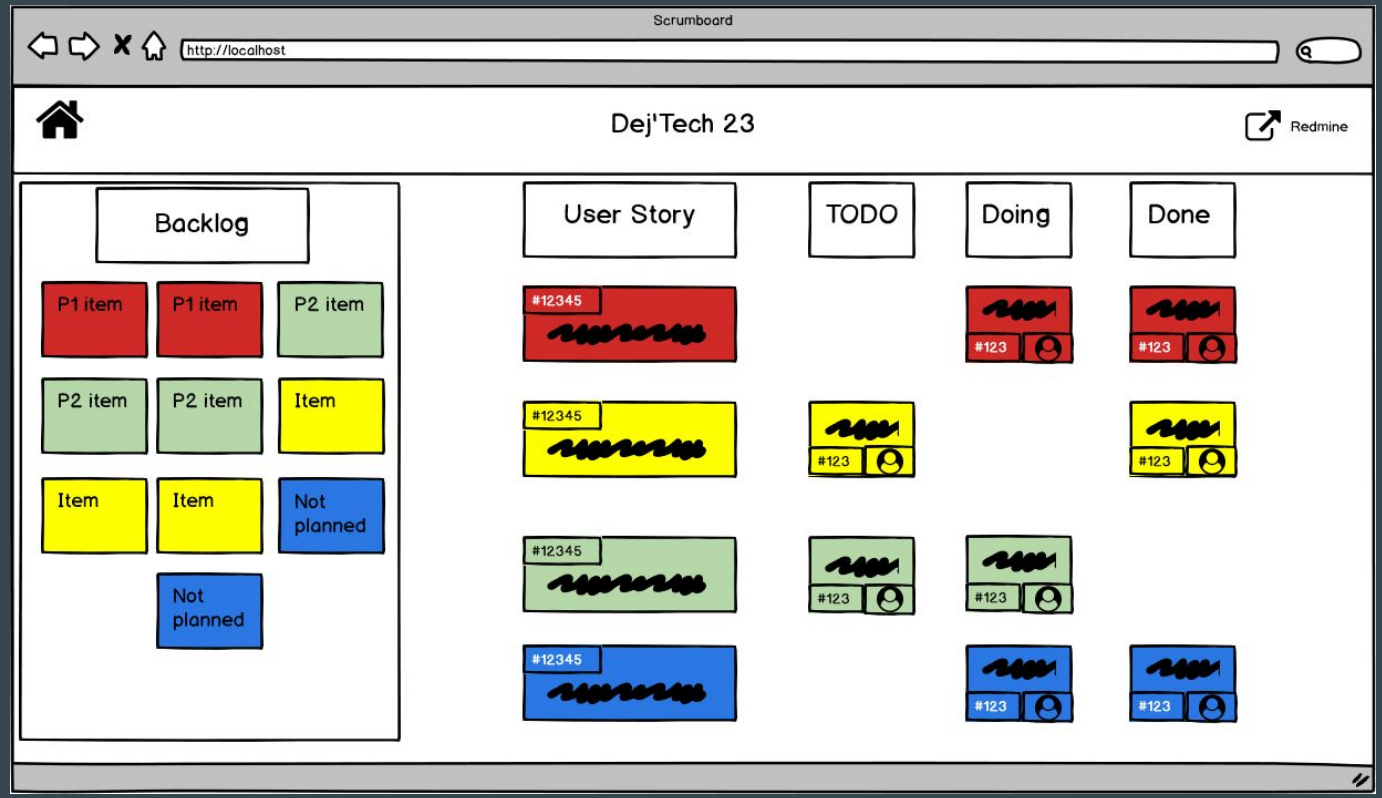
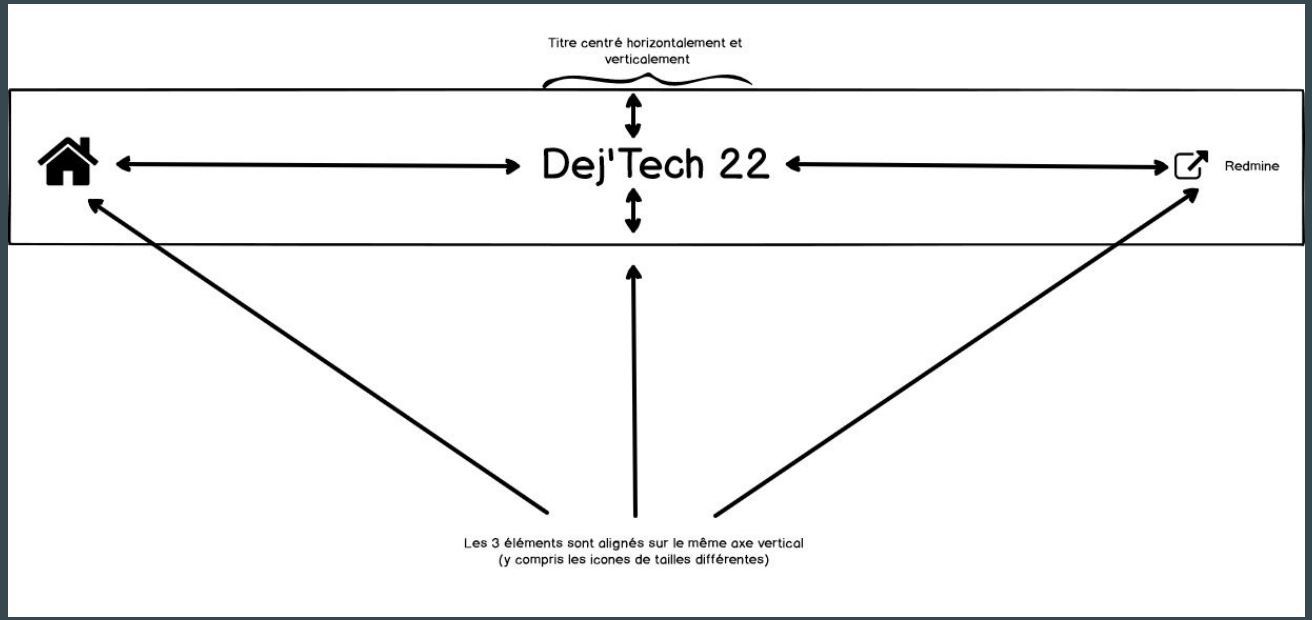
...

Focus CSS : Flex & Grid

Notre fil rouge : Réalisation d'un Scrumboard



Premier élément : le menu principal



Le menu : comment faire ?

Positionner les éléments horizontalement :

```
.menu {
   float: left;
}
```

Centrer verticalement ?

Il faut un article : https://css-tricks.com/centering-css-complete-guide/ (et pas un petit).

Le menu : comment faire ?

CSS Flex

a CSS box model optimized for user interface design. In the flex layout model, the children of a flex container can be laid out in any direction, and can “flex” their sizes, either growing to fill unused space or shrinking to avoid overflowing the parent. Both horizontal and vertical alignment of the children can be easily manipulated

https://www.w3.org/TR/css-flexbox-1/

CSS Flexible Box Layout Module - CR

Method of positioning elements in horizontal or vertical stacks. Support includes all properties prefixed with `flex`, as well as `display: flex`, `display: inline-flex`, `align-content`, `align-items`, `align-self`, `justify-content` and `order`.

Usage	% of all users		
Global	93.2%	+ 3.47% =	96.68%
unprefixed:	92.92%	+ 2.75% =	95.67%
France	93.79%	+ 2.75% =	96.54%
unprefixed:	93.5%	+ 2.13% =	95.63%

Current aligned | Usage relative | Date relative | Show all

Notes | Known issues (0) | Resources (12) | Feedback

Le menu avec Flex

http://localhost:3000/menu/menu.html

Le menu avec Flex

HTML

```
<nav class="flex-container">
      <a class="flex-item" link='#'>
            <img src="../assets/home.svg">
      </a>
      <h1 class="flex-item">Scrumboard</h1>
      <a class=
      "flex-item exit flex-container" link='#'>
            <img class="flex-item"
            src="../assets/exit.svg">Redmine
      </a>
</nav>
```

CSS

```
.flex-container {
 display: flex;
 flex-direction: row;
 justify-content: flex-start;
 align-items: center;
}

h1.flex-item {
 font-size: 3em;
 margin: auto;
}
```

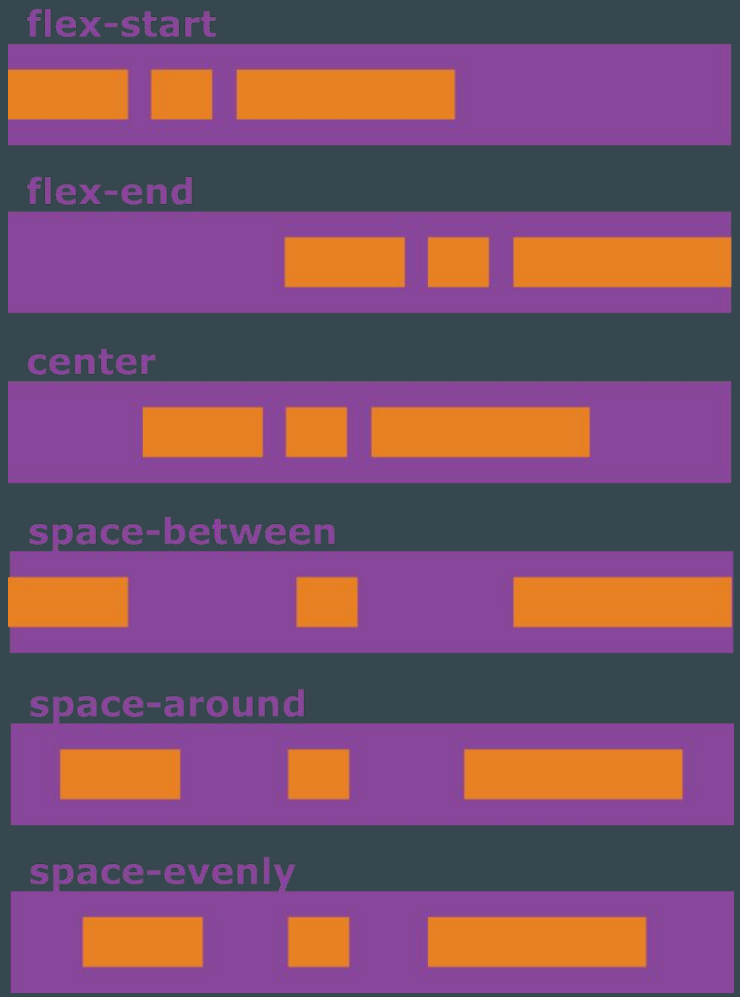
Flex-direction

```
.container {
 flex-direction: row | row-reverse | column | column-reverse;
}
```

Justify-content

```
.container {
 justify-content: flex-start | flex-end | center
      | space-between | space-around | space-evenly;
}
```

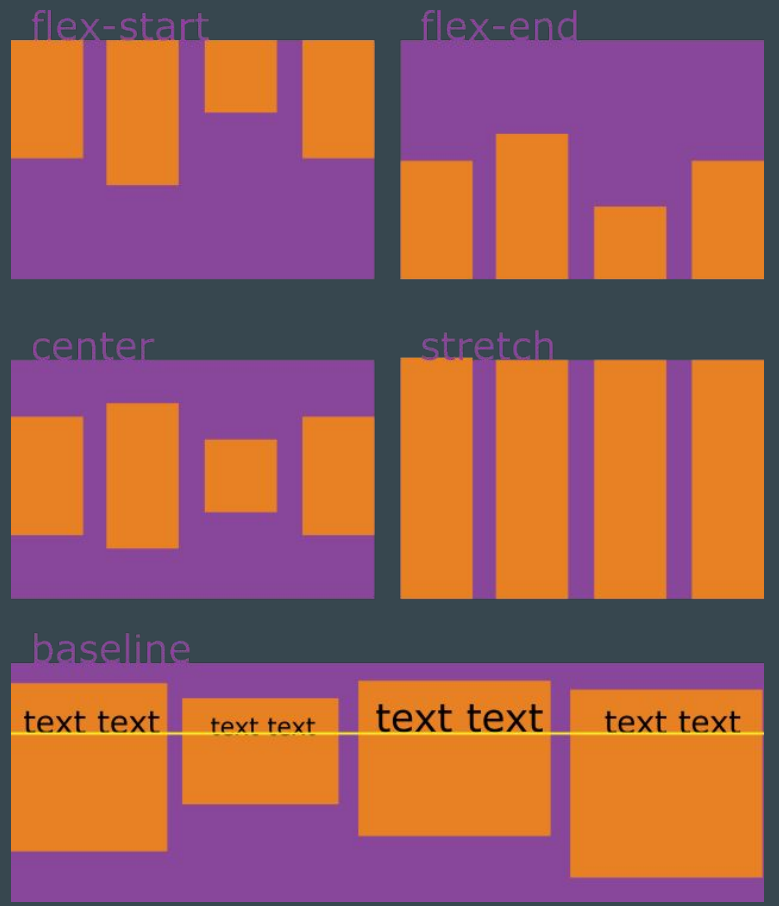
Alignement autour de l'axe principal du parent



Align-items

```
.container {
 align-items: flex-start | flex-end
            | center | baseline | stretch;
}
```

Alignement autour de l’axe transversal du parent



Position des éléments

```
h1.flex-item {
 font-size: 3em;
 margin: auto;
}
```

La magie : `margin: auto;`

margin: auto absorbe l’espace supplémentaire.

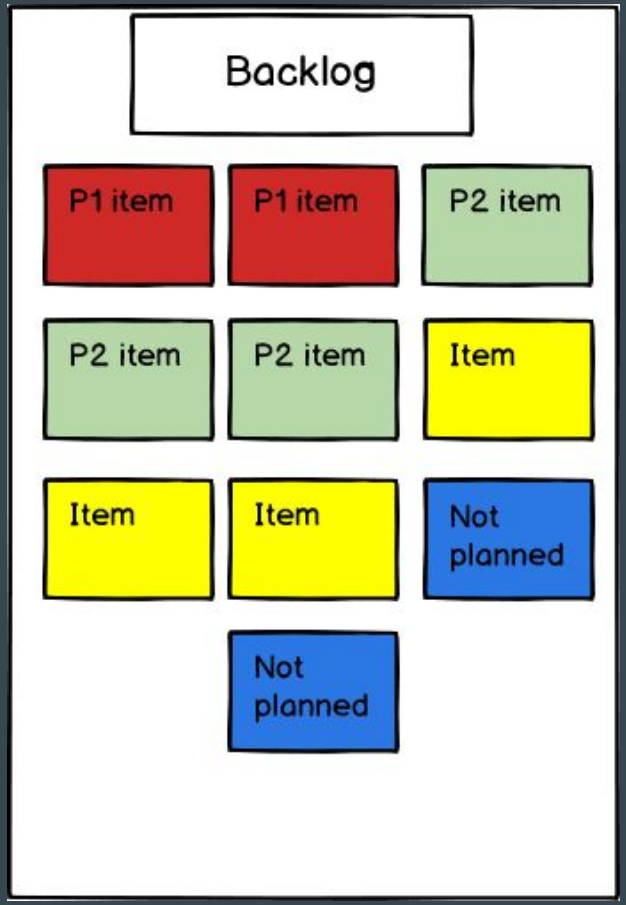
Deuxième élément : le backlog

Contraintes :

- S’adapte automatiquement

  à l’espace disponible

- Tri les post its par priorité.

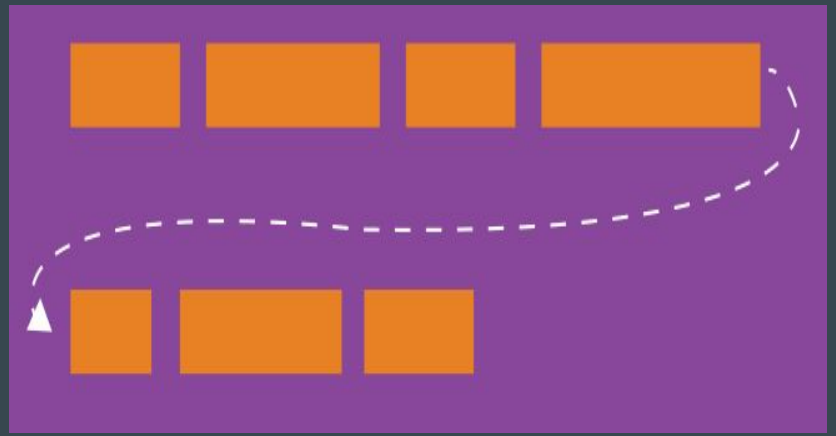


Le backlog

http://localhost:3000/backlog/backlog.html

Flex-wrap

```
.container{
 flex-wrap: nowrap | wrap | wrap-reverse;
}
```



Par défaut, on essaye de tenir sur une ligne.

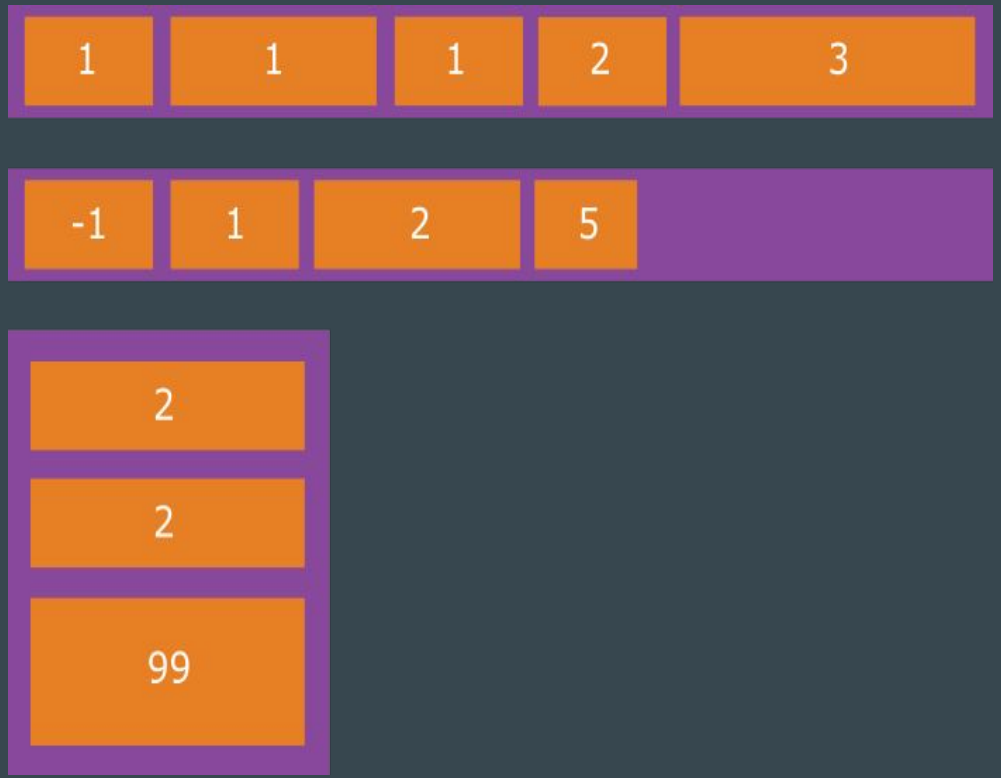
flex-wrap permet de passer à la ligne si besoin.

Order:

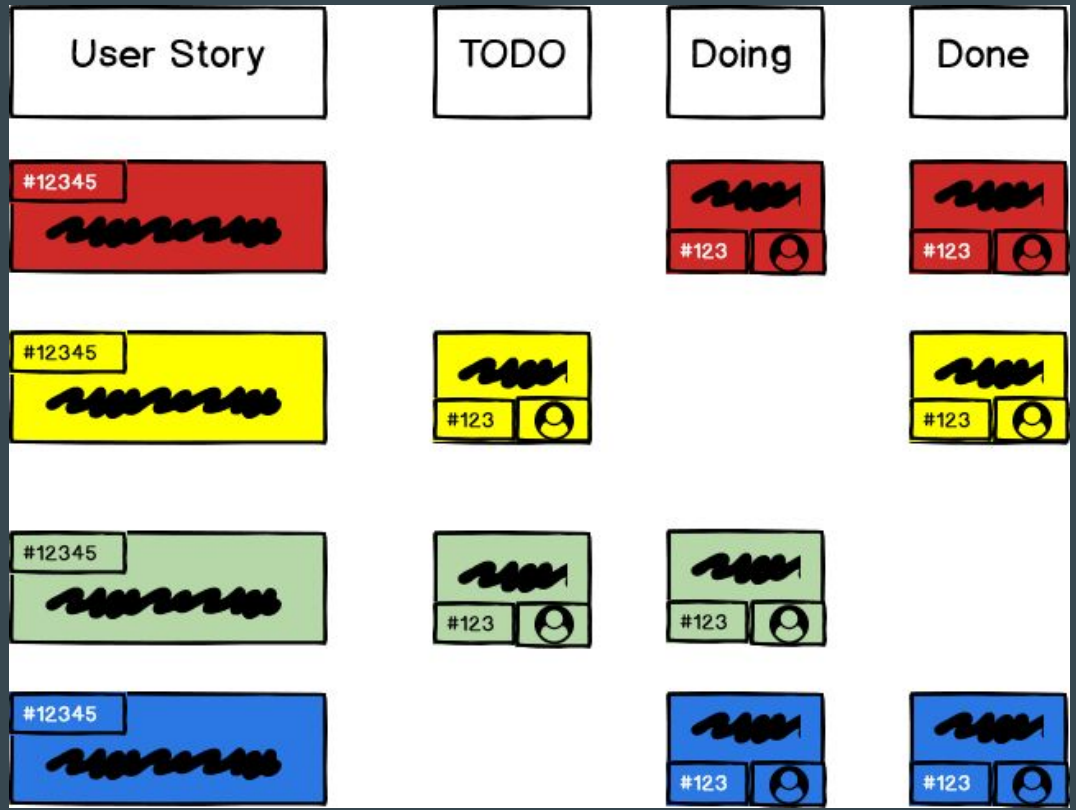
```
.item {
 order: <integer>; /* default is 0 */
}
```

Propriété de l'élément.

Par défaut : position dans le code.



Troisième élément: le scrumboard

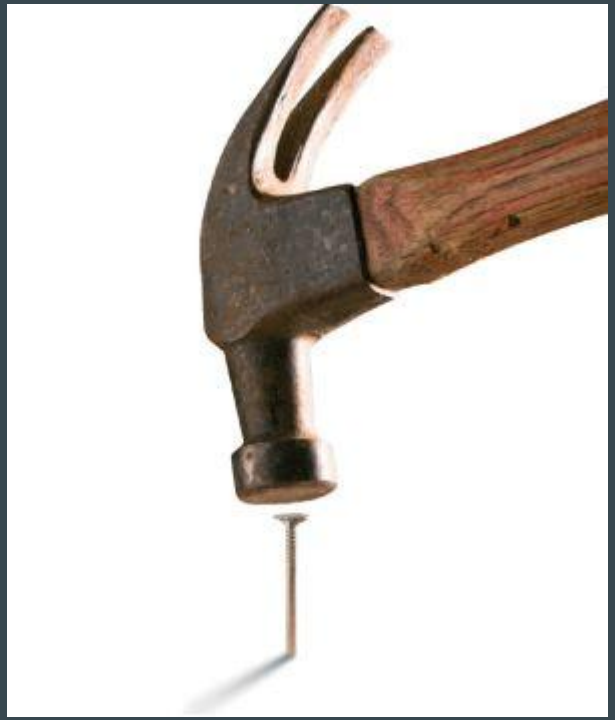


Flex ou Grid ?

On peut faire de la mise en page avec Flex :

https://codepen.io/chriscoyier/pen/vWEMWw

Mais il faut préférer CSS Grid pour les mises à en page à 2 dimensions



Utilisons CSS Grid

CSS Grid c'est quoi ?

> *This CSS module defines a two-dimensional grid-based layout system, optimized for user interface design. In the grid layout model, the children of a grid container can be positioned into arbitrary slots in a predefined flexible or fixed-size layout grid.*
>
> *https://www.w3.org/TR/css-grid-1/*

CSS Grid Layout - CR

Method of using a grid concept to lay out content, providing a mechanism for authors to divide available space for layout into columns and rows using a set of predictable sizing behaviors. Includes support for all `grid-*` properties and the `fr` unit.

Usage	% of all users
Global	84.53% + 3.2% = 87.73%
unprefixed:	84.53%
France	87.65% + 2.71% = 90.36%
unprefixed:	87.65%

Current aligned | Usage relative | Date relative | Show all

Le scrumboard

http://localhost:3000/scrumboard/scrumboard.html

Scrumboard: définition d’une grille

https://jsfiddle.net/g5ka31jz/

Déclarer une grille : grid-template-rows / grid-template-columns

3 grandes syntaxes possibles :

https://developer.mozilla.org/en-US/docs/Web/CSS/grid-template-rows

https://developer.mozilla.org/en-US/docs/Web/CSS/grid-template-columns

Le plus important : la syntaxe “repeat”

```
grid-template-columns: repeat(3, 1fr);
```

Nouvelle unité : 1fr

Correspond à une fraction de l'espace disponible.

https://css-tricks.com/introduction-fr-css-unit/

https://www.w3.org/TR/css3-grid-layout/#fr-unit

Positionner un élément

A l'aide des numéros de lignes / colonnes

```
/* syntaxe longue */
.box2 {
 grid-column-start: 3;
 grid-column-end: 4;
 grid-row-start: 1;
 grid-row-end: 3;
}
```

```
/* écriture raccourcie */
.box2 {
 grid-column: 3 / 4;
 grid-row: 1 / 3;
}
```

Positionner un élément

A l'aide des "aires de grille"

```
.box2 {
 grid-area: 1 / 3 / 3 / 4;
}
```

On définit une aire à l'aide de 4 coordonnées (dans l'ordre) :

grid-row-start / grid-column-start / grid-row-end / grid-column-end

Positionner un élément

Il est possible d'étendre un élément sur plusieurs lignes / colonnes.

```
.box2 {
 grid-row-end: span 2;
}
```

span est applicable à grid-row-start / grid-row-end et grid-column-start / grid-column-end

Améliorons nos posts its !

Le scrumboard

http://localhost:3000/better-postit/better-postit.html

Améliorons nos posts its !

Nommons les aires associés à un élément :

```
.small-postit {
  display: grid;
  grid-template-columns: repeat(2, 1fr);
  grid-template-rows: repeat(3, 1fr);
  grid-template-areas: "desc desc "
                       "desc desc "
                       "issue avatar";
}

.small-postit>.avatar {
  grid-area: avatar;
}
```

Le scrumboard

http://localhost:3000/better-scrumboard/scrumboard.html

https://gridbyexample.com/examples/example21/

Utilisons CSS Grid pour la mise en page globale

```
.main-grid {
   display: grid;
   grid-template-columns: repeat(4, 1fr);
   grid-template-areas: "header header header header"
                        "aside main main main";
}
.main-grid>aside {
   grid-area: aside;
}
.main-grid>header {
   grid-area: header;
}
.main-grid>main {
   grid-area: main;
}
```

```
@media only screen and (max-width: 1200px) {
   .main-grid {
       grid-template-columns: auto;
       grid-template-areas:  "header"
                             "main"
                             "aside";
   }
}
```

Mise en page globale

http://localhost:3000/main.html

Un peu de ménage

HTML

```
<body class='main-grid'>
    <header>

    </header>
    <main >

    </main>
    <aside>

    </aside>
</body>
```

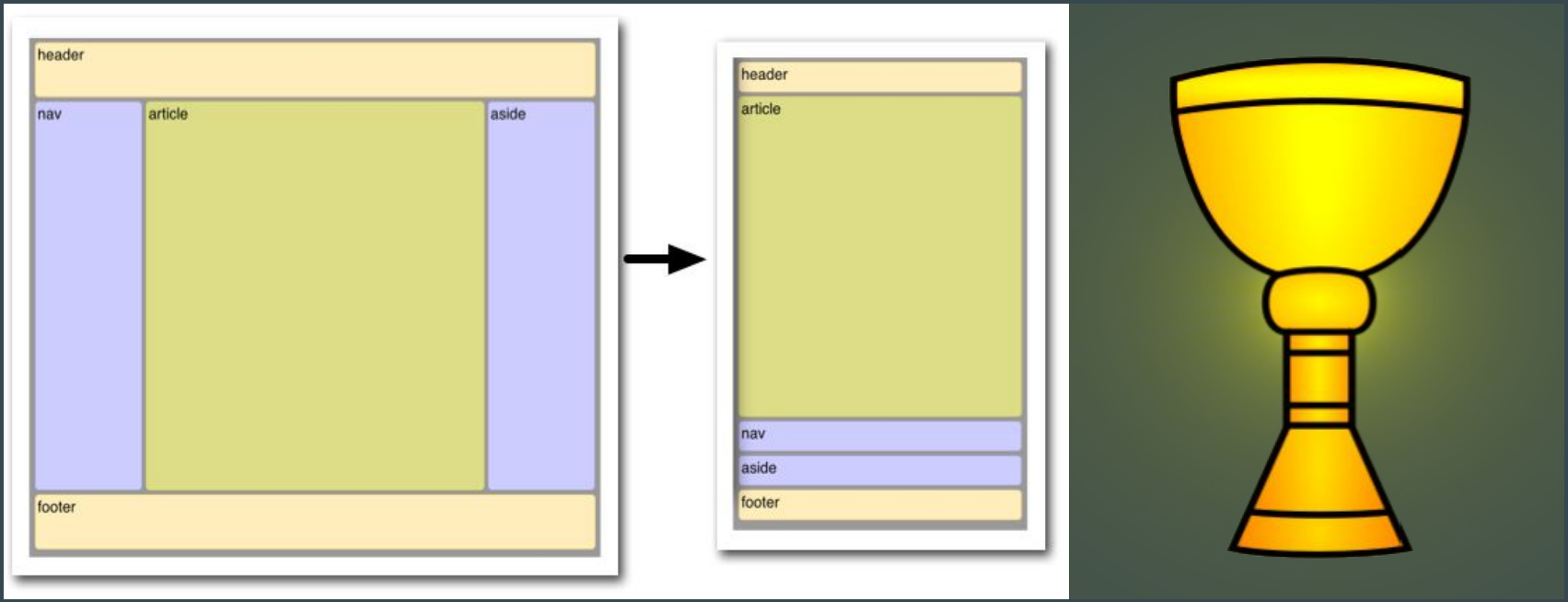
CSS

```
.main-grid {
 height: 100vh;
}

.main-grid > aside {
 background-color: rgba(0, 255, 0, 0.4);
}

.main-grid > header {
 background-color: rgba(255, 255, 0, 0.4);
 height: 10vh;
}
```

Résultat

http://localhost:3000/layout.html

Résultat


header
nav
article
aside
footer
header
article
nav
aside
footer

Pour aller plus loin

https://css-tricks.com/snippets/css/a-guide-to-flexbox/

https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout

https://css-tricks.com/snippets/css/complete-guide-grid/

https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Grid_Layout

https://developer.mozilla.org/en-US/docs/Tools/Page_Inspector/How_to/Examine_grid_layouts

https://rachelandrew.co.uk/archives/2016/03/30/should-i-use-grid-or-flexbox/

Quelques exemples

https://gridbyexample.com/

http://labs.jensimmons.com/

https://demos.scotch.io/visual-guide-to-css3-flexbox-flexbox-playground/demos/