CSS Tu peux pas test !

#Codeurs2017

Avant de commencer…

Les slides sont disponibles :

https://tzi.fr/slides/codeurs2017

https://tzi.fr/slides/codeurs2017.pdf

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 [...] where the CSS isn’t the most hated part of that system.
>
> — *Andy Hume*

Qui suis-je ?

Thomas Zilliox, expert CSS indépendant.

Je sais produire du code CSS maintenable…

…au moins par moi-même !

Qui suis-je ?

Et du code maintenable avec une équipe de 12 ?

Le bug CSS typique

Celui responsable d'au moins 18 tickets JIRA

Le bug CSS typique

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

Le bug CSS typique

```css
.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

```css
.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

- 501 propriétés CSS
- Une infinité de déclarations par sélecteurs
- Une infinité de sélecteurs par projet

w3c

La complexité du CSS

- Le Monde.fr — 5 545 sélecteurs 9 789 déclarations
- LE FIGARO.fr — 4 525 sélecteurs 7 443 déclarations
- L'EXPRESS.fr — 3 280 sélecteurs 6 655 déclarations
- 20 minutes.fr — 2 511 sélecteurs 4 811 déclarations
- Le nouvel Observateur — 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
- Le Monde.fr 3 776 éléments HTML,
- LE FIGARO.fr 3 492 éléments HTML
- L'EXPRESS.fr 2 751 éléments HTML
- Le nouvel Observateur 1 791 éléments HTML

La complexité des navigateurs

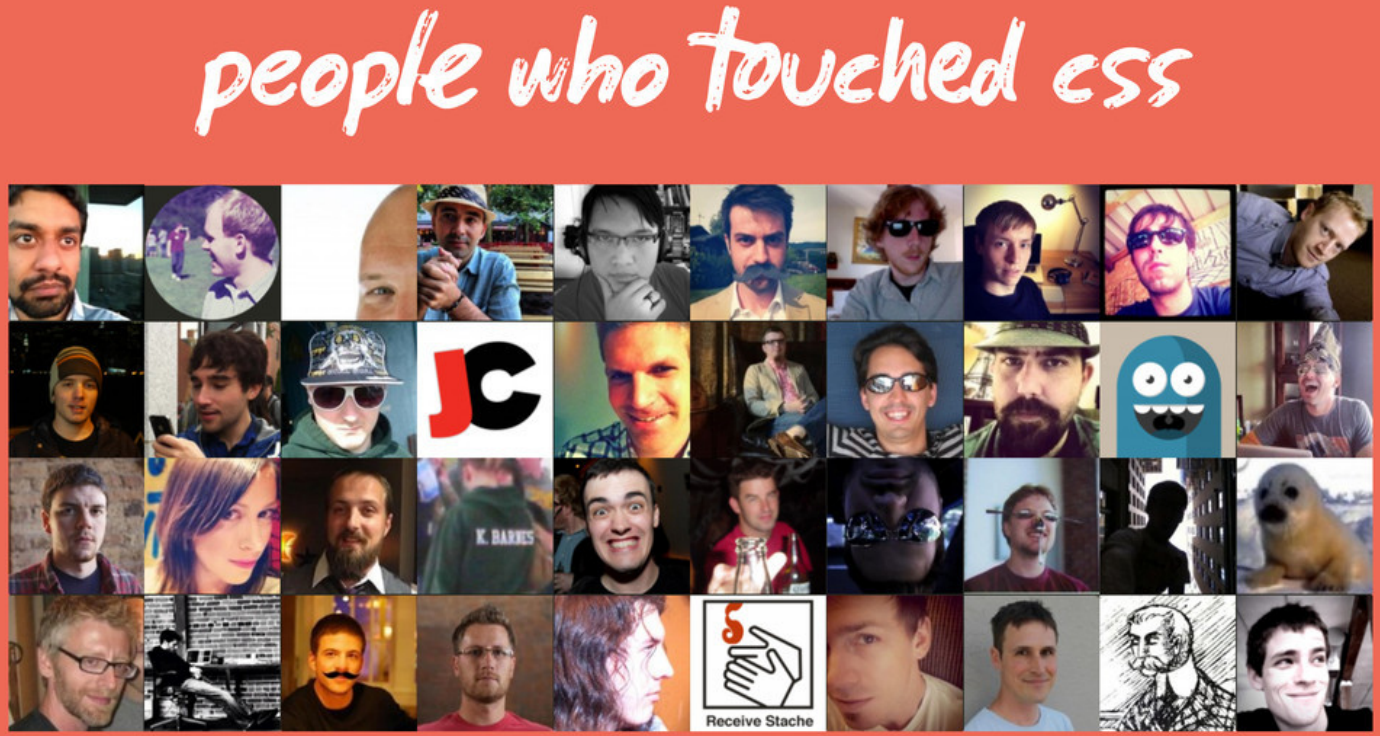
- Une infinité de configurations visiteurs possibles : appareils, OS, navigateurs, extensions, etc.
- Chaque 6 semaines, une version de Chrome et Firefox
- Du code qui a pu connaître d'autres temps

Device Detector by Piwik

La complexité de l'intégration

- CSS
- HTML
- Les navigateurs
- ET ?

La complexité humaine



La complexité humaine

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

Et pour les autres langages ?

en assumant que CSS est un vrai langage

Et pour les autres langages ?

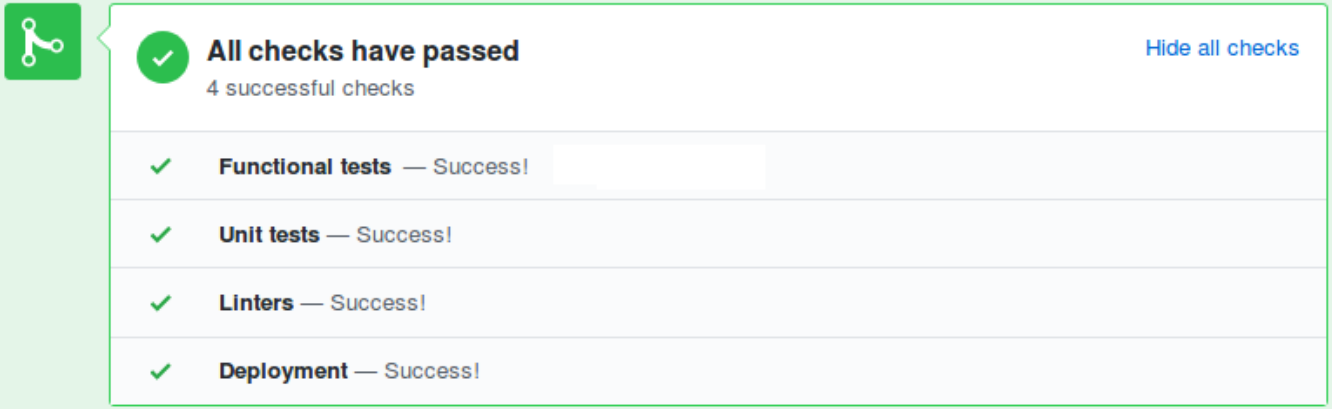
Ce problème a été résolu du côté du code algorithmique :

- Revues de code (PR)
- Tests unitaires (TU)
- Tests fonctionnels (TF)
- Analyseur de code statique (linter)

Et pour les autres langages ?

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

Et pour les autres langages ?



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

```
    @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;



    }
```

```
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      }
```

Tester du CSS

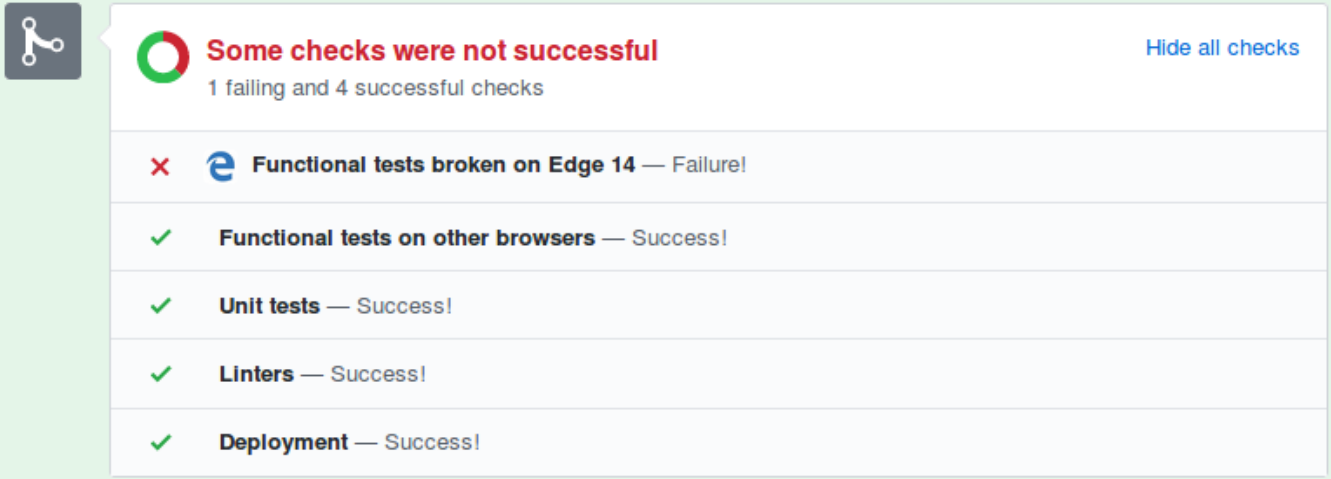
Sauf que c'est très difficile de relire du CSS !

Comment faciliter la relecture ?

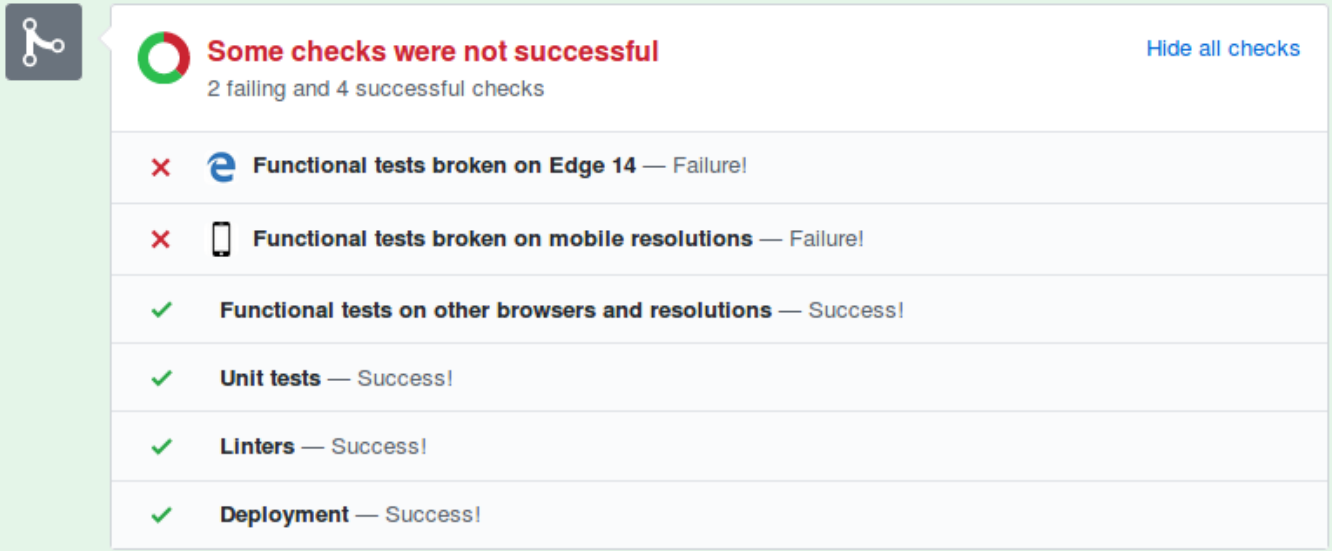
Tester du CSS

1 - Faire des bilans automatiques

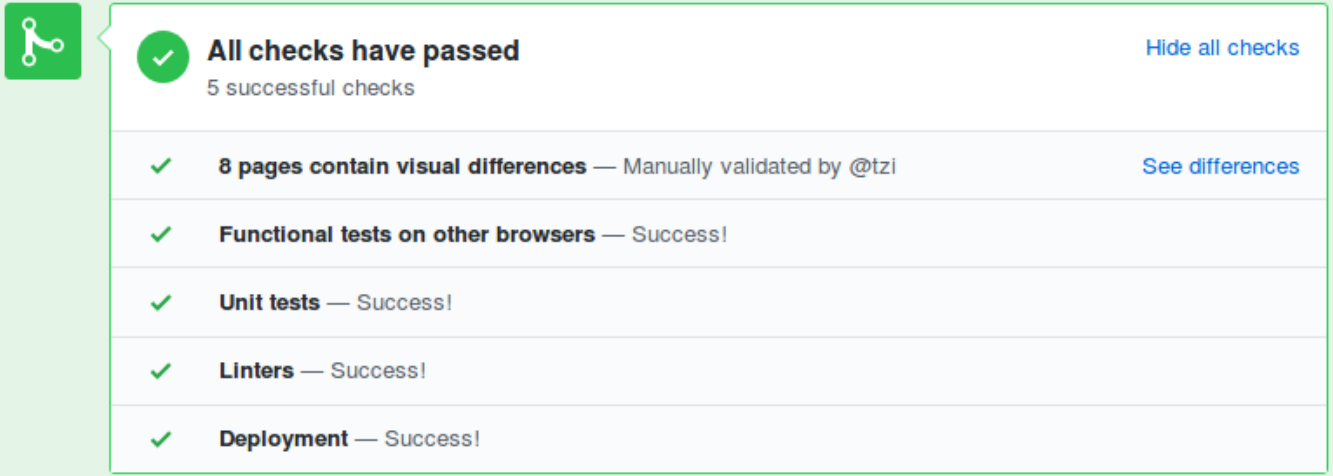
Tester du CSS



Tester du CSS



Tester du CSS



Tester du CSS

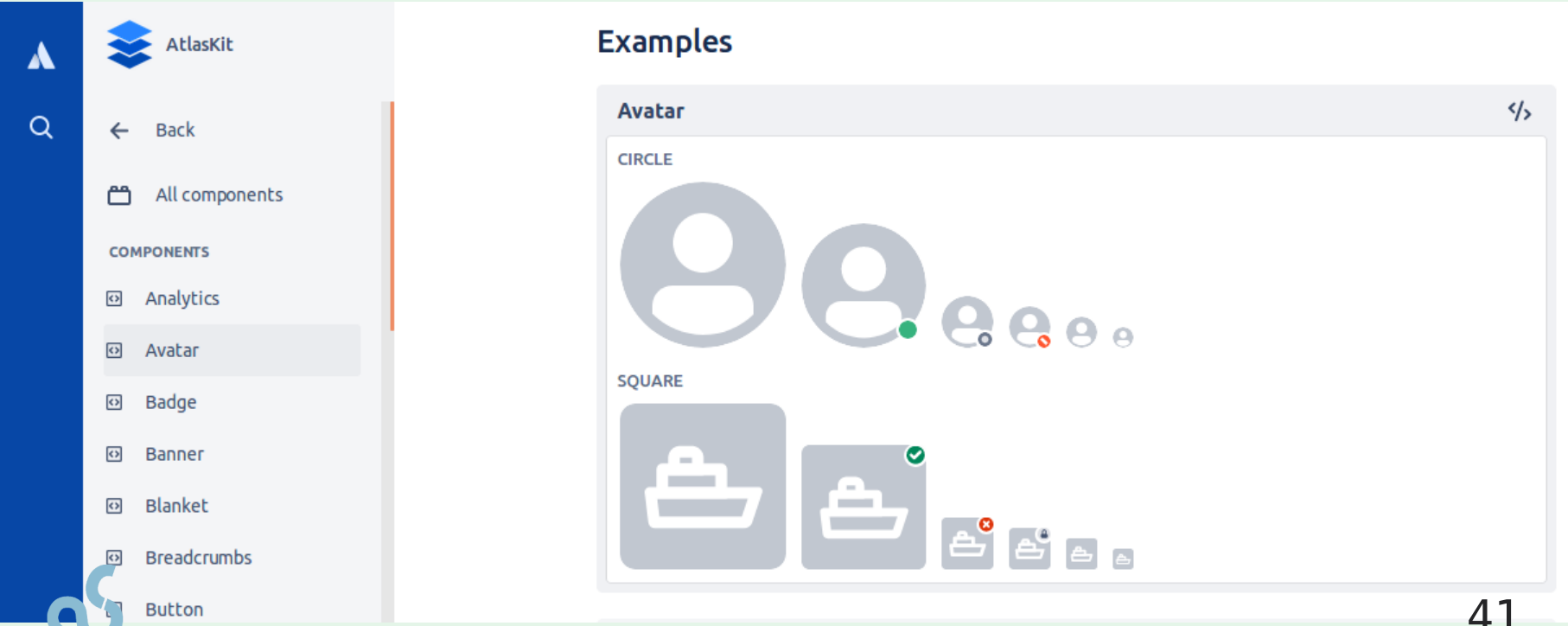
2 - Isoler les développements

Tester du CSS

Le style guide permet de documenter les composants :

- Aux designers
- Aux développeurs

Tester du CSS



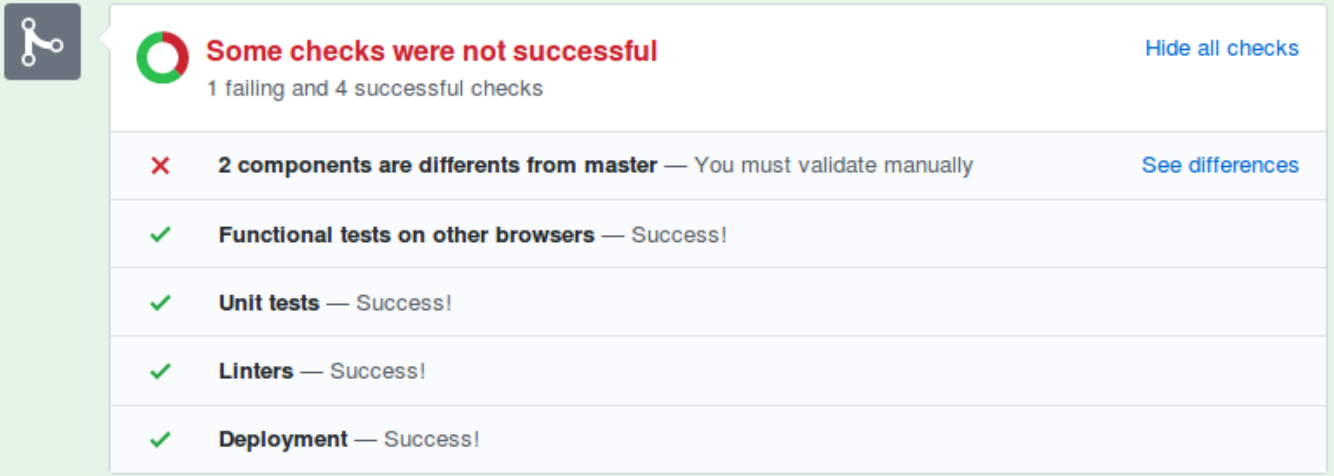
Tester du CSS

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



Tester du CSS

Un seul bémol sur ces techniques…

Tester du CSS

Un seul bémol sur ces techniques...

Ca n'existe pas, ou pas encore :(

Tester du CSS aujourd'hui

et pour toutes les tailles de projet stp

Les linters

Les linters allègent la revue de code

Exemple : stylelint.io

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 la syntaxe BEM

Et l'isolation des composants

https://github.com/M6Web/bemlinter

Les linters

```
  ✗ cytron / 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

Retour d'expérience

Attention à la mise en place d'outils de validation

Tests sur différents navigateurs

Faire des tests automatisés sur tous les navigateurs et appareils.

Exemple : BrowserStack

Tests sur différents navigateurs

Les tests fonctionnels peuvent contenir des tests graphiques.

Est-ce visible ?

Est-ce dans le viewport ?

Est-ce en dehors de son parent ?

Tests sur différents navigateurs

Par contre c'est très long !

Impossible de valider tous les navigateurs sur toutes les PRs.

Tests sur différents navigateurs

On fait le choix de lancer les tests IE 11 la nuit.

On souhaite rajouter iPhone, iPad, etc.

https://github.com/M6Web/bemlinter

Test visuel

Les outils de test de regression visuel se démocratisent.

Exemple : Percy.io

Visual Regression Testing Tools

Test visuel

On peut l'utiliser sur notre site, mais

un seul navigateur

du contenu statique

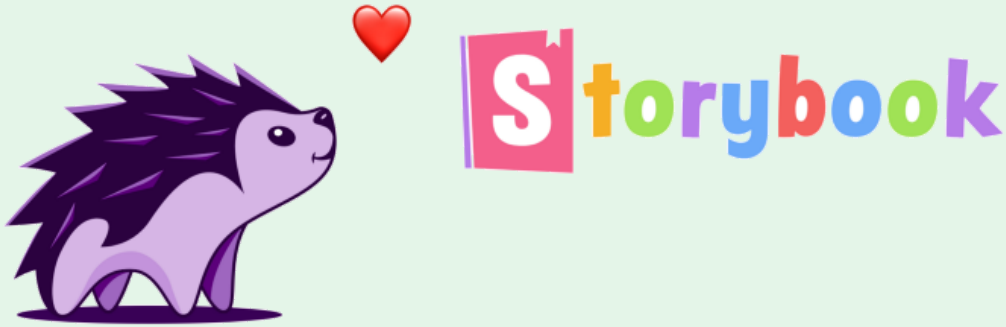
des pages entières

Test visuel

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

Encore faut-il en avoir un !

Test visuel



Turbocharge your React development with Percy and Storybook

Conclusion

1. Ajoutez des linters
2. Ajoutez des TFs
3. Ajoutez des contrôles visuels aux TFs
4. Jouez vos TFs sur plusieurs navigateurs
5. Partagez un style guide
6. Testez les regressions visuelles

Mais ce ne sont pas des tests !?

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

Accompagner la revue de code

Le problème et la solution, c'est l'humain !

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 et intégratrices <3

Merci !

Des questions ?

@iamtzi

tzi.fr/slides/codeurs2017

tzi.fr/slides/codeurs2017.pdf