Tests sur tous les fronts

L'état de l'art des tests front-end

There is a better way

octo.com

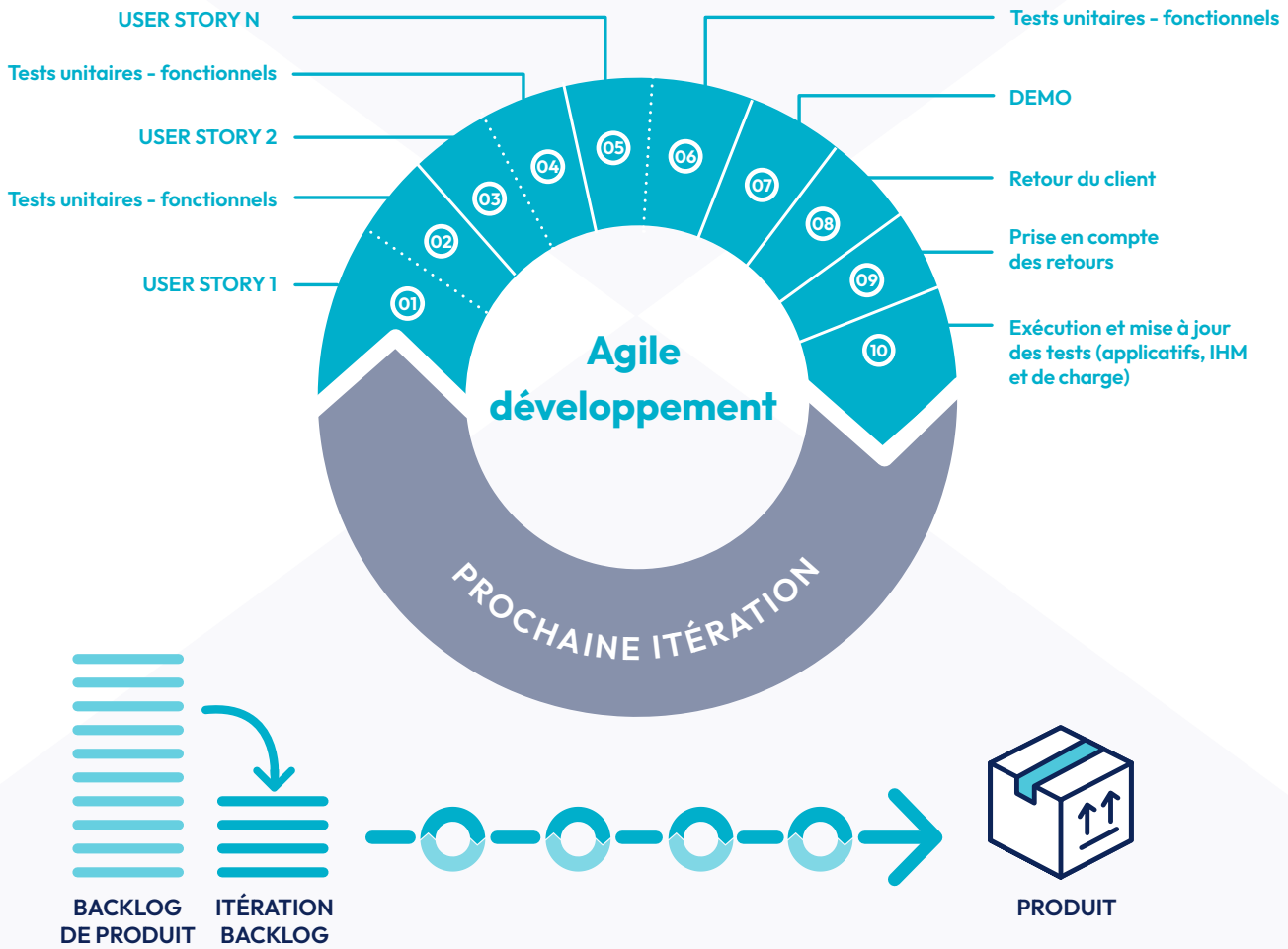
L'état de l'art des tests front-end.

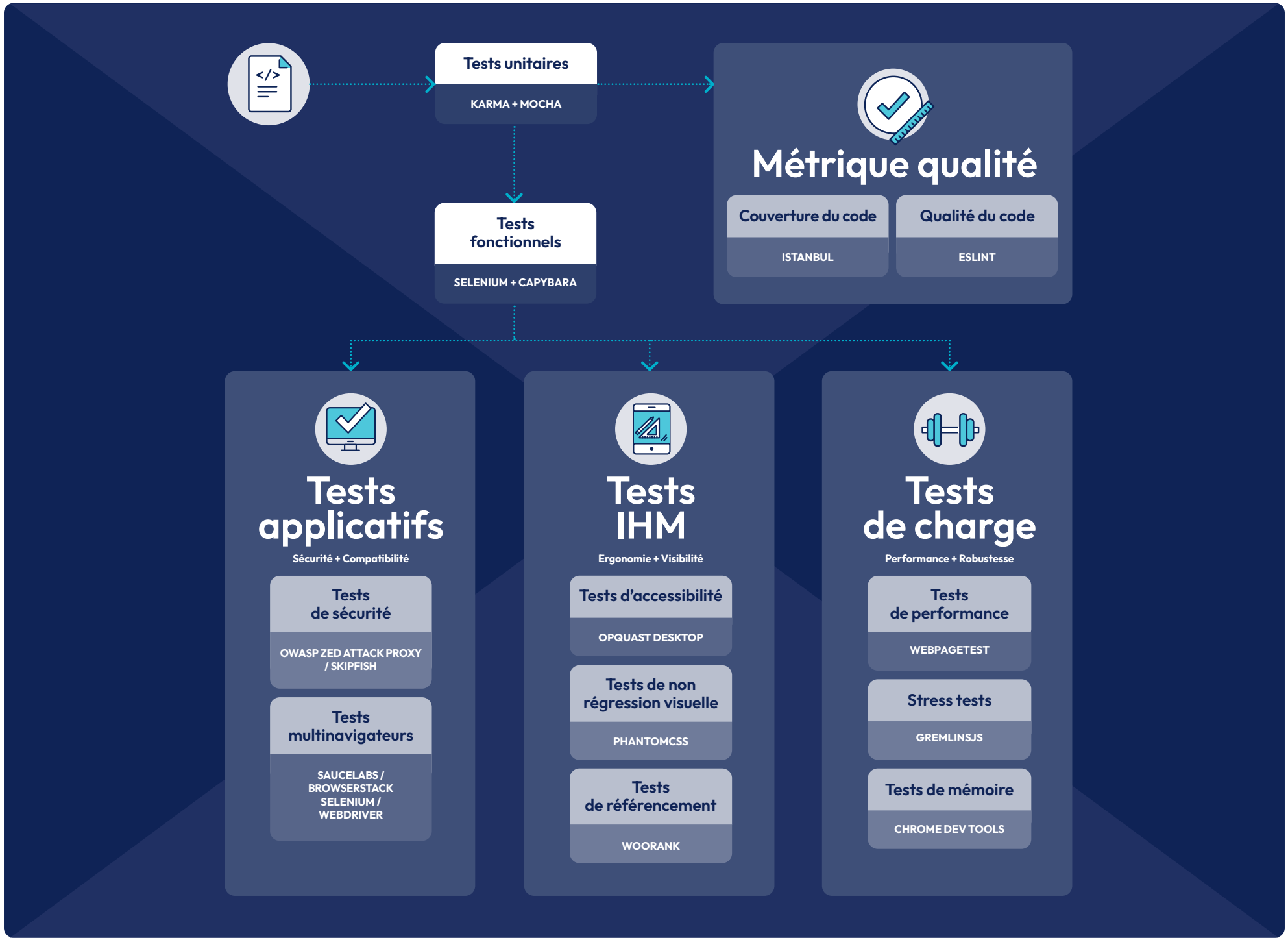
Maîtriser et fiabiliser son code sont aujourd'hui devenus incontournables pour tout développeur devant faire face à des architectures Web de plus en plus riches et complexes.

Il existe des outils pour réaliser des tests front-end d'applications Web et répondre aux besoins d'un développement de qualité.

Nous vous invitons ici à parcourir l'écosystème de ces tests front-end d'applications Web. Que vous soyez déjà convaincus par les tests ou tout simplement curieux, ce document vous guidera pour les mettre en place sur vos projets.

L'intégration des tests en agile.




Tests unitaires
KARMA + MOCHA
Tests fonctionnels
SELENIUM + CAPYBARA
Métrique qualité
Couverture du code
ISTANBUL
Qualité du code
ESLINT
Tests applicatifs
Sécurité + Compatibilité
Tests de sécurité
OWASP ZED ATTACK PROXY / SKIPFISH
Tests multinavigateurs
SAUCELABS / BROWSERSTACK SELENIUM / WEBDRIVER
Tests IHM
Ergonomie + Visibilité
Tests d'accessibilité
OPQUAST DESKTOP
Tests de non régression visuelle
PHANTOMCSS
Tests de référencement
WOORANK
Tests de charge
Performance + Robustesse
Tests de performance
WEBPAGETEST
Stress tests
GREMLINSJS
Tests de mémoire
CHROME DEV TOOLS

Tests unitaires.

Ils assurent la stabilité du code en testant chaque portion (fonction) individuellement. Les régressions seront ainsi remontées très rapidement ce qui permettra de manipuler la base de code avec confiance.

RISQUES

Sur des projets existants, sans test, il est plus complexe de mettre en place des tests unitaires. Il y a une étape de rétro-ingénierie et de séparation du code avant de pouvoir tester.

OUTILS

- Structurer son code : **Mocha** (notre préféré) ou **Jasmine** (supporte IE)
- Écrire des assertions lisibles : **Chai.js**
- Construire des mocks, des stubs et des spy : **Sinon.js**
- Exécuter des tests **Karma** et **PhantomJS**

COÛTS

La mise en place de tests unitaires ne coûte pas cher en termes de developpement (écriture et outils à mettre en place) et temps d'exécution (de l'ordre de la seconde).

EXEMPLE

AVEC SINON.JS ET CHAI.JS

Tester la récupération des amis Facebook d'un utilisateur pour les rendre à la vue :

```
var should = require('chai').should;
 var sinon = require('sinon');
suite('get_my_friends', function() {
test('should return 2 friends of mine',function() {
var user = {
facebook_id : 'my_facebook-id', name : 'my_name'
};
var friends = [
{ name : 'friend 1'},
{ name : 'friend 2'} ];
var stub = sinon.stub(fb, 'getFriends').returns(friends);
var result = get_my_friends(user);
stub.should.be.calledWith(user.facebook_id); });
});
```

Résultat affiché :

```
get_my_friends
  ✔should return 2 friends of mine (50ms)

✔1 test complete (50ms)
```

Tests fonctionnels.

Ils assurent la stabilité de l'application en reproduisant le parcours d'un utilisateur sur le navigateur. Ils testent le bon fonctionnement de l'application et remontent les régressions fonctionnelles.

COÛTS

Leur mise en place peut coûter cher en termes de développement (les différents outils à mettre en place), de temps d'exécution (plusieurs minutes) et surtout de maintenance (sensible aux changements HTML/JS/CSS).

RISQUES

D'expérience les tests fonctionnels ne sont pas toujours stables, cela génère des faux négatifs. Il est donc important de lancer les tests périodiquement pour être certain que l'erreur se reproduise.

OUTILS

- Framework de test pour Angular : **Protractor**
- Framework de test pour applications Web : **Capybara**
- Outil d'automatisation de tests d'interface Web : **Selenium**

FONCTIONNEMENT

Démarrage du serveur Selenium > exécution des tests (Protractor ou Capybara) > communication avec Selenium via un WebDriverJS > exécution des tests sur le navigateur.

EXEMPLE
AVEC CAPYBARA

Tester la fonctionnalité d'ajout d'un commentaire :

1. Implémenter le fonctionnement à tester

```
<form name=»inscription»>
<textarea name=»comment»></textarea>
<button type=»submit»>Valider</button>
<form>
<div class=»list-comments»></div>
```

2. Écrire un test fonctionnel qui vérifie la création d'un commentaire

```
describe 'Comment'' do
it 'should add comment' do
visit(inscription_page)
fill_in comment, :with => "j'ajoute un commentaire'"
click_on 'Valider'
expect(find(:css, '.list-comments).text).to eql('j'ajoute un commentaire')
end
end
```

3. Résultat des tests en sortie s'il n'y a aucune erreur

```
Running E2E tests using environment settig for environment dev
```

Tests multinavigateurs.

Rejouer les tests fonctionnels sur les différents navigateurs et plateformes, permet de fiabiliser la compatibilité de votre application.

COÛTS ET RISQUES

Les coûts peuvent être importants tant sur la mise en place de l'infrastructure, pour exécuter les tests (navigateurs/OS), que sur la maintenance et plus particulièrement sur mobile. Il existe des solutions en SaaS qui répondent à ce besoin.

OUTILS

- Solution SAAS : **SauceLabs, BrowserStack**
- Infrastructure : **Selenium, WebDriverJS**
- Navigateurs : **Chrome, Internet Explorer**
- Plateformes : **Windows**

Tests de non régression visuelle.

Ces tests permettent de garder une interface stable et évitent les régressions visuelles. Ils sont destinés aux sites avec une interface complexe ou avec un grand nombre de visiteurs.

COÛTS

Ils sont plutôt simples à mettre en place car peu d'outils sont nécessaires. L'écriture des tests est simple si on teste uniquement des pages entières ou partielles.

RISQUES

Le résultat des tests retourne parfois des faux négatifs causés par une intolérance aux changements de design. Pour minimiser les risques, évitez les parcours utilisateurs et favorisez l'accès aux pages directement par l'URL.

OUTILS ET PROCESSUS

- Framework de tests visuels : **PhantomCSS**
- Parcours des écrans : **CasperJS**
- Rendu des écrans : **PhantomJS**
- Capture d'écran : **CasperJS**
- Comparaison des écrans : **Resemble.js**

EXEMPLE AVEC PHANTOMCSS

Ces tests ne doivent être mis en place qu'une fois tous les autres développements effectués.

Tests de sécurité.

Le navigateur est le principal vecteur des failles de sécurité sur le Web, il est important de connaître les risques et de s'en prémunir, à la fois pour protéger vos utilisateurs et votre application.

COÛTS

Il faut connaître les failles les plus critiques et comprendre leurs concepts. Vous pouvez retrouver les 10 principaux risques de sécurité sur le Web sur le site de l'OWASP.

OUTILS

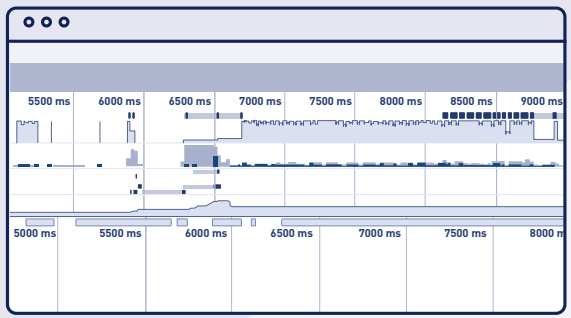
- **Outils automatisation de test :** OWASP Zed Attack Proxy / Skipfish
- **HTML5 Security Cheatsheet :** http://html5sec.org/

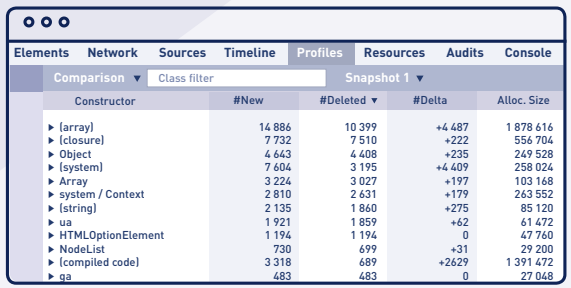
Tests d'endurance/ mémoire.

Les applications Web (SPA, RIA), peuvent bloquer le navigateur à cause de « fuites mémoires ». Elles sont caractérisées par une augmentation permanente de la mémoire et sont difficiles à détecter.

OUTILS

› Profilage de l'usage de la mémoire :
Chrome Dev Tools





COMMENT FONCTIONNE UN TEST DE MÉMOIRE ?

1

DÉTECTER LES FUITES

Le principe consiste à lancer le profileur de mémoire, à faire fonctionner l'application, et à vérifier qu'il n'y ait pas une hausse anormale de la quantité de mémoire utilisée.

2

INVESTIGUER

On commence par trouver un scénario utilisateur qui permette de reproduire la fuite de manière systématique. Une fois cela fait, une technique consiste à prendre des snapshots de la pile mémoire en début et fin de scénario, et comparer leur contenu : on doit normalement voir apparaître les objets qui ne sont jamais supprimés.

Tests d'accessibilité.

Les tests vérifient que les spécifications (RGAA, AccessiWeb, WCAG) sont correctement appliquées ou que le parcours utilisateur n'est pas bloqué par une navigation réduite.

OUTILS

- Audit de code : **Accessibility Developer Tools / Opquast Desktop**
- Validation en ligne : **http://validator.w3.org**
- Plugin de test automatisé : **capybaraaccessible (Capybara), accessibility plugin (Protractor)**
- Outil en SaaS de tests de scénarios automatisés : **Tanaguru**

AMÉLIORER L'ACCESSIBILITÉ D'UN NAVIGATEUR

- Simulateur de lecteur navigateur : **Fangs Screen Reader Simulator**
- Contraster les couleurs : **high contraste**
- Lecteur audio pour vidéo : **HTML5 Audio Description**

Tests de référencement.

Un bon référencement repose sur le contenu et la structure du site, c'est la clé pour être plus visible sur internet.

COÛTS ET RISQUES

Des outils assez perfomants existent pour améliorer son référencement.
Les solutions suggérées peuvent être difficiles à mettre en place.

OUTILS

- Audit de code : **Google tools for webmasters**
- Évolution du classement : **https://www.woorank.com/fr**
- Extension Chrome : **SEOquake**
- Référenciel pour les données structurées : **schema.org**

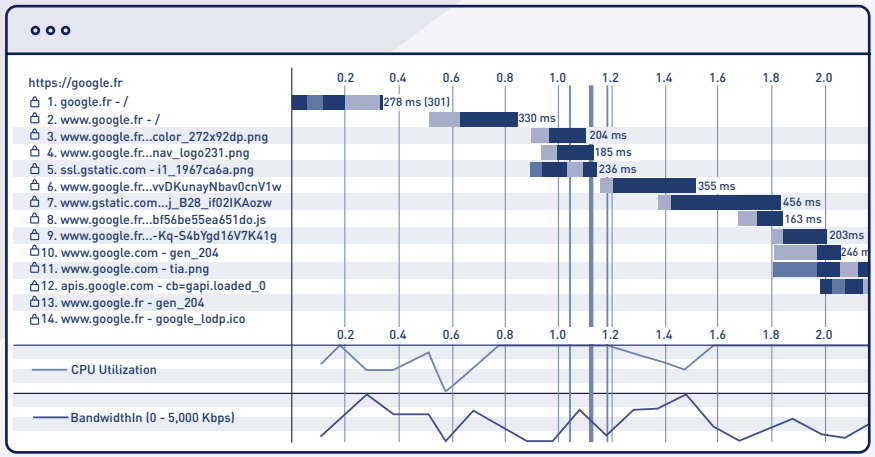
Tests de performance.

Les performances d'une application Web ne reposent plus uniquement sur la partie serveur. Un travail d'optimisation doit aussi être réalisé côté navigateur.

OUTILS

> Solution SaaS : **Google Page Speed Insight, WebPageTest, AgileLoad**

EXEMPLE
AVEC WEBPAGETEST



Stress tests.

Tester la robustesse de l'application Web en exécutant une multitude d'interactions sur l'interface, de façon aléatoire, afin de permettre la détection des points critiques, de problème de mémoire ou de performance.

COÛTS ET RISQUES

Simples à mettre en place, ces tests répondent essentiellement aux besoins d'applications Web de type SPA. En revanche il peut être complexe de détecter la source du problème.

OUTILS

Il n'existe qu'un seul outil :
Gremlins.js

EXEMPLE
AVEC GREMLINS.JS

NOUS VOUS accompagnons
DANS LA MISE EN PLACE DE
frameworks
et architectures
RÉPONDANT À VOS **BESOINS**
d'applications Web
toujours **PLUS RICHES**
ET **MULTIPLA+EFORMES.**

TESTS SUR TOUS LES FRONTS

OCTO Technology

CABINET DE CONSEIL ET DE RÉALISATION IT

“ Dans un monde complexe aux ressources finies, nous recherchons ensemble de meilleures façons d'agir. Nous œuvrons à concevoir et à réaliser les produits numériques essentiels au progrès de nos clients et à l'émergence d'écosystèmes vertueux”

– Manifeste OCTO Technology -

There is a Better Way

1000 OCTOS

3 IMPLANTATIONS

Paris
Toulouse
Hauts-de-France

FORMATION

octo academy
Learn to Change

OCTO EN TÊTE DU PALMARÈS

6x GREAT PLACE TO WORK®

Certified B Corporation

autre cercle

3 CONFÉRENCES

USI
Unexpected Sources of Inspiration

DUCK CONF
La conférence tech par OCTO

School OF Product

TESTS SUR TOUS LES FRONTS

octo.com

Conçu, réalisé et édité par OCTO Technology.

Technology

Part of Accenture