

CSS

Tu peux pas test !

Le CSS et la qualité

la vérité qui fâche

Le CSS, ce n'est pas simple !



“

CSS is like a bear cub.

– *Andres Galante*

Le CSS, ce n'est pas simple !

☾ It's almost a challenge to find a development team that's working on a codebase that's more than a couple of years old where the CSS isn't the most frightening and hated part of that system.

– Andy Hume, [CSS for Grownups](#)

Qui suis-je ?

Thomas Zilliox.

Je suis un expert CSS indépendant.

Je sais produire du code CSS maintenable...

...au moins par moi-même !

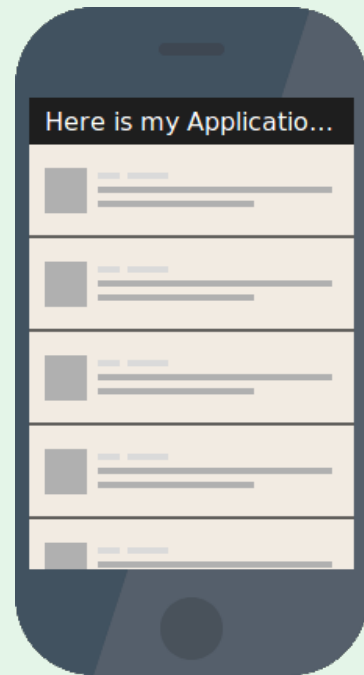
Et après ?

Le bug CSS typique

Celui responsable d'au moins 18 tickets JIRA

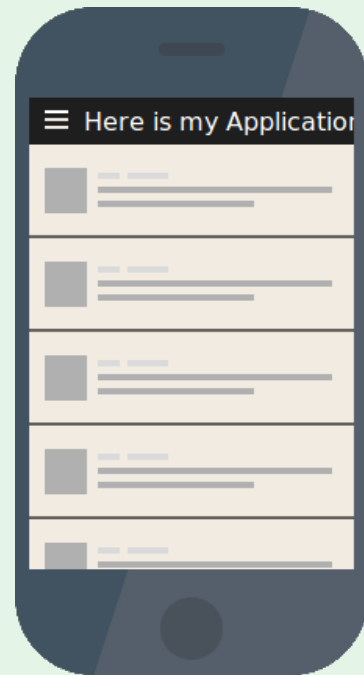
Le bug CSS typique

```
.header {  
  padding: 0.5em 1em;  
  overflow: hidden;  
  background: black;  
  color: white;  
  white-space: nowrap;  
  text-overflow: ellipsis;  
}
```



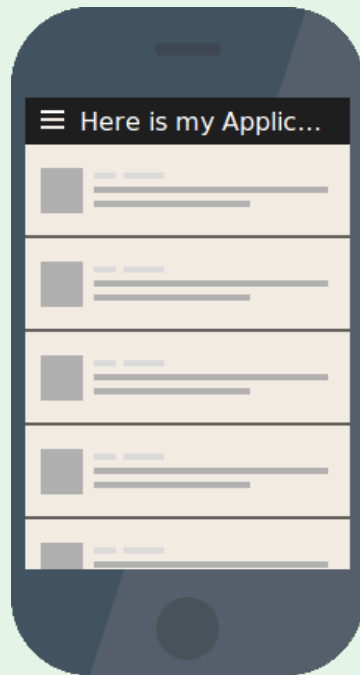
Le bug CSS typique

```
.header {  
  padding: 0.5em 1em;  
  /* overflow: hidden; */  
  background: black;  
  color: white;  
  white-space: nowrap;  
  text-overflow: ellipsis;  
}
```



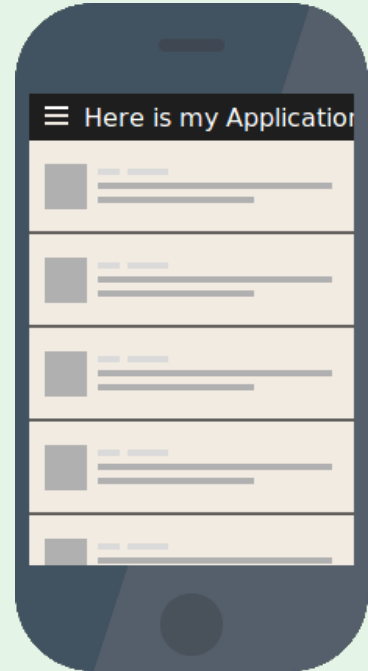
Le bug CSS typique

```
.header {  
  [...]  
}  
  
.ellipsis {  
  overflow: hidden;  
  white-space: nowrap;  
  text-overflow: ellipsis;  
}
```



Le bug CSS typique

```
.header {  
  display: inline-block;  
}  
.ellipsis {  
  overflow: hidden;  
  white-space: nowrap;  
  text-overflow: ellipsis;  
}
```



Le bug CSS typique

```
.header { [...] }
```

```
.ellipsis {  
  display: block;  
  overflow: hidden;  
  white-space: nowrap;  
  text-overflow: ellipsis;  
}
```



Le bug CSS typique

```
.header { [...] }  
.home-item { display: flex }  
.ellipsis {  
  display: block;  
  overflow: hidden;  
  white-space: nowrap;  
  text-overflow: ellipsis;  
}
```



Le bug CSS typique

```
body {  
  word-wrap: break-word;  
}  
  
.ellipsis {  
  overflow: hidden;  
  white-space: nowrap;  
  text-overflow: ellipsis;  
}
```

Le bug CSS typique

```
body {  
  word-wrap: break-word;  
}  
.ellipsis {  
  overflow: hidden;  
  white-space: nowrap;  
  text-overflow: ellipsis;  
}
```



Les sources de la complexité

Résumons !

La complexité du CSS

- 355 propriétés dans CSS3
- Une infinité de déclarations par sélecteurs
- Une infinité de sélecteurs par projet




La complexité du CSS

-  5 545 sélecteurs 9 789 déclarations
-  4 525 sélecteurs 7 443 déclarations
-  3 280 sélecteurs 6 655 déclarations
-  2 511 sélecteurs 4 811 déclarations
-  1 447 sélecteurs 2 443 déclarations

La complexité du HTML

- Une infinité de sélecteurs CSS par élément HTML
- Une infinité d'éléments HTML par sélecteurs CSS
- Des propriétés héritées des noeuds HTML parents

La complexité du HTML

-  4 052 éléments HTML
-  3 776 éléments HTML,
-  3 492 éléments HTML
-  2 751 éléments HTML
-  1 791 éléments HTML

La complexité des navigateurs

- Une infinité de configurations visiteurs possibles : appareils, OS, navigateurs, extensions, etc.
- Chaque 6 semaines, une version de Chrome et Firefox
- Du code qui a pu connaître d'autres temps

La complexité de l'intégration

- CSS
- HTML
- Les navigateurs
- ET?

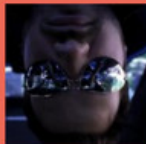
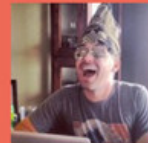
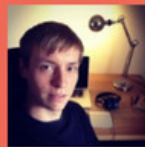
La complexité humaine

people who touched css



La complexité humaine

css developers



Quelle que soit l'énergie que vous y investirez,
un niveau de qualité n'est jamais un acquis en CSS !

Tester du code

Ceux qui nous préviennent quand on fait une bourde !

Tester du code

Ce problème a été résolu du côté du code algorithmique :

- Tests unitaires
- Tests fonctionnels
- Analyseur de code statique (linter)

Tester du code



All checks have passed

[Hide all checks](#)

4 successful checks



Functional tests — Success!



Unit tests — Success!



Linters — Success!



Deployment — Success!

Tester du code

Sur 6play.fr :

- 1500+ tests unitaires
- 2300+ étapes de tests fonctionnels

Tester du CSS

C'est à partir de maintenant que c'est intéressant !

Tester du CSS

Les revues de code c'est bien !

Elles permettent de prévenir la majorité des erreurs.

Tester du CSS

<pre>@include at-media('<medium') { - display: block; - visibility: visible; - opacity: 1; top: 0; - width: calc(100% - 51px); - max-width: none; height: 100vh; - transform: translateX(100%); - padding-bottom: 130px; overflow-y: auto; -webkit-overflow-scrolling: touch; background-color: \$common-gray-2; }</pre>	<pre>20 @include at-media('<medium') { 21 top: 0; 22 + left: 100%; 23 + right: auto; 24 + width: calc(100vw - 51px); 25 height: 100vh; 26 + 27 overflow-y: auto; 28 -webkit-overflow-scrolling: touch; 29 + 30 background-color: \$common-gray-2; 31 + visibility: visible; 32 + opacity: 1; 33 }</pre>
--	--

Tester du CSS

Sauf que c'est illisible !

Comment rendre la revue de code de CSS efficace ?

Tester du CSS




Some checks were not successful

[Hide all checks](#)

1 failing and 4 successful checks



 **Functional tests broken on Edge 14** — Failure!



Functional tests on other browsers — Success!



Unit tests — Success!



Linters — Success!



Deployment — Success!

Tester du CSS



Some checks were not successful

[Hide all checks](#)

2 failing and 4 successful checks



 **Functional tests broken on Edge 14** — Failure!



 **Functional tests broken on mobile resolutions** — Failure!



Functional tests on other browsers and resolutions — Success!



Unit tests — Success!



Linters — Success!



Deployment — Success!

Tester du CSS



All checks have passed

[Hide all checks](#)

5 successful checks



8 pages contain visual differences — Manually validated by @tzi

[See differences](#)



Functional tests on other browsers — Success!



Unit tests — Success!



Linters — Success!



Deployment — Success!

Tester du CSS

Le style guide permet de documenter les composants :

- Aux designers
- Aux développeurs

Tester du CSS

AtlasKit

← Back

All components


COMPONENTS

- Analytics
- Avatar
- Badge
- Banner
- Blanket
- Breadcrumbs
- Button


Examples

Avatar

CIRCLE



SQUARE



The image shows a preview of the 'Avatar' component in AtlasKit. It is divided into two sections: 'CIRCLE' and 'SQUARE'. The 'CIRCLE' section displays five circular avatars of decreasing size from left to right. The largest is a light gray circle with a white person icon. The second is a smaller light gray circle with a white person icon and a small green dot at the bottom right. The third is a very small light gray circle with a white person icon and a small blue dot at the bottom right. The fourth is a very small light gray circle with a white person icon and a small red dot at the bottom right. The fifth is a very small light gray circle with a white person icon. The 'SQUARE' section displays five square avatars of decreasing size from left to right. The largest is a light gray square with a white boat icon. The second is a smaller light gray square with a white boat icon and a small green checkmark at the top right. The third is a very small light gray square with a white boat icon and a small red 'x' at the top right. The fourth is a very small light gray square with a white boat icon and a small blue dot at the top right. The fifth is a very small light gray square with a white boat icon.

Tester du CSS

Il permet aussi de les visualiser sans leur contexte d'utilisation.

C'est un excellent bac à sable.

C'est l'équivalent des TUs!

Tester du CSS



Some checks were not successful

[Hide all checks](#)

1 failing and 4 successful checks



2 components are different from master — You must validate manually

[See differences](#)



Functional tests on other browsers — Success!



Unit tests — Success!



Linters — Success!



Deployment — Success!

Un seul bémol

Il y a un seul bémol sur ces tests automatiques...

Un seul bémol

Il s'agit de notre objectif !

Tester du CSS aujourd'hui

et pour toutes les tailles de projet stp.

Captures d'image

Les différentiel de capture d'écrans existent :

- [Gemini](#) – basé sur Selenium, open-source
- [WebdriverCSS](#) – plugin webdriver, open-source
- [Backtrac.io](#) – crawler, saas
- [Percy.io](#) – GitHub CI, saas

Captures d'image

Mais ce n'est pas utilisable :

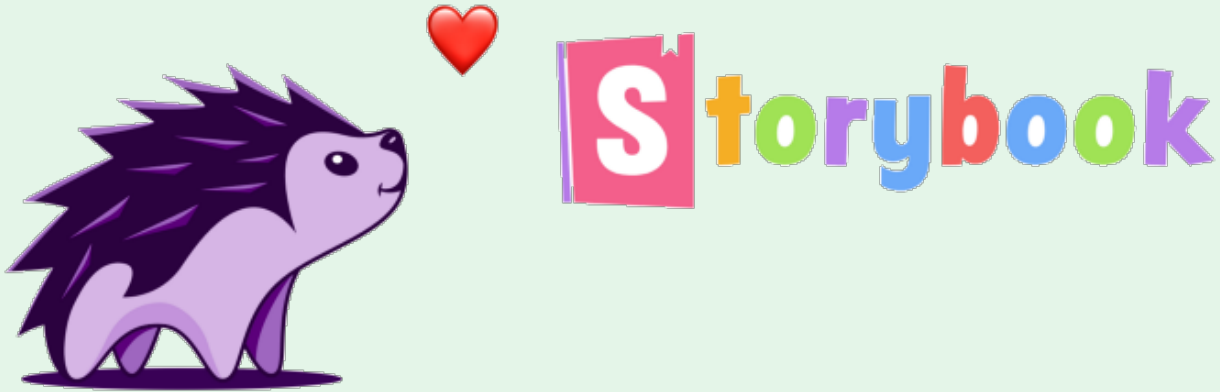
- Sur tous les navigateurs
- Sur du contenu dynamique
- Sur des pages entières

Captures d'image

Pour être pérenne, il faudrait le coupler avec un style guide.

Encore faut-il en avoir un !

Captures d'image



Tests sur différents navigateurs

Les tests automatisés sur device sont possibles :

- [BrowserStack](#)
- [Saucelabs](#)

Tests sur différents navigateurs

Les tests fonctionnels peuvent contenir des tests graphiques.

Vérifier qu'un élément :

- est visible
- est dans le viewport
- ne dépasse pas de son parent

Tests sur différents navigateurs

Par contre c'est très long !

Impossible de valider tous les navigateurs sur toutes les PRs

Les linters

Les linters et formateurs CSS et Scss existent :

- stylelint.io – prévient les erreurs de syntaxes
- csscomb.com – ordonnancement des propriétés
- [prettier](https://prettier.io) – formateur JS également

Les linters

```
visual.css
2:12 ✘ Unexpected invalid hex color "#4f"           color-no-invalid-hex
4:1  ⚠ Expected ".foo.bar" to have a specificity no more than "0,1,0" selector-max-specificity
6:13 ✘ Unexpected unit "px" for property "margin"    declaration-property-unit-blacklist
7:17 ✘ Expected single space after "," in a single-line function function-comma-space-after
```

Les linters

Bemlinter permet de valider sur CSS et SCSS :

- La syntaxe BEM
- L'isolation des composants

Les linters

```
X citron / app-header
[app-header.scss:477] Error: ".user-label__fullname" is incoherent with the file name, ".app-header" expected.
bemlinter has detected 651 errors on 194 blocks.
FAIL: bemlinter has detected 1 new error.
```

Les linters

Première tentative :

- Analyser tous les fichiers
- Autoriser une blocklist

Les linters

Deuxième tentative :

- Enregistrer les erreurs existantes
- Alerter sur les nouvelles erreurs

Conclusion

1. Ajoutez un linter et formateur
2. Ajoutez bemlinter
3. Ajoutez des contrôles visuels à vos TFs
4. Jouez vos TFs sur plusieurs navigateurs
5. Partagez un style guide

Mais ce ne sont pas des tests !?

En même temps c'était dans le titre de la conf.

Et le côté humain ?

1. Évitez les excès de confiance
2. Adoptez une méthodologie
3. Formez toute l'équipe aux CSS
4. Aimez vos intégrateurs <3

Merci !



Des questions ?

@iamtzi

tzi.fr/slides/pw2017

tzi.fr/slides/pw2017.pdf

