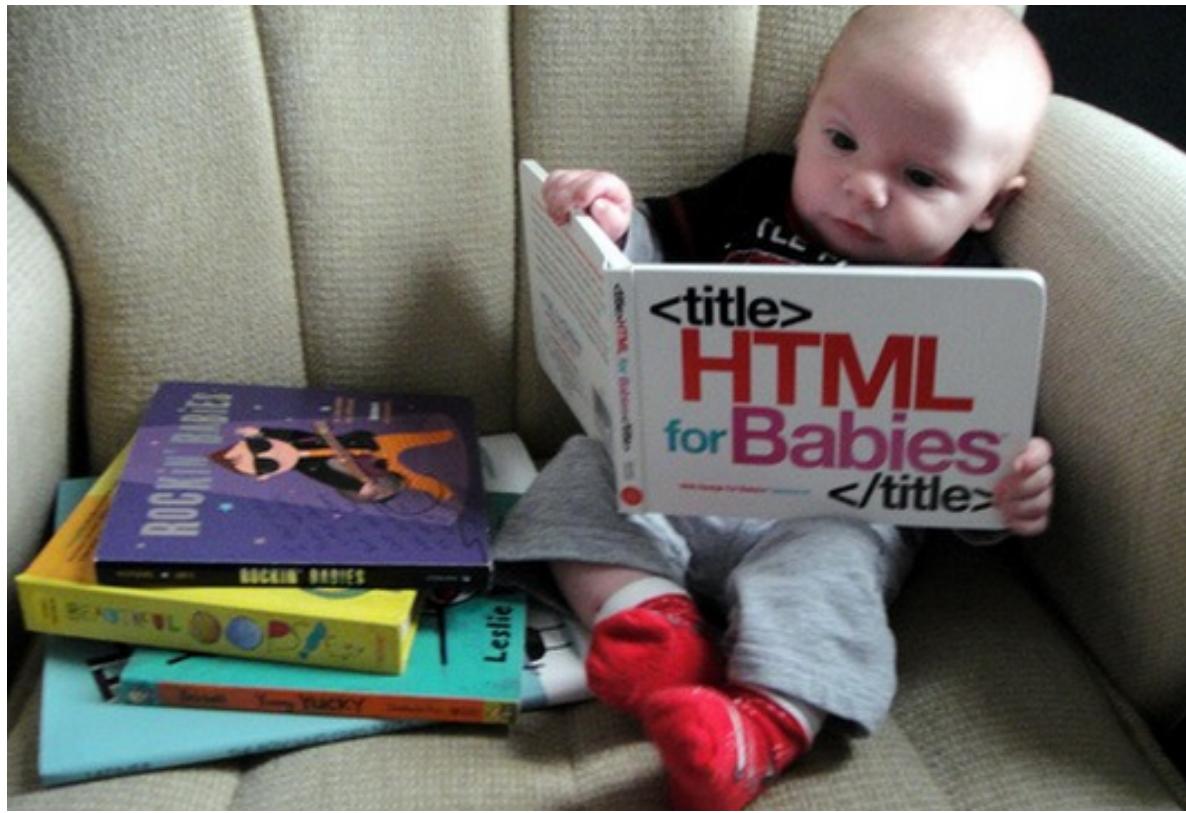


CSS

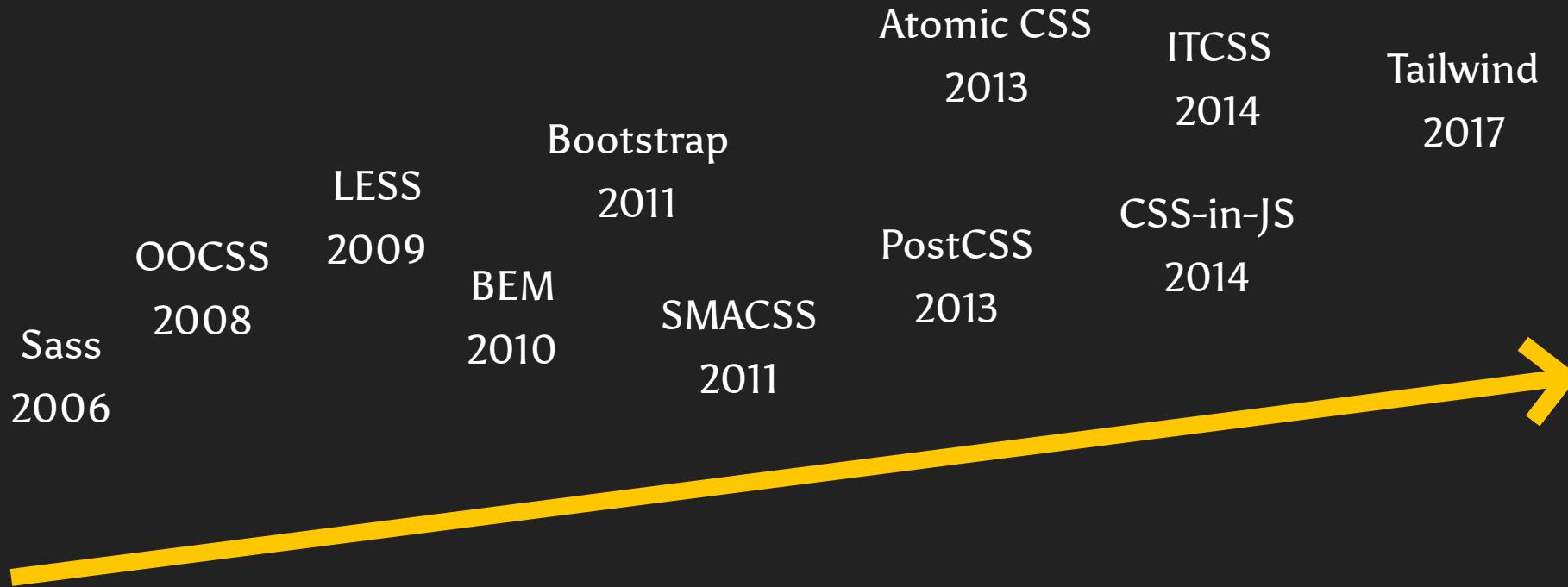
Méthodologies et conventions

conventions de nommage, BEM, Tailwind, etc.



Évolution des méthodologies CSS

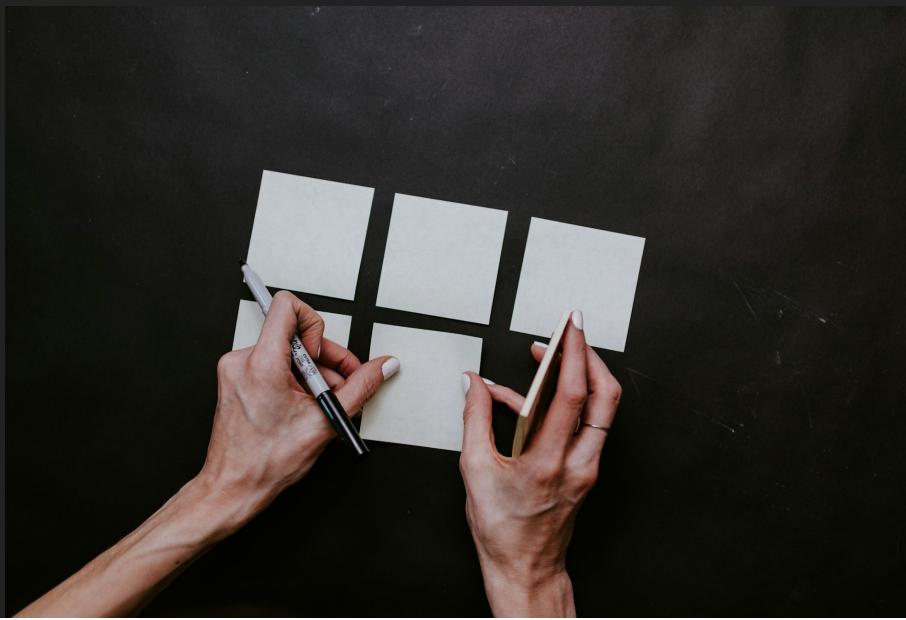
un peu d'histoire



Au programme

"un peu d'organisation, diantre!"

1. Les problématiques à résoudre
2. Appliquer une méthodologie (BEM, Atomique)
3. Du design à l'intégration

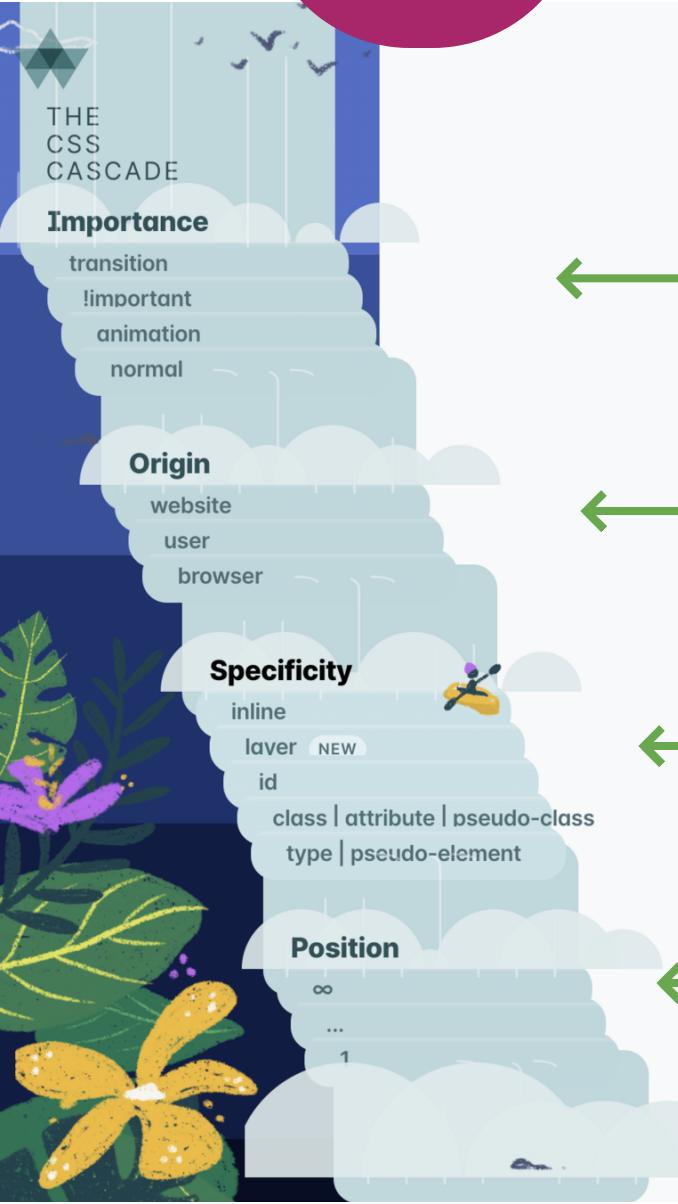


Les principales Problématiques à résoudre



1

La Cascade CSS



!important, transitions, animations

styles utilisateur, styles navigateur, styles du webdeveloppeur

règles de spécificité (poids) des sélecteurs

ordre d'affichage (le dernier écrase le précédent)

La Cascade CSS

```
ul#navigation > .link { color: salmon }
ul#navigation li a { color: pink }
#navigation a.link { color: olive }
body ul#navigation li a { color: tomato }
body nav ul li a.link:first-child { color: chocolate }
```

Inconvénients majeurs :

1. Chaque type de sélecteur CSS a sa propre spécificité
2. Ajouter du poids pour écraser un sélecteur a un effet "boule de neige" pour la prochaine modification à opérer
3. Pourquoi ce    de bouton ne devient pas rose ?!?!?

Solutions envisageables :

1. Connaître la cascade et la spécificité des sélecteurs
2. Employer des astuces modernes (@layer, :where, @scope)
3. Choisir des sélecteurs légers (.class et non #id)
4. C'est le principal job des méthodologies telles que BEM ou Atomique

Nommage des éléments

```
.img, .image, .blog-img, .img-blog, .blogImage, .thumbnail_img { ... }  
.btn, .bouton, .button, .btn--primary, .close-btn, .btn-close { ... }  
.wrapper, .container, .items-container, .inner-wrapper,  
.wrapper__inner--alternate { ... }  
.title-main, .post-title, .heading { ... }
```

Inconvénients majeurs :

1. Difficile de trouver des noms cohérents au fur et à mesure que le projet grossit
2. Risque de choisir un nom déjà existant et d'écraser une partie des styles

Solutions envisageables :

1. Avoir une convention de nommage (français/anglais, camelCase, BEM, etc.)
2. Préfixer les noms (.l-wrapper, .c-button)
3. Générer des noms dynamiquement (.abc23f4, .xyr421337) à l'aide d'un outil

Duplication des sélecteurs

```
.wrapper { }
.wrapper h1, .wrapper h2, .wrapper p { }
.wrapper:hover, .wrapper:focus { }
.wrapper::before { }

@media (width > 640px) {
  .wrapper { }
  .wrapper::before { }
}
```

```
.wrapper {
  & h1,
  & h2,
  & p {}

  &:hover,
  &:focus {}

  &::before {}

  @media (width > 640px) {
    &::before {}
  }
}
```

Inconvénients majeurs :

1. Difficile de trouver, renommer, déplacer, supprimer les sélecteurs

Solutions envisageables :

1. Nesting Scss pour réduire les duplications
 2. Nesting en CSS natif
- ... mais attention au nesting multiple !

4

Duplication des règles CSS

```
.button-primary {  
  display: inline-block;  
  padding: 1rem 2rem;  
  border: 1px solid hotpink;  
  background: pink;  
  cursor: pointer;  
}  
.button-secondary {  
  display: inline-block;  
  padding: 1rem 2rem;  
  border: 1px solid chocolate;  
  background: bisque;  
  cursor: pointer;  
}
```

```
.button {  
  display: inline-block;  
  padding: 1rem 2rem;  
  cursor: pointer;  
}  
.button-primary {  
  border: 1px solid hotpink;  
  background: pink;  
}  
.button-secondary {  
  border: 1px solid chocolate;  
  background: bisque;  
}
```

Inconvénients majeurs :

1. Code inutilement répété
2. Maintenabilité complexe

Solutions envisageables :

1. Rassembler les styles identiques
2. Proposer des classes (variantes) complémentaires

```
1 <!-- lien à remplacer par un button -->
2 <a class="alternate special">
3   Cliquez-moi !
4 </a>
```

```
1 /* on préfère ne rien
   supprimer */
2 a.special {color: tomato;}
3 a.alternate {color: pink;}
```

Inconvénients majeurs :

1. Le code inutilisé s'accumule au fur et à mesure que le projet grossit
2. Il pollue les feuilles de styles et les rend difficilement compréhensibles

Solutions envisageables :

1. Détection des codes inutilisés ([dans les DevTools](#))
2. Suppression des codes inutilisés (unusedCSS, PurgeCSS)

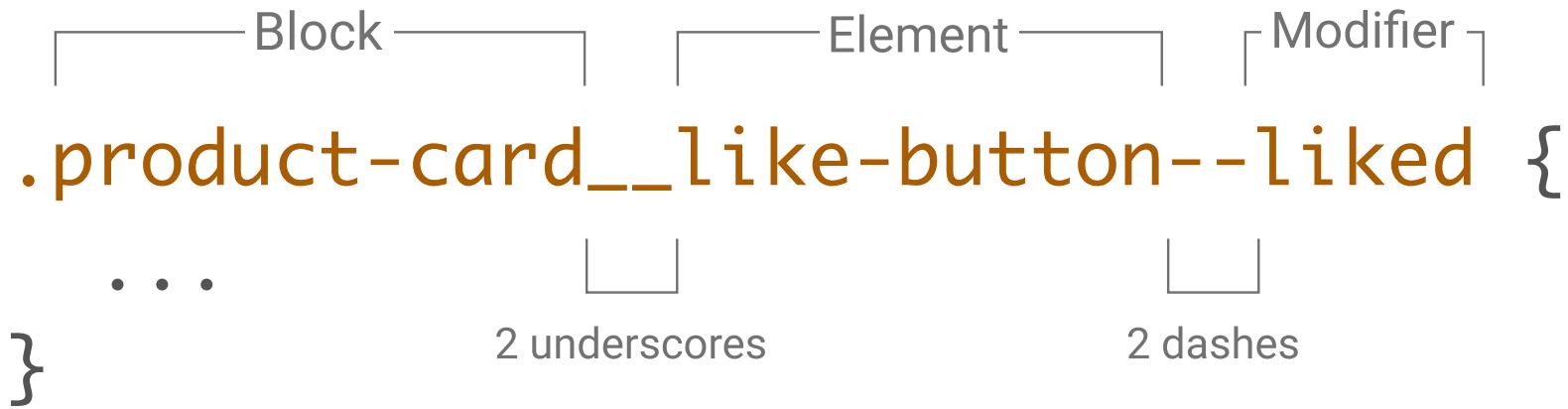
OK, alors on fait quoi ?

Les Méthodologies



la Méthodologie "BEM"

Une Convention de nommage à toute épreuve !



Block Element Modifier

.product-card__like-button--liked {

... 2 underscores 2 dashes

}



<http://getbem.com/>

Méthodologie "BEM"

Block - Element - Modifier

```
<p class="author__bio--current">
```

1. Block

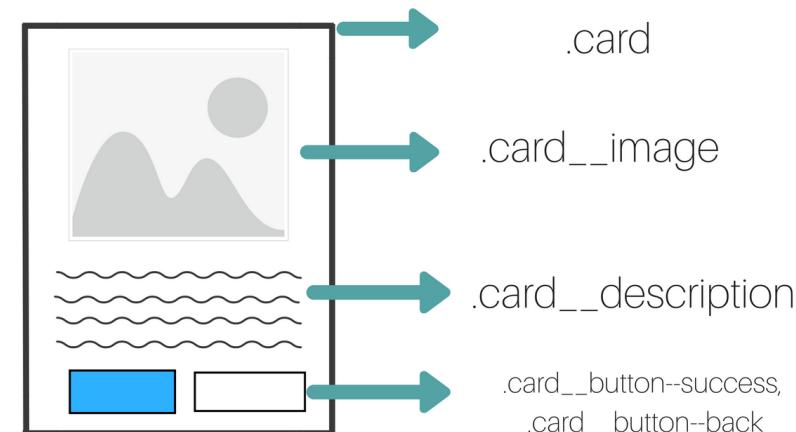
entité indépendante et autonome

2. Element

entité dépendante de son Block

3. Modifier

variante d'un Block ou d'un Element



Méthodologie "BEM"

une convention de nommage... contraignante

Nommage des blocks :

.card

.contact

.post

.main-navigation

Nommage des éléments :

Règle : block name + __ + element name

.card__title

.post__author

.post__date

Nommage des modifieurs :

Règle : block name OU element name + -- + modifier name

.card__title--disabled

.post--important

Méthodologie "BEM"

les principes essentiels

```
<p class="author__bio--current">
```

1

Tous les éléments HTML doivent chacun avoir un nom, sous forme de classe(s) CSS (pas de nommage via *id*)

2

Les sélecteurs CSS ne doivent cibler que des classes (pas de sélecteur *nav*, ni *a*, ni *ul* par exemple)

3

Les sélecteurs CSS composés sont à éviter (pas de *.menu .list*, ou de *.navigation > a*)

Frameworks "BEM" : Bootstrap (en partie), bemskel, BULMA

Méthodologie "BEM"

concrètement ?

```
<nav class="navigation">
  <ul>
    <li><a href="" class="active"></a></li>
    <li><a href=""></a></li>
    <li><a href=""></a></li>
    <li><a href=""></a></li>
  </ul>
</nav>
```

←... aheum, ça c'est pas
du BEM hein ?

OK ouais je vois ce
que tu veux dire

```
<nav class="navigation">
  <ul class="navigation_list">
    <li class="navigation_list-item"><a href="" class="navigation_link navigation_link--active"></a></li>
    <li class="navigation_list-item"><a href="" class="navigation_link"></a></li>
    <li class="navigation_list-item"><a href="" class="navigation_link"></a></li>
    <li class="navigation_list-item"><a href="" class="navigation_link"></a></li>
  </ul>
</nav>
```

La réutilisabilité dans BEM

BEM

```
1 .card
2   .card__image
3   .card__body
4     .card__title
5     .card__subtitle
6     .card__button-cta
```



Ce titre, ce sous-titre et ce bouton doivent-ils vraiment être conditionnés par leur block "card" ?

Pourquoi ne pas les rendre réutilisables ?

pas BEM

```
1 .card
2   img
3   figcaption
4   heading-3
5   text-intro
6   button-primary
```



Ce titre, ce sous-titre et ce bouton sont réutilisables dans tout le projet.

(oui mais on retombe sur le problème du nommage des éléments 😞)

(et il peut y avoir conflits de styles entre composants)



pros 🤝

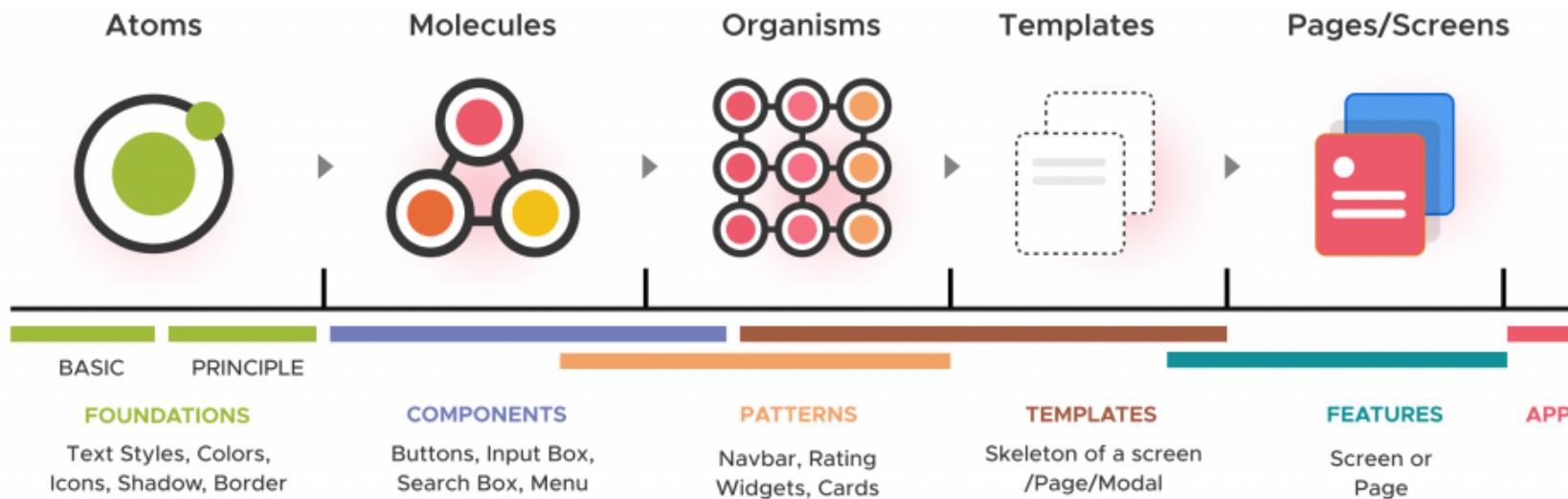
- Pas de surprises de nommage
- Pas de problème de cascade
- "Scalable" : excellente évolutivité du projet
- Écosystème et documentation riches (tutos, plugins navigateurs, etc.)
- S'intègre facilement à d'autres outils CSS (Bootstrap, Sass, etc.)

cons 😞

- Fichier HTML alourdi et complexifié en raison de (longues) classes sur chaque élément
- Pas toujours évident de trancher (Block ou Element ?)
- Problème de réutilisabilité des composants

la Méthodologie "Atomique"

l'ère des classes utilitaires !

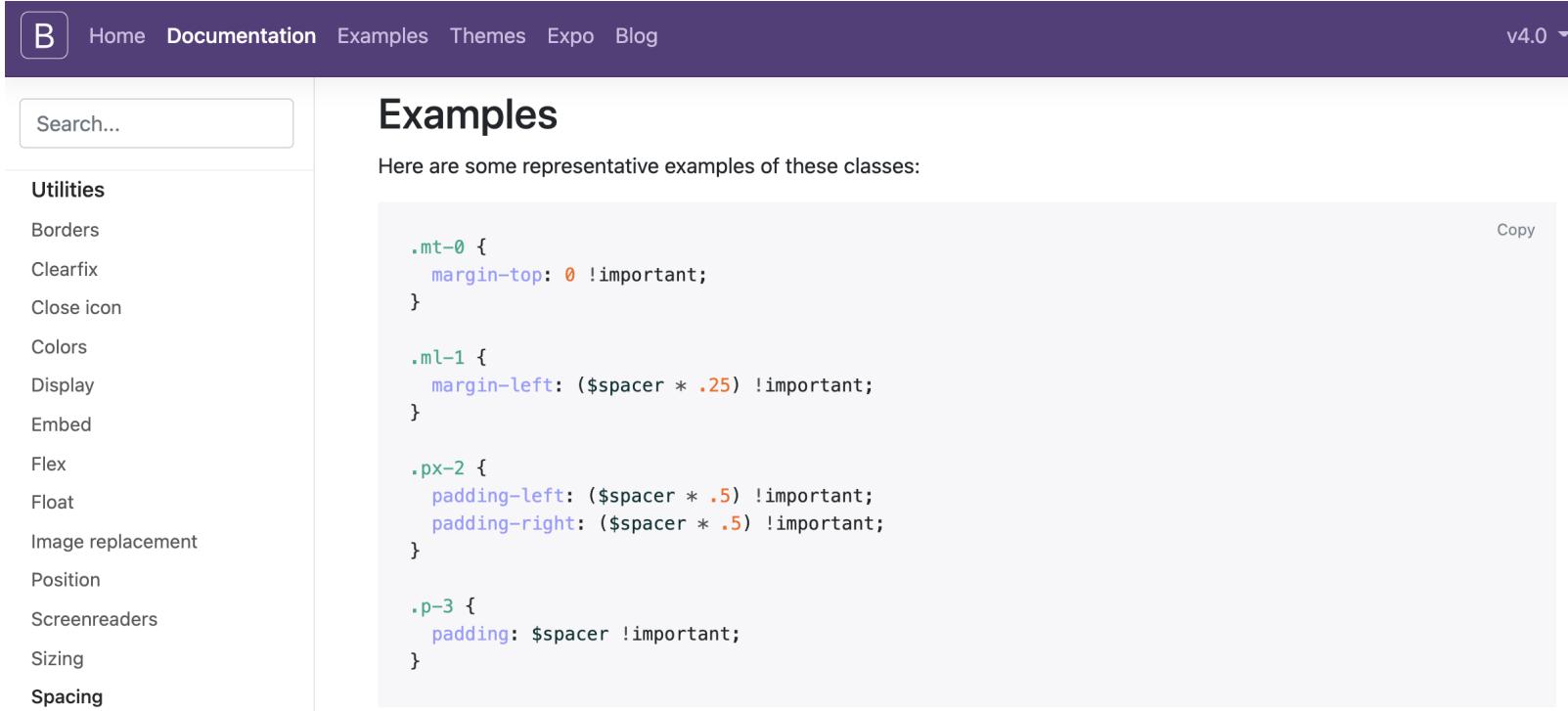


L'organisation atomique, source Laptrinhx

Méthodologie "Atomique"

les classes atomiques dans Bootstrap

```
<div class="d-block mr-1 p-2 bg-primary text-white">
```



The screenshot shows the Bootstrap 4.0 documentation website. The top navigation bar includes a logo with a stylized 'B', 'Home', 'Documentation', 'Examples', 'Themes', 'Expo', and 'Blog'. A dropdown menu 'v4.0' is also present. On the left, a sidebar titled 'Utilities' lists various classes: Borders, Clearfix, Close icon, Colors, Display, Embed, Flex, Float, Image replacement, Position, Screenreaders, Sizing, and Spacing. The main content area is titled 'Examples' and contains the following text: 'Here are some representative examples of these classes:' followed by five lines of CSS code. A 'Copy' button is located to the right of the code block.

```
.mt-0 {  
  margin-top: 0 !important;  
}  
  
.ml-1 {  
  margin-left: ($spacer * .25) !important;  
}  
  
.px-2 {  
  padding-left: ($spacer * .5) !important;  
  padding-right: ($spacer * .5) !important;  
}  
  
.p-3 {  
  padding: $spacer !important;  
}
```



<https://getbootstrap.com/docs/4.0/utilities/spacing/>

Méthodologie "Atomique"

les classes atomiques dans Tailwind

```
<div class="block mr-1 rounded bg-white md: bg-hotpink">
```

● ● ●

```
1 <div class="md:flex bg-white rounded-lg p-6">
2   <img class="h-16 w-16 md:h-24 md:w-24 rounded-full mx-auto md
3   <div class="text-center md:text-left">
4     <h2 class="text-lg">Erin Lindford</h2>
5     <div class="text-purple-500">Product Engineer</div>
6     <div class="text-gray-600">erinlindford@example.com</div>
7     <div class="text-gray-600">(555) 765-4321</div>
8   </div>
9 </div>
```

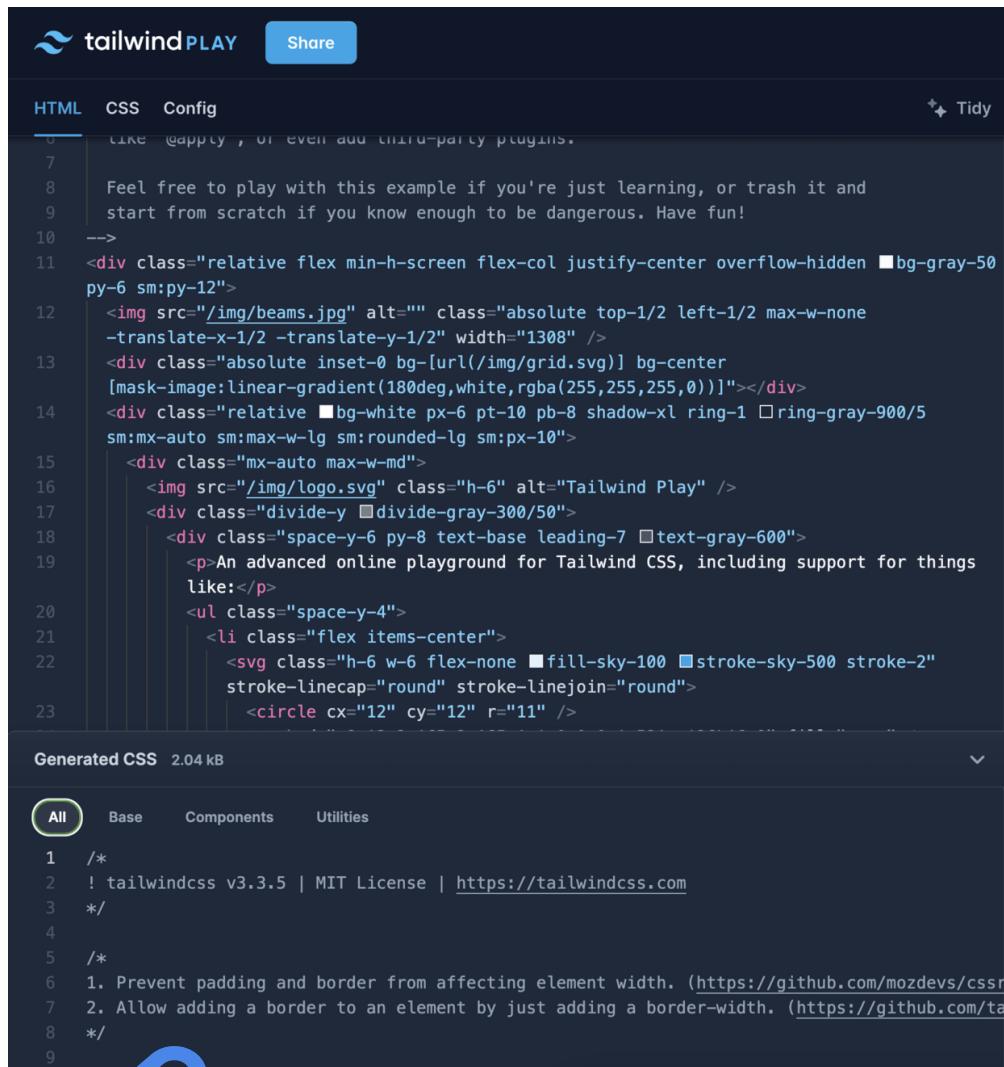


Erin Lindford
Product Engineer
erinlindford@example.com
(555) 765-4321



<https://tailwindcss.com/>

Tu veux tester?



The screenshot shows the Tailwind Play interface. At the top, there's a navigation bar with the Tailwind Play logo, a 'Share' button, and a version dropdown set to 'v3.3.5'. Below the bar is a toolbar with icons for file operations. The main area has tabs for 'HTML', 'CSS', and 'Config'. The 'HTML' tab is active, displaying a code editor with a snippet of Tailwind CSS. The code includes a grid system, a background image, and a logo. The 'Generated CSS' section shows the resulting CSS output, which is 2.04 kB. Below the code editor is a 'Generated CSS' section and a 'Utilities' tab. The 'Utilities' tab shows a list of utility classes, including ones for color, size, and spacing. The right side of the interface features a sidebar with the Tailwind Play logo and a list of features: 'Customizing your `tailwind.config.js` file', 'Extracting classes with `@apply`', and 'Code completion with instant preview'. It also includes a note: 'Perfect for learning how the framework works, prototyping a new idea, or creating a demo to share online.' At the bottom, there's a 'Want to dig deeper into Tailwind?' section with a 'Read the docs →' button.

tailwind PLAY

HTML CSS Config

Like `@apply`, or even add third-party plugins.

Feel free to play with this example if you're just learning, or trash it and start from scratch if you know enough to be dangerous. Have fun!

```
6  like @apply, or even add third-party plugins.
7
8  Feel free to play with this example if you're just learning, or trash it and
9  start from scratch if you know enough to be dangerous. Have fun!
10 -->
11 <div class="relative flex min-h-screen flex-col justify-center overflow-hidden bg-gray-50 py-6 sm:py-12">
12   
13   <div class="absolute inset-0 bg-[url(/img/grid.svg)] bg-center [mask-image:linear-gradient(180deg,white,rgba(255,255,255,0))]"></div>
14   <div class="relative bg-white px-6 pt-10 pb-8 shadow-xl ring-1 ring-gray-900/5 sm:mx-auto sm:max-w-lg sm:rounded-lg sm:px-10">
15     <div class="mx-auto max-w-md">
16       
17       <div class="divide-y divide-gray-300/50">
18         <div class="space-y-6 py-8 text-base leading-7 text-gray-600">
19           <p>An advanced online playground for Tailwind CSS, including support for things like:</p>
20           <ul class="space-y-4">
21             <li class="flex items-center">
22               <svg class="h-6 w-6 flex-none fill-sky-100 stroke-sky-500 stroke-2" stroke-linecap="round" stroke-linejoin="round">
23                 <circle cx="12" cy="12" r="11" />
```

Generated CSS 2.04 kB

All Base Components Utilities

```
1  /*
2  ! tailwindcss v3.3.5 | MIT License | https://tailwindcss.com
3  */
4
5  /*
6  1. Prevent padding and border from affecting element width. (https://github.com/mozdevs/cssremoval)
7  2. Allow adding a border to an element by just adding a border-width. (https://github.com/tailwindcss/tailwindcss)
8  */
9
```

v3.3.5

Share

Tidy

tailwind PLAY

An advanced online playground for Tailwind CSS, including support for things like:

- Customizing your `tailwind.config.js` file
- Extracting classes with `@apply`
- Code completion with instant preview

Perfect for learning how the framework works, prototyping a new idea, or creating a demo to share online.

Want to dig deeper into Tailwind?

[Read the docs →](#)



<https://play.tailwindcss.com/>

Méthodologie "Atomique"

les principes essentiels

```
<p class="block mr-1 rounded bg-white md: bg-hotpink">
```

- 1 À chaque classe CSS correspond une seule fonction
(ex. `.text-left {text-align: left}` ou `.mr-2 {margin-right: 2rem}`)
- 2 Les styles CSS sont dénués de contexte pour être totalement indépendants de la structure HTML
- 3 Les sélecteurs CSS composés n'existent pas
(pas de `.menu .list`, ou de `.navigation > a`)

Frameworks "atomiques" : Bootstrap (en partie), Tailwind, Tachyons, Windi CSS, ...



pros

- Pas de code mort
- Pas de problème de nommage
- Pas de problème de cascade
- Pas de duplication de sélecteurs
- Pas besoin de chercher du CSS dans X fichiers
- Toutes les valeurs dans un seul fichier de config
- Les classes utilitaires gèrent très finement les styles et leurs variations.
- L'outil s'adapte à différents contextes (responsive, survol/focus, dark mode, etc.).
- Écosystème riche (tutos, bibliothèques de composants, plugins navigateurs, plugins IDE : intellisense, etc.)
- Pas besoin d'autres outils CSS (Bootstrap, Sass, etc.)

cons

- Lecture du code HTML rendue fouillie et complexe (ouais, ça pique les yeux !)
- Nécessite une rigueur (guidelines) pour ne pas écrire n'importe quoi et dans n'importe quel ordre.
- Besoin d'une cheatsheet TW, parce que... *items-baseline backdrop-invert-0 md:row-start-5 sm:content-around leading-snug dark:tracking-wider*
- Impossible de trouver des éléments précis dans le code (modale, wrapper, navigation, etc.).
- Inadapté pour les layouts "complexes" (gabarits, grilles non classiques, flexbox exotique, etc.).
- Inadapté pour pas mal de fonctionnalités (transitions, animations, filtres, transformations)



Joshua

@CreeCoder

tailwind devs typing out a class name

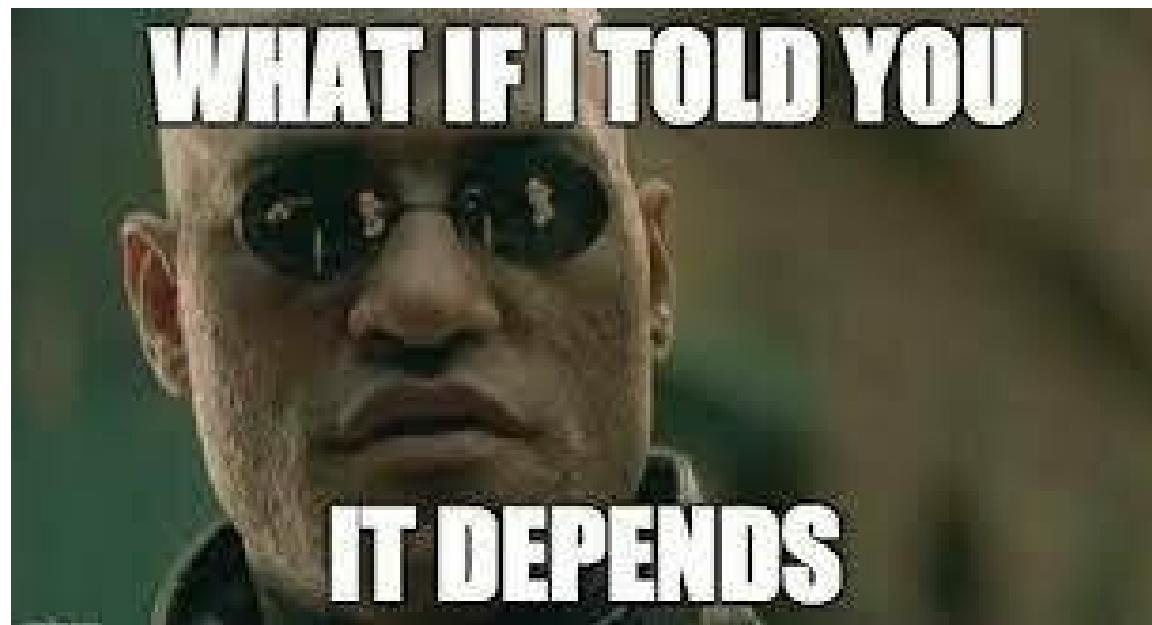
[Traduire le post](#)



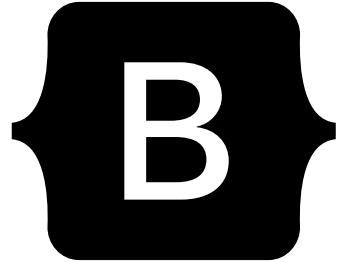
6:51 PM · 19 oct. 2024 · 79,3 k vues

Bon alors au final

on fait quoi de tout ça maintenant ?

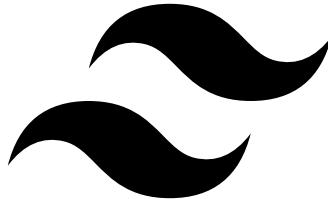


Bootstrap



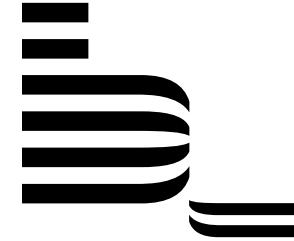
- spotify
- etsy
- fox news
- reuters
- twitter -> X
- linkedin
- snapchat
- mastercard
- duolingo
- alibaba

Tailwind (utility / atomic)



- netflix
- shopify
- twitch
- medium
- starbucks
- prestashop
- stackoverflow
- leafly
- blizzard
- github copilot

BEM



- yandex
- the guardian
- BBC
- financial times
- linkedin
- shopify
- duckduckgo
- uk.gov
- dropbox
- jetbrains

Choix du full-utilities

“ Nous utilisons TailwindCSS pour tous nos projets en front-end.

source : Guidelines CSS Alsacreations (2021)



Au bout de deux ans, notre bilan est mitigé

Concrètement : on a besoin de nommer les choses (ex. .navigation, .modal)

Choix du "ça dépend"

“ Nous utilisons les classes utilitaires uniquement lorsqu'il s'agit de propriétés liées aux espacements (margin, padding, gap), à la typographie et aux couleurs.

source : Guidelines CSS Alsacréations (2023)

⚠️ L'information est éparsillée et on ne trouve pas forcément ce qu'on cherche au bon endroit

Concrètement : *display: grid vs gap ou position vs top/right/bottom/left ou border-width vs border-color*

Choix du no-utilities

“ On n'utilise pas Tailwind par défaut, sauf si les contraintes du projet, ou le client, l'exigent.

source : [Guidelines CSS Alsacréations](#) (2025)



Pas de classes utilitaires, mais...

Concrètement : notre outillage interne permet de s'en passer



pour avoir la *class* en Alsace.

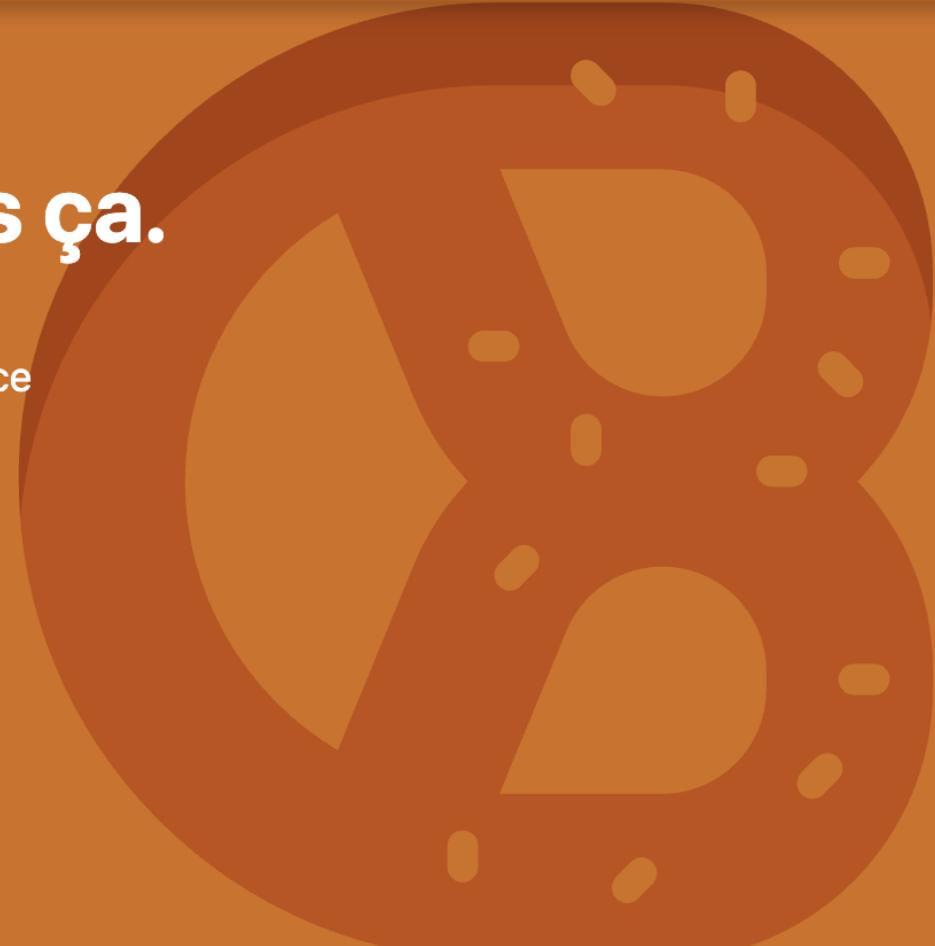
Tu vois Bootstrap ? Bah, c'est pas ça.

Bretzel est un ensemble de Layouts HTML/CSS utilitaires réunis par l'agence Alsacreations pour ses projets d'intégration.

 [Voir et récupérer nos Layouts CSS](#)

 [Télécharger la Cheatsheet \(PDF 700ko\)](#)

 [Installer l'extension Visual Studio Code](#)





tailwindcss

alsacréations 

Reset CSS	<code>@import "tailwindcss/preflight.css"; layer(base);</code>	<code>@import "reset.css" layer(config);</code>
Tailles de polices	<code>text-4 md:text-6 lg:text-8</code>	<code>--text-m (variables avec clamp())</code>
Espaces, gouttières	<code>mt-2 md:mt-4</code>	<code>--spacing-s (variables avec clamp())</code>
Couleurs (dark mode)	<code>bg-hotpink dark:bg-pink</code>	<code>--on-surface (variables avec light-dark())</code>
Layouts	<code>grid grid-cols-2 gap-4</code>	<code>data-layout="" (layouts HTML)</code>

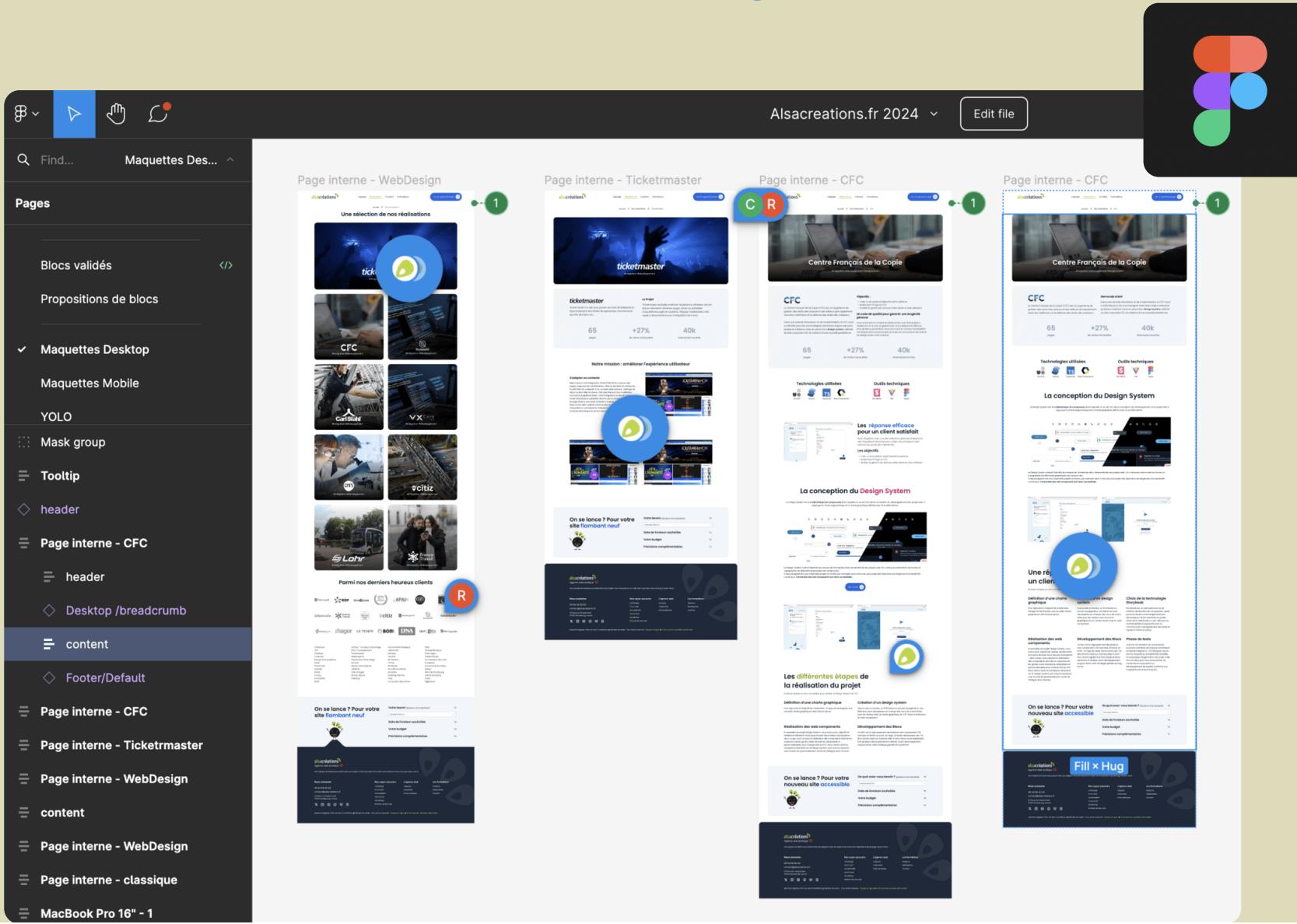


 Reset CSS

 Primary
Pour starter ses projets avec style.

du design à l'intégration

UI-Kit + bonne webdesigneuse = ❤️



The image shows a screenshot of a Figma UI-Kit for web design. The interface includes a sidebar with navigation for 'Pages', 'Blocs validés', 'Propositions de blocs', and a 'Maquettes Desktop' section. The 'Maquettes Desktop' section contains sub-sections for 'Maquettes Mobile', 'YOLO', 'Mask group', 'Tooltip', 'header', and specific pages for 'Page interne - CFC', 'Page interne - Ticketmaster', and 'Page interne - WebDesign'. Each page is shown with a preview and a blue circular icon with a green dot and a red border. The main workspace displays four internal pages: 'Page interne - WebDesign', 'Page interne - Ticketmaster', 'Page interne - CFC', and 'Page interne - CFC' (duplicated). The 'Page interne - CFC' pages show a detailed design system section with various components and steps. The top right corner of the Figma interface features the Figma logo.

UI-Kit indispensable

afficher des specs dans l'UI-Kit

The image shows a UI-Kit documentation page with a dark header and a light background. On the left, a tooltip component is detailed with a purple border and rounded corners. It contains a title 'Dev Note' with a 'Size' dropdown set to '1920px'. The 'Content' section includes a note about max-width: 800px. The 'Shapes' section shows a gray rounded rectangle with a 'Shape / Geometry' description: 'Colors: gray,800', 'Opacity: 5%', and 'Background-image: repeat'. The 'Optionnel' section lists three items: 1. Possibilité de mettre une image, un médaillon ou un avatar; 2. Fond de couleur unis avec ou sans élément graphique → laisser le choix; and 3. Button CTA. To the right, three mail templates are shown with dashed lines pointing to specific UI elements. The first template, 'Footer citation', has a 'Size' dropdown set to '1440px' and a 'Footer citation' section. The second template has a 'Size' dropdown set to '768px' and a 'Footer citation' section. Both templates include a 'Button CTA' button. The third template has a 'Size' dropdown set to '1920px' and a 'Footer citation' section. It also includes a 'Button CTA' button.

UI-Kit indispensable

afficher des specs dans l'UI-Kit

Tooltip

Dev Note

Content

1 Max-width : 800px

Shapes

1 Shape / Geometry :
Colors: gray,800
Opacity : 5%
Background-image: repeat

Optionnel

1 Possibilité de mettre une image, un médaillon ou un avatar

2 Fond de couleur unis avec ou sans élément graphique → laisser le choix.

3 Button CTA

Footer citation

1920px

1440px

768px

Options : Possibilité de mettre une image, un médaillon ou un avatar

Option : Bouton

• Fond de couleur unis avec ou sans élément graphique

Un mail par semaine des informations pour savoir partager les savoirs.

Button CTA >

Un mail par semaine avec l'essentiel des informations pour savoir partager les savoirs.

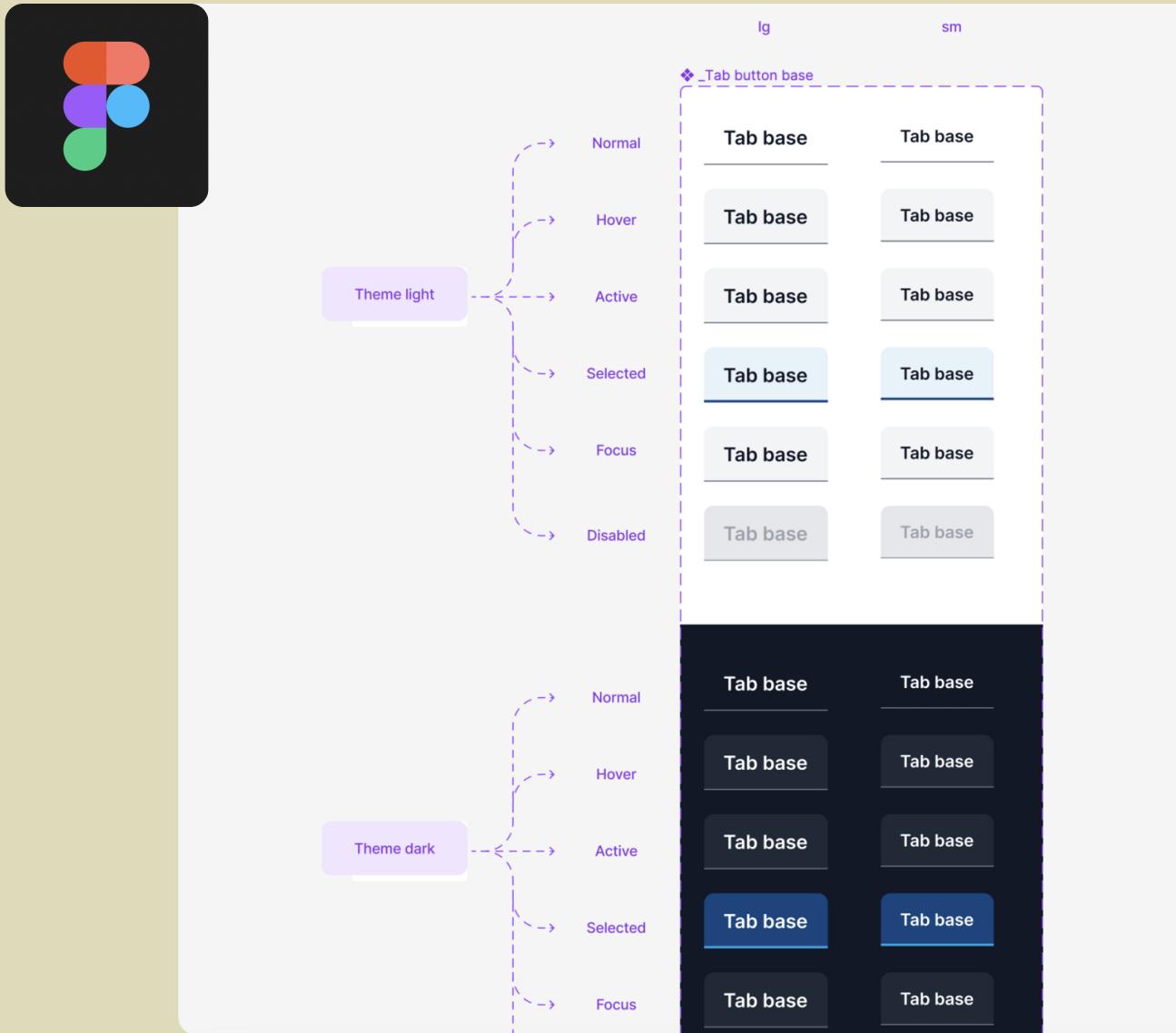
Button CTA >

Un mail par semaine avec l'essentiel des informations pour savoir partager les savoirs.

Button CTA >

UI-Kit *complet*

Penser à présenter toutes les variantes dans l'UI-Kit



Des variables dès la conception



ces variables seront transmises au développeurs

Primitives ... □ Beta X

All variables	64	Name	Light	+
colors		⌚ 50		F9FAFB
gray		⌚ 100		F3F4F6
blue		⌚ 200		E5E7EB
teal		⌚ 300		D1D5DB
green		⌚ 400		9CA3AF
red		⌚ 500		6B7280
violet		⌚ 600		4B5563
yellow		⌚ 700		374151
orange		⌚ 800		1F2937
spacing		⌚ 900		111827

Primitives ... □ Beta X

All variables	64	Name	Light	+
colors		# 0	0	+
gray		# 1	4	
blue		# 2	8	
teal		# 3	12	
green		# 4	16	
red		# 5	20	
violet		# 6	24	
yellow		# 8	32	
orange		# 10	40	
spacing		# 12	48	
rounded		# 14	56	
		# 16	64	
		# 20	80	
		# 24	96	

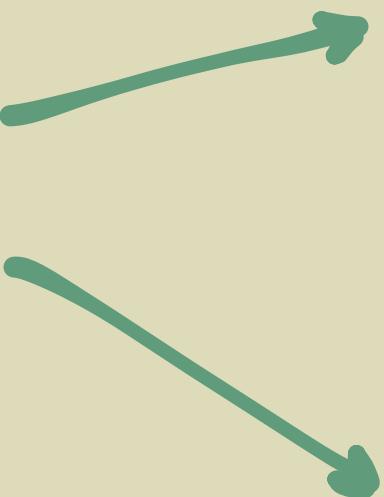
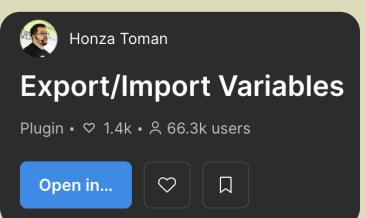
Create variable

Figma vers CSS

Figma



Plugin Export/Import
Variables

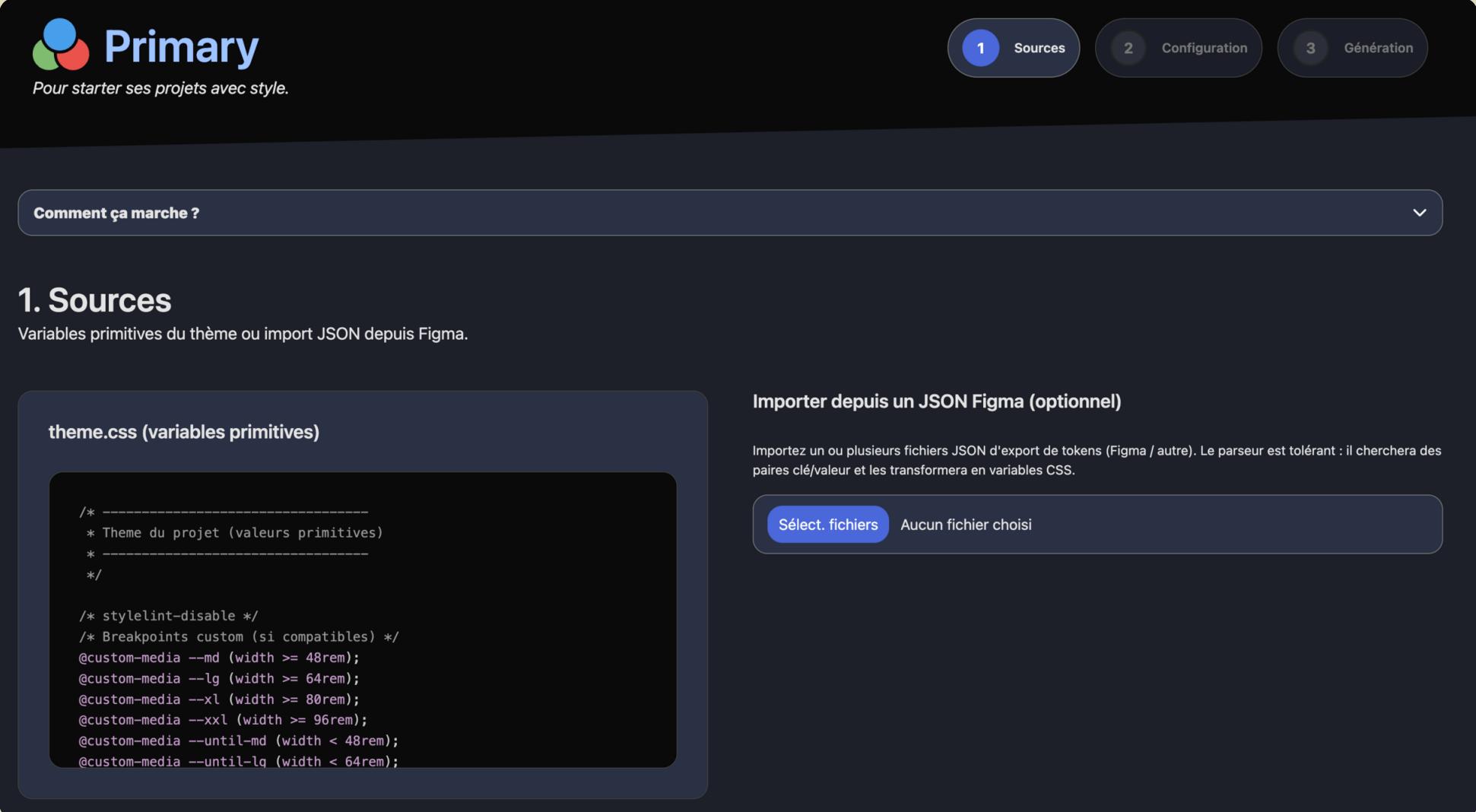


```
1  /* -----
2   * Theme du projet (valeurs primitives)
3   * -----
4  */
5
6  /* stylelint-disable */
7  /* Breakpoints custom (si compatibles) */
8  @custom-media --md (width >= 48rem);
9  @custom-media --lg (width >= 64rem);
10 @custom-media --xl (width >= 80rem);
11 @custom-media --xxl (width >= 96rem);
12 @custom-media --until-md (width < 48rem);
13 @custom-media --until-lg (width < 64rem);
14 @custom-media --until-xl (width < 80rem);
15 @custom-media --until-xxl (width < 96rem);
16 /* stylelint-enable */
17
18 :root {
```

```
1  /* -----
2   * Theme-tokens, généré par primary.alsacreations.com
3   * Surcouche de theme.css
4   * Configuration :
5   * - Couleur primaire : blue
6   * - Theme : light et dark
7   * - Typographie responsive : oui
8   * - Espacements responsive : oui
9   * -----
10 */
11 :root {
12   color-scheme: light dark;
13
14   &[data-theme="light"] {
15     color-scheme: light;
16   }
17
18   &[data-theme="dark"] {
```

Figma vers CSS

primary.alsacreations.com



The screenshot shows the Primary theme editor interface. At the top, there is a navigation bar with three tabs: 'Sources' (highlighted in blue), 'Configuration' (grey), and 'Génération' (grey). Below the navigation bar, there is a header with the Primary logo and the tagline 'Pour starter ses projets avec style.' A 'Comment ça marche ?' button is visible on the left, and a dropdown arrow is on the right. The main content area is divided into two sections: '1. Sources' and 'Importer depuis un JSON Figma (optionnel)'. The 'Sources' section contains a code editor for 'theme.css (variables primitives)' with the following content:

```
/* -----
 * Thème du projet (valeurs primitives)
 * -----
 */

/* stylelint-disable */
/* Breakpoints custom (si compatibles) */
@custom-media --md (width >= 48rem);
@custom-media --lg (width >= 64rem);
@custom-media --xl (width >= 80rem);
@custom-media --xxl (width >= 96rem);
@custom-media --until-md (width < 48rem);
@custom-media --until-lg (width < 64rem);
```

The 'Importer depuis un JSON Figma (optionnel)' section contains a button labeled 'Sélect. fichiers' and the text 'Aucun fichier choisi'.

kiwikipedia

ressources et guidelines Alsacréations

README

Kiwikipedia, made by Alsacréations 🍋

Kiwikipedia est une base de connaissance technique librement partagée et dont les objectifs sont:

- D'uniformiser et d'harmoniser les processus de conception de l'agence web Alsacreations.fr.
- De favoriser l'intervention de profils de compétences variés au sein d'une équipe.
- De faciliter la maintenance des projets.

💡 Le Web est un métier de plus en plus tentaculaire, comportant un vaste ensemble de thématiques qui vont avoir au final un impact sur le succès auprès de l'internaute: l'accessibilité, la qualité, l'ergonomie, la performance, le référencement (SEO), la maintenabilité. Tous les critères ne seront pas toujours applicables et le seront bien souvent en fonction du temps alloué.

Cette présente base de connaissances techniques est le fruit de note veille technologique quotidienne et se compose de:

- Guidelines (bonnes pratiques internes concernant divers langages et technologies web)
- Ressources (documentations diverses, checklists, cheatsheets)
- Starters (tutoriels rapides pour initier des projets)
- Configs (fichiers de configuration tels que `.editorconfig`, `stylelint.config.js`, `tsconfig.json`, `settings.json` pour VScode)



Contributors 11



Deployments 11

✓ [github-pages](#) 2 years ago
+ 10 deployments

OK, la suite ?

et sois en forme !



*Bring on the
next challenge.*

THE
NEXT
STEP