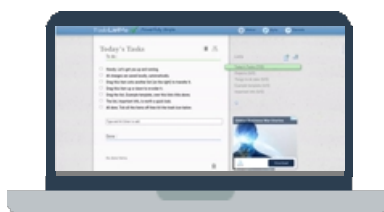


Rapport d'analyse performance et qualité



<http://todolistme.net/>

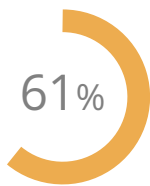
Ce rapport a été généré sur [Dareboost.com](https://dareboost.com), **service en ligne d'analyse et de surveillance de la performance et de la qualité des sites web.**

N'hésitez pas à découvrir [nos offres](#) ou à nous contacter :
contact@dareboost.com.

Sommaire

Résumé	3
Conseils et bonnes pratiques	4
Accessibilité	4
Nombre de requêtes	6
Optimisation du rendu	9
Comment spécifier un jeu de caractères dans l'en-tête Content-Type ?	12
Politique de cache	13
Qualité	16
Référencement (SEO)	19
Sécurité	21
Volume de données	26
Google Analytics	27
jQuery	28

Résumé



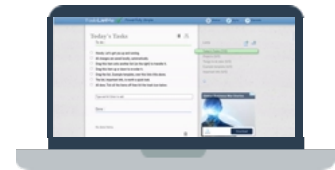
Problèmes



Améliorations



Succès



VISITEUR SIMULÉ :



Chrome



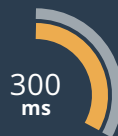
Paris

8.0/1.5Mbps (Latence : 50 ms)

Largest Contentful Paint



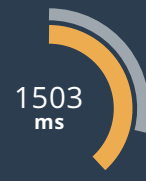
Total Blocking Time



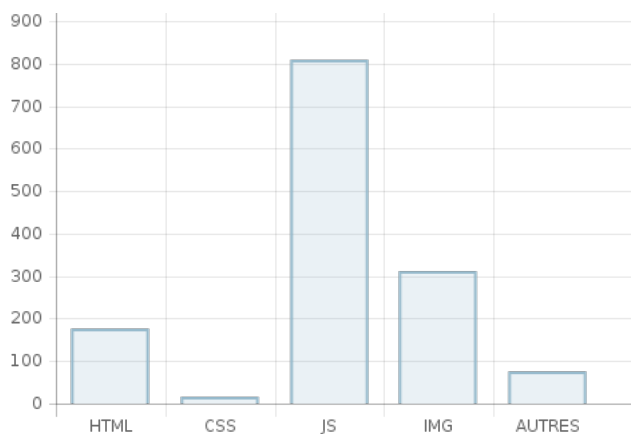
Cumulative Layout Shift



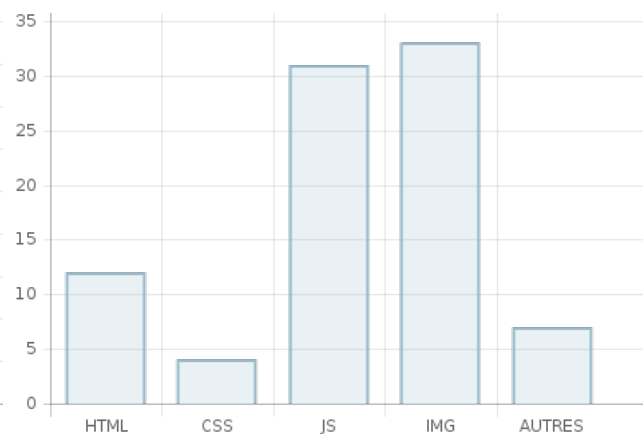
Speed Index



Répartition du poids



Répartition du nombre de requêtes



Technologies détectées



Apache



Google Analytics

DoubleClick Ad Exchange (AdX)



Google AdSense



Google Font API



jQuery



jQuery UI

Conseils et bonnes pratiques

Accessibilité

Vos principaux axes d'optimisation

0/100

#2334

! La langue de la page n'est pas spécifiée

Votre page devrait posséder un attribut `lang` sur le nœud racine `html` : cela permettra aux liseuses d'écran de prendre correctement en charge votre site.

0/100

#2569

! 1 élément vide peut perturber les lecteurs d'écrans

Les éléments `<p>`, ``, `<button>`, `<legend>`, `<caption>`, `<figcaption>` and `<quote>` ne doivent pas être vides car s'ils le sont, certains lecteurs d'écran auront des difficultés à interpréter leur présence.

Supprimez ces éléments vides de votre code ou décorez-les avec l'attribut `aria-hidden` pour que les lecteurs d'écran les ignorent.

```
<p aria-hidden="true"></p>
```

Exemple

L'élément suivant est vide:

- `<p id="synctime">`

! Expliquez le but de chaque champ de formulaire

Améliorez l'expérience utilisateur sur votre site en spécifiant une étiquette descriptive pour chaque champ de formulaire.

Un formulaire est composé de plusieurs champs qui doivent être les plus explicites possibles pour que l'utilisateur comprenne rapidement leur fonction.

Définir un label

Pour indiquer l'objectif d'un champ, nous vous recommandons l'utilisation de la balise `label` :

```
<label for="nom">Renseignez votre nom :</label>
<input id="nom" type="text" name="nom">
```

Exemple

Si vous ne souhaitez pas afficher de label, vous pouvez utiliser les attributs `aria-label` ou `title` (ce dernier n'est cependant pas supporté par certaines liseuses d'écran). À noter qu'il n'est **pas conseillé** d'utiliser l'attribut `placeholder` seul.

En savoir plus sur [l'association de texte à un champ de formulaire](#).

Cette page contient **14 champs sans explications sur leur objectif** :

- `<input type="text" id="syncname">`
- `<input type="password" id="syncpassword">`
- `<input type="button" id="syncbutton" value="Sync" onclick="sync_login();">`
- `<input type="text" id="registersyncname">`
- `<input type="password" id="registersyncpassword">`
- `<input type="button" id="registerbutton" value="Create Account" onclick="create_account();">`
- `<input type="checkbox" onclick="finish_todo(0, 0)">`
- `<input type="checkbox" onclick="finish_todo(0, 1)">`
- `<input type="checkbox" onclick="finish_todo(0, 2)">`
- `<input type="checkbox" onclick="finish_todo(0, 3)">`
- `<input type="checkbox" onclick="finish_todo(0, 4)">`
- `<input type="checkbox" onclick="finish_todo(0, 5)">`
- `<input type="checkbox" onclick="finish_todo(0, 6)">`
- `<input id="newtodo" type="text" value="New Todo" class="newtodonormal" placeholder="Type and hit Enter to add">`

Vos principaux axes d'optimisation

0/100

#2344

1 ressource est inaccessible

Attention, certaines requêtes liées à cette page rencontrent des erreurs. Il est par exemple possible :

- que vous ayez commis une erreur dans votre code HTML, CSS ou JavaScript
- que votre serveur ait rencontré une erreur non gérée
- que certains services utilisés par votre site ne répondent pas

Les codes HTTP suivants ont été renvoyés :

- 404:
 - [apis.google.com/se/0/_/+1/fastbutton?us\[...\]2Fm%3D__features__](https://apis.google.com/se/0/_/+1/fastbutton?us[...]2Fm%3D__features__)

Ces erreurs nuisent probablement à certains contenus ou comportements de votre site, et provoquent un trafic réseau inutile, ce qui impacte donc le temps de chargement de votre site.

0/100

#2442

! Économisez 19 requêtes à l'aide de sprites CSS

Combiner les petites images dans une sprite CSS réduit le nombre de fichiers que le navigateur doit télécharger et accélère donc le chargement de votre page.

Les sprites CSS

Une sprite CSS est un fichier unique dans lequel plusieurs petites images sont regroupées, positionnées les unes à côté des autres. L'affichage de chaque petite image au sein de votre page se fait ensuite par l'application de styles CSS. Une seule requête est alors nécessaire pour récupérer l'ensemble des images. Cette technique est à utiliser uniquement pour les petites images, comme des icônes par exemple, afin que la sprite CSS ne soit pas trop lourde.

Exemple

Voici un exemple de sprite CSS, regroupant plusieurs icônes :



La page applique par ailleurs les styles associés à la sprite, du type :

```
.sprite {
  background-image: url(img/sprite.png);
  background-repeat: no-repeat;
  display: block;
}

.sprite-browsers-firefox {
  width: 31px;
  height: 28px;
  background-position: -74px 0;
}
```

Exemple

Il reste à définir la bonne classe dans votre fichier HTML, et l'icône apparaît :

```
<span class="sprite sprite-browsers-firefox"></span>
```

Exemple

Comment créer une sprite CSS ?

Des outils facilitent leur création qui peut se révéler complexe. Par exemple :

- <https://draeton.github.io/stitches/>
- <http://spritegen.website-performance.org/>

Vous obtenez alors votre sprite CSS avec les styles à appliquer.

20 images peuvent être regroupées au sein d'une sprite CSS. Le nom de domaine todolistme devrait utiliser cette technique pour les ressources suivantes :

- http://todolistme.net/images/top_sync.png
- http://todolistme.net/images/sort_order.png
- http://todolistme.net/images/top_saved.png
- http://todolistme.net/images/top_sync_waiting.png
- http://todolistme.net/images/top_sync_error.png
- <http://todolistme.net/images/info.png>
- http://todolistme.net/images/arrow_down.png
- <http://todolistme.net/images/undo.png>
- <http://todolistme.net/images/purge.png>
- http://todolistme.net/images/category_up.png
- et 10 autres

92/100

#2418

✓ Regroupez 3 fichiers JavaScript

Chaque requête HTTP représente un coût en termes de performance (roundtrip time, utilisation de bande passante...).

Il est ainsi préférable de faire une requête vers un fichier de 50ko de données plutôt que 10 requêtes vers des fichiers de 5ko.

Quelle répartition adopter ?

Répartissez vos scripts en les intégrant directement dans votre code HTML ou en les regroupant dans des fichiers de taille plus conséquente. Nous vous conseillons l'emploi de cette dernière méthode pour profiter des mécanismes de cache.

Vous devriez reconsidérer les ressources suivantes :

- <http://todolistme.net/javascript/lib.js>
- http://todolistme.net/javascript/javascript_e.js
- <http://todolistme.net/javascript/lists.js>

Vos principaux axes d'optimisation

0/100

#2531

! 2 dépendances critiques détectées

La défaillance d'un fournisseur de contenu externe peut rendre indisponible votre site.

Les Single Point Of Failure

Un Single Point Of Failure (SPOF) front-end se caractérise par une dépendance critique d'une page web à un contenu externe, susceptible d'empêcher totalement l'affichage de la page en cas de problème avec le fournisseur externe.

Par exemple, si votre page web appelle un script bloquant hébergé sur les serveurs de Google, votre page se trouve dépendante de toute défaillance provenant de l'appel à ce script.

Que faire pour éviter les SPOF ?

Éliminez ces dépendances autant que possible, même si elles concernent des fournisseurs renommés. Si vous devez faire appel à un contenu externe, assurez-vous que l'appel est effectué de manière asynchrone, et que vous disposez de fallbacks (alternatives en cas d'échec), afin de maîtriser le comportement de la page en cas de panne d'un service externe.

Nous nous assurons ici que la page analysée ne dépend pas de manière critique de ressources externes parmi les plus connues (googleapis, typekit, etc). Il s'agit en effet de cas de Single Point Of Failure front-end

Les ressources suivantes constituent un SPOF pour cette page :

- [//fonts.googleapis.com/css?family=Abel|Architects+Daughter](https://fonts.googleapis.com/css?family=Abel|Architects+Daughter)
- [//code.jquery.com/ui/1.10.3/themes/smoothness/jquery-ui.css](https://code.jquery.com/ui/1.10.3/themes/smoothness/jquery-ui.css)

0/100

#2353

! Différez l'utilisation du code JavaScript

Lorsque le navigateur web rencontre du code JavaScript en interprétant le code source d'une page web, cela peut ralentir considérablement l'affichage de la page, surtout s'il est nécessaire de télécharger un script externe.

Différez au maximum l'utilisation du JavaScript pour assurer un début rapide de l'affichage de la page.

Que faire ?

Avant toute chose, distinguez les portions de votre code JavaScript qui sont critiques au chargement de la page, et doivent être chargées le plus tôt possible. Placez-les dans un fichier externe JavaScript et chargez-le au plus tôt. Gardez ce fichier aussi petit que possible et différez les chargements ou l'exécution de tous les autres scripts JS.

Utilisez si possible l'une des techniques suivantes pour des appels à des fichiers externes :

- utilisation de l'attribut `async`
- utilisation de l'attribut `defer`
- ajout du script dans le DOM en JavaScript, lors de l'évènement onload
- placer les scripts à la fin de votre code source (idéalement à la fin du `<body>`)

306.1 Ko du code JavaScript sont analysés lors du chargement initial de la page. Différez l'analyse de ce code pour éviter de bloquer l'affichage de la page.

- [www.googletagservi\[...\].js](#) (109.5 Ko)
- [platform.twitter.c\[...\].js](#) (84.3 Ko)
- [pagead2.googlesynd\[...\].js](#) (69.3 Ko)
- [googleads.g.double\[...\].js](#) (16.6 Ko de code JavaScript intégré)
- [tpc.googlesyndicat\[...\].js](#) (14.7 Ko)
- [tpc.googlesyndicat\[...\].js](#) (10.0 Ko)
- [connect.facebook.n\[...\].js](#) (1.2 Ko)
- [http://todolistme.net/](#) (551 o de code JavaScript intégré)

! Vous pouvez économiser 2 interprétations et exécutions de scripts

Une librairie ou un script externe sont le plus souvent destinés à être appelé une seule fois par page. Cependant l'utilisation de widgets par exemple peut conduire à plusieurs exécutions inutiles si aucune précaution n'est apportée.

Duplication de scripts

Il est de plus en plus courant de voir des scripts utilisés plusieurs fois au sein d'une même page. Le cas le plus courant est l'intégration de widgets de réseaux sociaux, qu'il est parfois utile de retrouver à plusieurs endroits sur la page pour améliorer son taux d'engagement par exemple. Il ne s'agit pas d'une mauvaise pratique en soit, il faut cependant veiller à certains critères afin de ne pas voir le temps de chargement de la page s'allonger.

Que se passe-t-il si un script est inclu 2 fois dans le code ? Combien de fois est-il chargé ? Interprété ? Exécuté ?

La plupart des navigateurs web récents ne téléchargent qu'une seule fois un script inclu 2 fois. Une exception subsiste : Firefox, qui chargera le fichier autant de fois qu'indiqué si aucune politique de cache efficace n'est configurée.

Outre cette exception, les problèmes de performances vont se ressentir au moment de l'interprétation et l'exécution des scripts. En effet, si le script est mentionné 3 fois dans le code, il sera exécuté 3 fois, et ce sur tous les navigateurs.

Pour plus de détails, n'hésitez pas à [consulter cet article sur le sujet \(EN\)](#).

La solution

Il reste tout de même une solution pour pouvoir utiliser un script plusieurs fois sans l'interpréter et l'exécuter plus d'une fois. Vous devez écrire du code JavaScript, qui vérifie si le script est présent. Si oui, il l'utilise, sinon, il l'injecte afin de l'utiliser.

Prenons l'exemple du widget Facebook, toujours tiré de l'article ci-dessus. À chaque fois que vous voulez intégrer cette fonctionnalité dans votre page, il est nécessaire d'inclure le code suivant :

```
(function(d, s, id){  
  var js, fjs = d.getElementsByTagName(s)[0];  
  if (d.getElementById(id)) {return;}  
  js = d.createElement(s); js.id = id;  
  js.src = "//connect.facebook.net/en_US/sdk.js";  
  fjs.parentNode.insertBefore(js, fjs);  
}(document, 'script', 'facebook-jssdk'));
```

Exemple

Notez en gras la ligne de vérification de la présence du script dans le document. Le script est alors inclu, interprété et exécuté uniquement lors du premier appel dans la page. Les autres appels tomberont dans le cas de la ligne mise en valeur, et ne feront donc qu'utiliser le script déjà inclu et exécuté.

Les scripts suivants sont interprétés et exécutés plusieurs fois sur votre page :

- [adservice.google.nl/adsid/integrator\[...\]](#)[n=todolistme.net](#) (interprété et exécuté 2 fois)
- [adservice.google.com/adsid/integrato\[...\]](#)[n=todolistme.net](#) (interprété et exécuté 2 fois)

Les autres conseils

! Évitez d'utiliser des <meta> http-equiv

Les en-têtes HTTP sont plus performants que les metas http-equiv.

Les balises <meta http-equiv="">

Les metas `http-equiv` permettent de communiquer au navigateur web des informations équivalentes à celles d'en-têtes HTTP. Par exemple, la meta `<meta http-equiv="content-type">` aura la même portée que l'en-tête HTTP `Content-Type`.

Deux points n'encouragent pas à l'utilisation des metas http-equiv :

- passer par la meta nécessite d'interpréter le début de la page HTML, ce qui en termes de performances est plus lent que de passer par les en-têtes HTTP
- si l'en-tête HTTP est déjà présent, la meta est ignorée

Dans quel cas les <meta http-equiv=""> sont utiles ?

Un seul cas peut justifier la présence de ces metas : si vous n'avez pas accès à la configuration de votre serveur, et donc des en-têtes HTTP.

Nous vous conseillons cependant d'utiliser un serveur configurable, pour établir un site le plus performant possible.

Cette page contient 16 balises meta `http-equiv`. Vous devriez les remplacer si possible :

- origin-trial
- origin-trial
- origin-trial
- origin-trial
- et 12 autres

✓ Spécifiez un jeu de caractères dans vos en-têtes HTTP de réponse

Aucun jeu de caractères n'est spécifié dans les en-têtes HTTP des ressources suivantes. La spécification d'un jeu de caractères permet d'accélérer l'affichage des pages dans le navigateur.

Les ressources suivantes sont hébergées par un parti tiers, il se peut donc qu'elles ne soient pas sous votre responsabilité. Vous devriez cependant considérer toute alternative possible à ces fichiers pour rester en accord avec la bonne pratique.

- [tpc.googlesyndicat\[...\].html](#)

En spécifiant le jeu de caractères utilisé dans l'en-tête HTTP **Content-Type**, vous permettez au navigateur d'analyser immédiatement la page. Par exemple : `content-type: text/html; charset=UTF-8`.

Lorsque le navigateur reçoit des octets, il doit identifier la collection de lettres et de symboles qui ont été utilisés pour écrire le texte qui a été converti en ces octets, ainsi que le codage utilisé pour cette conversion, afin de l'inverser. Si aucune information de ce type n'a été transmise, le navigateur tentera de trouver des motifs reconnaissables dans les octets pour déterminer l'encodage lui-même, et éventuellement essayer certains jeux de caractères courants, ce qui prendra du temps, retardant le traitement ultérieur de la page.

Comment spécifier un jeu de caractères dans l'en-tête Content-Type ?

Dans l'explication suivante, nous considérerons UTF-8 comme le jeu de caractères à définir, mais n'oubliez pas que le jeu de caractères déclaré dans votre en-tête HTTP Content-Type doit refléter le jeu de caractères utilisé pour encoder le fichier, qui peut ne pas être UTF-8.

Dans **Apache 2.2+**, la configuration de l'UTF-8 comme jeu de caractères par défaut pour les fichiers de types `text/plain` ou `text/html` implique d'utiliser la directive `AddDefaultCharset` :

```
AddDefaultCharset utf-8
```

[Exemple](#)

Pour les autres types de fichiers, vous aurez besoin d'utiliser la directive `AddCharset` :

```
AddCharset utf-8 .js .css ...
```

[Exemple](#)

Dans **nginx**, vous devrez vous assurer que le module `ngx_http_charset_module` est chargé, puis utiliser la directive `charset` :

```
charset utf-8;
```

[Exemple](#)

Ici aussi, il est possible d'affiner le champs d'application pour que d'autres types de fichiers que `text/html` soient livrés en UTF-8, en utilisant la directive `charset_types` :

```
charset_types text/html text/css application/javascript
```

[Exemple](#)

Vos principaux axes d'optimisation

0/100

#2425

! 25 de vos requêtes ne définissent pas de politique de cache avec Apache

Les en-têtes de cache (`Cache-Control` , `ETag` , anciennement `Expires`) sont indispensables pour une politique de cache efficace, et vont grandement impacter le temps de chargement de vos pages lors des prochaines visites.

L'en-tête `Cache-Control`

Chaque ressource peut définir ses règles de mises en cache via l'en-tête HTTP `Cache-Control` . La propriété `max-age` permet de définir la durée de la mise en cache (en secondes), et peut être accompagnée d'instructions concernant la mise en cache des ressources sur des serveurs intermédiaires, situés entre le navigateur et le serveur délivrant la ressource.

L'en-tête suivant indique que la réponse peut être mise en cache sur des serveurs intermédiaires et sur le navigateur (`public` , par opposition à `private` lorsque seul le navigateur est autorisé à mettre en cache) pendant deux heures :

```
Cache-Control: private, max-age=7300
```

Exemple

Si certaines de vos ressources ne doivent pas être mises en cache, vous pouvez également l'expliquer :

```
Cache-Control: no-store
```

Exemple

L'en-tête `Expires`

L'en-tête `Expires` est l'en-tête le plus ancien permettant de gérer la mise en cache des ressources, et vous permettra de gérer le cache pour des navigateurs ne supportant pas `Cache-Control` .

Grâce à l'en-tête `Expires` , vous pouvez définir une date d'expiration pour chaque ressource : tant que la date n'est pas dépassée, l'utilisateur stocke et utilise la ressource en cache.

La date d'expiration des ressources se paramètre à l'aide de l'en-tête HTTP `Expires` :

```
Expires: Thu, 25 Dec 2014 20:00:00 GMT
```

Exemple

Vous pouvez indiquer une date d'expiration lointaine pour les ressources statiques (1 an maximum), et plus proche pour des ressources susceptibles d'être modifiées (48h minimum).

Revalidation des ressources à expiration du cache

Si aucune politique de cache n'est définie pour une ressource ou que la durée de sa mise cache est dépassée, le navigateur procède à une nouvelle requête pour télécharger la ressource.

Pour éviter au navigateur de télécharger une ressource qui n'aurait pas été modifiée depuis sa mise en cache, utilisez l'en-tête HTTP `ETag` . Il permet d'associer à chaque version d'une ressource un jeton de validation. Lorsque le cache d'une ressource expire, le navigateur demandera à nouveau au serveur la ressource, en lui passant ce jeton avec l'en-tête `If-None-Match` contenant la valeur du jeton. Le serveur comparera sa version du jeton avec celle fournie. Si la ressource n'a pas été modifiée, le serveur autorisera le navigateur à renouveler la mise en cache de la ressource sans la télécharger à nouveau, via une réponse HTTP 304 Not Modified.

Invalidation forcée du cache

Lorsque vous mettez en production **une nouvelle version de votre site, pensez bien à renommer les ressources statiques** ayant été modifiées (versioning), afin de forcer le téléchargement de ces nouvelles versions par vos utilisateurs, qui risquent sinon de conserver les ressources stockées en cache dans leurs navigateurs. Ils risquent alors de se retrouver dans une version instable de votre page. Par exemple :

```
maresource.min.20140101.js
```

Exemple

Pour en savoir plus sur la mise en cache HTTP, n'hésitez pas à consulter [les recommandations de Google](#).

Actions à mener

🔗 Cette page est retournée par un serveur Apache. Vérifiez s'il utilise bien le module `mod_expires`. Voici un exemple de configuration possible, à adapter selon vos besoins (fichier `.htaccess` par exemple) :

```
<IfModule mod_expires.c>
ExpiresActive On
ExpiresDefault "access plus 1 month"
ExpiresByType image/x-icon "access plus 1 year"
ExpiresByType image/gif "access plus 1 month"
ExpiresByType image/png "access plus 1 month"
```

Exemple

```
ExpiresByType image/jpeg "access plus 1 month"
ExpiresByType text/css "access plus 1 year"
ExpiresByType application/javascript "access plus 1 year"
</IfModule>
```

25 de vos ressources **n'ont pas de date d'expiration** :

- http://todolistme.net/css/style_g.css
- <http://todolistme.net/javascript/lib.js>
- http://todolistme.net/javascript/javascript_e.js
- <http://todolistme.net/javascript/lists.js>
- http://todolistme.net/images/top_sync.png
- <http://todolistme.net/images/info.png>
- http://todolistme.net/images/top_new_window.png
- et 18 autres

90/100

#2437

✓ 10 requêtes provenant de partis tiers n'adoptent pas une politique de cache assez longue

En permettant la conservation de vos ressources statiques plusieurs jours en cache, vous diminuez la charge de votre serveur.

Certaines de vos ressources utilisent l'en-tête HTTP **Expires** pour disposer d'une politique de cache efficace, ce qui correspond à une bonne pratique. Cependant, vous devriez envisager d'améliorer sa configuration pour profiter au mieux des mécanismes de cache.

Quelle configuration adopter ?

Nous vous conseillons de paramétrer l'en-tête **Expires** afin que la date mentionnée soit comprise entre 2 jours et 1 an par rapport à aujourd'hui.

Cette page contient 10 ressources **n'ayant pas une date d'expiration assez lointaine** :

Les ressources suivantes sont hébergées par un parti tiers, il se peut donc qu'elles ne soient pas sous votre responsabilité. Vous devriez cependant considérer toute alternative possible à ces fichiers pour rester en accord avec la bonne pratique.

- http://connect.facebook.net/en_US/all.js
- http://pagead2.googlesyndication.com/pagead/show_ads.js
- <http://platform.twitter.com/widgets.js>
- <http://www.google-analytics.com/ga.js>
- pagead2.googlesyndication.com/pagead/js/...jstme.net&amaexp=1
- <http://apis.google.com/js/plusone.js>
- www.googletagservices.com/activeview/js/...js?cb=%2Fr20100101
- www.googletagservices.com/activeview/js/...js?cache=r20110914
- <https://pagead2.googlesyndication.com/pagead/js/adsbygoogle.js>
- <https://tpc.googlesyndication.com/sodar/sodar2.js>

Vos principaux axes d'optimisation

0/100

#2491

! Plusieurs éléments utilisent le même identifiant

Utiliser un même identifiant pour 2 éléments différents peut entraîner des effets de bord, notamment lors de l'exécution de JavaScript ou lors de l'application des règles CSS.

Identifiant d'éléments

Chaque élément d'une page web peut disposer d'un identifiant, défini par l'attribut `id` :

```
<p>
  <span id="mySpan1"></span>
</p>
```

Exemple

Ces identifiants vous permettent de manipuler les éléments de votre page avec des instructions CSS ou JavaScript.

Que faire ?

Vous devez faire en sorte qu'aucun élément de la page ne dispose du même identifiant. Si vous souhaitez partager des propriétés/comportements entre plusieurs éléments, il faudra utiliser l'attribut `class`, dédié à cet effet :

```
<p>
  <span class="mySpans"></span><span class="mySpans"></span>
</p>
```

Exemple

L'identifiant suivant est utilisé plusieurs fois dans la page :

- `later_number`, utilisé 2 fois

0/100

#2517

! Le mot clé !important est utilisé 17 fois

Le mot clé !important s'approche d'un hack permettant d'interdire toute surcharge d'une propriété définie. Si parfois il peut s'avérer utile, cela doit rester exceptionnel !

Si vous en abusez, songez à revoir votre structure de code CSS. Dans ce conseil, vous êtes pénalisé au-delà de 10 utilisations de ce mot clef.

Voici la liste des !important détectés :

http://todolistme.net/css/style_g.css

- `.dropontotitle {color: #f7ba79 !important}` (ligne 6, col 5418)
- `.currentlist {border: 1px solid #279c4e !important}` (ligne 6, col 12473)
- `.currentlist {padding-left: 45px !important}` (ligne 6, col 12587)
- `.currentlist {color: #444 !important}` (ligne 6, col 12628)
- `.currentlist:hover .mergelist {visibility: hidden !important...}` (ligne 6, col 13072)
- `.dropontotomorrow {background-color: #f7ba79 !important}` (ligne 6, col 14869)
- `#backbutton {width: 150px !important}` (ligne 6, col 15261)
- `body.remote .currentlist {padding: 0 !important}` (ligne 6, col 18333)
- et 9 autres

95/100

#2516

✓ 1 sélecteur CSS est trop complexe

Des sélecteurs CSS trop complexes nuisent à la lisibilité générale du code et constituent un gain possible en termes de performance.

Les règles CSS

Les règles CSS permettent de sélectionner un ensemble d'éléments du DOM, et de leur appliquer des propriétés de mise en forme.

Elles sont composées d'un ensemble de sélecteurs (identifiants, classes, noms d'éléments, etc.) permettant d'adresser les éléments à cibler. Une page bien structurée permet l'écriture de règles CSS simples, rapidement lisibles et performantes.

Comment améliorer ce point ?

Simplifiez vos sélecteurs tant que possible, quitte à introduire de nouvelles classes dans votre code pour éviter des enchaînements trop longs.

Par exemple, la règle suivante :

```
body td .maClasse div .maClasse2 span .maClasse3 {}
```

Exemple

Pourrait être améliorée, en passant éventuellement par un identifiant :

```
#monIdentifiant .maClasse3 {}
```

Exemple

Les fichiers suivants définissent des sélecteurs CSS trop complexes :

http://todolistme.net/css/style_g.css

- `// This code is copyright Ryan Chadwick 2012 // Please do not use this code your...`

Les autres conseils

92/100

#2383

✓ Séparez vos styles de votre code HTML

Dissocier les balises HTML de leur style améliore la maintenabilité du code et favorise sa factorisation.

Définir les styles CSS

Les styles CSS permettent de mettre en forme la page. Pour les définir, vous disposez de 3 méthodes principales :

- la déclaration dans un fichier CSS
- la déclaration dans un bloc "inline" (balises <style>)
- la déclaration dans un attribut style d'un élément HTML

Que faire ?

Nous vous conseillons de regrouper vos styles CSS au sein de balises `<style>` ou dans des fichiers CSS dédiés. Ainsi, la partie HTML se charge uniquement de fournir la structure de la page, et sa mise en forme est externalisée. L'attribut `style` devrait uniquement être généré en cas de besoin par du code JavaScript (exemple : nécessité de connaître la taille de l'écran).

Cette page utilise 1 attribut(s) `style` :

- ``

Vos principaux axes d'optimisation

0/100

#69

! Ajoutez l'attribut alt sur vos balises

L'attribut `alt` est un critère important en terme de référencement (SEO). En effet, les robots d'indexation des moteurs de recherche ne peuvent pas analyser de contenu graphique. Il se servent alors de leur texte alternatif pour répondre aux requêtes des internautes. [C'est par exemple le cas pour Google images](#).

```

```

Exemple

De plus, l'attribut `alt` s'avère utile dans d'autres cas :

- utilisation d'un lecteur d'écran (personnes malvoyantes)
- connexion trop lentes pour charger l'image
- affichage d'un contenu dans le cas d'une erreur dans l'attribut `src`

Vous avez 12 balises `img`, mais 12 d'entre elles ne définissent pas d'attribut `alt` :

- ``
- ``
- ``
- ``
- ``
- ``
- ``
- ``
- ``
- ``
- ``
- ``

Il reste possible de définir un texte alternatif vide si aucune description ne semble cohérente pour l'image, mais soyez vigilant. Il est préférable de veiller à ce que la majorité de vos images indiquent un texte cohérent. Voir [les recommandations du W3C \(EN\)](#).

0/100

#2503

! Le fichier robots.txt doit être défini

Facilitez au maximum la découverte de votre site par les robots en leur indiquant quelles URLs ne doivent pas être explorées.

Fichier robots.txt

Ce fichier est placé à la racine du site et est interprété par les robots en charge du référencement de votre site. Il délivre des instructions pour indiquer les pages à explorer par les robots.

Notez que ces instructions sont données à titre indicatives. Un robot quelconque ne sera pas bloqué par les restrictions du fichier.

Nous n'avons pas détecté de fichier robots.txt sur ce site, vous devriez le mettre en place à l'adresse suivante :

- <http://todolistme.net/robots.txt>

0/100

#2399

! Votre page n'expose pas de propriété Open Graph

Vous pouvez aider les réseaux sociaux à comprendre votre page en utilisant les propriétés Open Graph.

Les propriétés de Open Graph

Plusieurs propriétés permettent aux réseaux sociaux d'en savoir plus sur le contenu de la page. Nous recommandons d'utiliser au moins les propriétés requises :

- `<meta property="og:title" content="Le titre" />` Exemple
- `<meta property="og:type" content="Le type" />` Exemple
- `<meta property="og:url" content="http://url.com/" />` Exemple
- `<meta property="og:image" content="http://image.jpg" />` Exemple

Ces informations permettent d'améliorer l'interprétation de la page par les réseaux sociaux, y compris Facebook. [En savoir plus sur Open Graph](#).

Cette page ne fournit pas d'informations à destination des réseaux sociaux.

Vos principaux axes d'optimisation

0/100

#2549

! Utilisez HTTPS pour collecter des données sensibles

La page contient des champs pour récolter des mots de passe ou des informations bancaires, vous devez garantir à l'utilisateur une connexion sécurisée.

Google multiplie les actions en faveur du HTTPS. Après avoir ajouté ce critère dans ses algorithmes de référencement ([voir l'annonce](#) - en anglais), Google a fait progressivement évoluer Chrome pour mettre en valeur l'absence d'un contexte sécurisé dans différents cas de collecte d'informations provenant des utilisateurs. Les autres navigateurs suivent également cette tendance.

De plus, cette page récolte des données sensibles (mot de passes ou informations bancaires). Sur [Google Chrome](#), vos internautes seront avertis ([EN](#)) que le site est "Non Sécurisé" :



La mise en place du HTTPS sur un site provoque parfois quelques réticences (coût, impact sur les performances...). Mais le marché a évolué et nombre de ces craintes sont désormais injustifiées. Le passage à HTTPS doit être envisagé..

Comment mettre en place le protocole HTTPS ?

Vous devez mettre en place un certificat obtenu auprès d'une autorité de certification fiable. Renseignez vous notamment auprès de votre hébergeur qui peut vous aider à obtenir ce certificat. [La page suivante devrait par ailleurs vous aider dans votre démarche de migration vers le protocole HTTPS.](#)

Un certificat gratuit ? Let's Encrypt !

[Let's Encrypt](#) est une autorité de certification gratuite, automatisée et open source. De nombreux hébergeurs proposent d'activer la génération et le renouvellement automatique de certificats gratuits directement depuis l'interface d'administration de votre domaine. Contactez votre hébergeur pour plus d'informations.

! Il manque une politique de sécurité sur la provenance de vos ressources

Protégez votre site Web contre les attaques de type XSS (Cross-Site Scripting) en mettant en place une politique restrictive de sécurité du contenu.

Les attaques XSS

Les attaques XSS sont un type d'attaque dans laquelle des données malveillantes sont malicieusement ajoutées aux sites Web. Le nombre de vulnérabilités permettant ces attaques est assez important, c'est pourquoi il est aussi utile de les prévenir que de limiter leurs effets néfastes.

Vous pouvez protéger vos pages contre ces attaques en limitant l'exécution à des portions de code identifiées par le domaine auquel elles appartiennent, ou légitimées par un identifiant d'intégrité. Le code ne correspondant pas à cette politique de sécurité ne sera pas exécuté et l'utilisateur sera informé.

Configurer un en-tête HTTP "Content-Security-Policy" (CSP)

Pour prévenir ou limiter les dégâts potentiels d'une attaque XSS, vous devez configurer votre serveur afin que la réponse de la ressource principale contienne l'en-tête HTTP "Content-Security-Policy".

Voici un exemple d'utilisation de cet en-tête :

```
Content-Security-Policy: script-src 'self' https://apis.google.com
```

Exemple

Dans ce cas, seuls les scripts provenant de l'hôte courant ou de <https://apis.google.com> seront exécutés.

Découvrez-en davantage sur les Content Security Policy dans [les directives du W3C](#).

Soyez vigilant, si l'en-tête est mal configuré, certains de vos contenus, scripts, ou encore styles pourront être bloqués, ce qui pourrait engendrer des effets de bords non souhaités. De plus, les restrictions s'appliquent à toutes les pages du site. Nous vous conseillons de tester les différentes pages de votre site avant de déployer l'instruction dans votre environnement de production.

La CSP peut se configurer avec Apache. Assurez vous tout d'abord que le module `mod_headers` est bien activé. Ensuite, dans votre configuration (fichier `.htaccess` par exemple), vous pouvez indiquer votre politique de sécurité en adéquation avec votre cas. Ci-dessous un exemple :

```
<IfModule mod_headers.c>
Header set Content-Security-Policy "script-src 'self' https://www.google.com"
</IfModule>
```

Exemple

Dans cet exemple, la page n'accepte que les scripts provenant du même nom de domaine et de google.com.

Aucune CSP n'a été détectée sur cette page : elle est plus facilement exposée à des attaques de type XSS.

! Cette page est exposée à des attaques du type "clickjacking"

Ne permettez pas à des personnes malveillantes d'intégrer vos pages sur leur site.

Clickjacking

Ce type d'attaque consiste à intégrer votre page sur un site malveillant via des balises <frame> ou <iframe>. Ainsi il est possible de faire croire à un internaute qu'il est sur votre propre page. L'internaute peu averti sera en confiance et sera potentiellement amené à saisir des informations que le site malveillant sera à même d'intercepter.

Vous devez donc indiquer quels domaines sont autorisés à intégrer votre page.

Comment se prémunir du Clickjacking ?

Il y a deux méthodes permettant de se protéger contre cette attaque.

1/ Utilisez un en-tête HTTP "X-Frame-Options". Configurez votre serveur de telle sorte que la réponse de la ressource principale contienne l'en-tête HTTP "X-Frame-Options".

Trois types de valeurs peuvent être définies :

- **DENY** pour refuser toute frame ou iframe intégrant la page
- **SAMEORIGIN** pour n'autoriser que les frames provenant du même nom de domaine
- **ALLOW-FROM uri** pour préciser les domaines pouvant intégrer la page dans une frame ([non compatible avec tous les navigateurs \(EN\)](#))

2/ Définissez une directive `frame-ancestors` explicite dans votre politique de sécurité. Cette approche, plus récente et supportée par moins de navigateurs, vous permettra cependant d'autoriser plusieurs domaines plutôt qu'uniquement le domaine courant. Configurer cette directive avec la valeur 'none' revient à mettre en place l'en-tête **X-Frame-Options: DENY**.

Quelle approche choisir ? Si vous n'avez que le domaine courant à autoriser, mettez en place les deux politiques de sécurité pour une meilleure compatibilité avec les anciens navigateurs. Si vous souhaitez autoriser plusieurs domaines à intégrer le vôtre, privilégiez la directive de sécurité "frame-ancestors".

L'en-tête HTTP "X-Frame-Options" peut se configurer avec Apache. Assurez vous tout d'abord que le module `mod_headers` est bien activé. Ensuite, vous pouvez indiquer l'en-tête dans votre configuration (fichier `.htaccess` par exemple), comme sur l'exemple ci-dessous :

```
<IfModule mod_headers.c>
Header always set X-FRAME-OPTIONS "DENY"
</IfModule>
```

Exemple

Ni l'en-tête "X-Frame-Options" ni la directive "frame-ancestors" ne sont définis sur cette page, vous êtes plus facilement exposé au clickjacking.

Les autres conseils

! 3 ressources tierces sont délivrées sans contrôle d'intégrité (SRI)

Si cette page charge des contenus provenant de tiers, assurez-vous de l'intégrité de ces données.

Contrôle d'intégrité SRI

Pour vous assurer qu'une ressource provenant d'une tierce partie n'a pas été altérée, utilisez un contrôle d'intégrité SRI (SubResource Integrity). Pour cela, ajoutez l'attribut `integrity` sur les balises `<script>` et `<link>` qui chargent ces ressources. Exemple :

```
<script src="https://exemple.com/exemple-framework.js"
  integrity="sha384-oqVuAfXRKap7fdgcCY5uykM6+R9GqQ8K/uxy9rx7HNQIGY11kPzQho1wx4JwY8wC"
  crossorigin="anonymous">
</script>
```

Exemple

L'attribut `integrity` a pour valeur une empreinte, correspondant au SHA du contenu souhaité, encodé en base64. Le navigateur comparera alors ce SHA avec celui du contenu téléchargé, pour déterminer si la ressource correspond bien à celle attendue.

Plusieurs outils vous aideront dans la création de cette empreinte. En ligne de commande, vous pouvez utiliser openssl. Des outils en ligne existent par ailleurs, tels que srihash.org ou report-uri.io. [En savoir plus sur SubResource Integrity.](#)

Les ressources suivantes sont chargées par des tierces parties, ajoutez-leur un contrôle d'intégrité :

- //code.jquery.com/jquery-2.2.4.min.js
- //code.jquery.com/ui/1.12.1/jquery-ui.js
- //code.jquery.com/ui/1.10.3/themes/smoothness/jquery-ui.css

! Désactivez la détection automatique du type des ressources

Protégez-vous de l'exploitation malveillante du MIME Sniffing.

Le MIME-Type sniffing


Les navigateurs Internet Explorer et Chrome disposent d'une fonctionnalité dite de "MIME-Type sniffing", qui consiste à détecter automatiquement le type d'une ressource web. Ainsi, une ressource déclarée comme étant une image pourra être interprétée comme un script si tel est son contenu.

Une personne malveillante pourrait profiter d'un envoi de fichier sur votre site par exemple pour injecter du code malveillant. Nous vous conseillons de désactiver le MIME-Type sniffing pour limiter les effets de l'envoi d'un tel fichier.

Chrome a travaillé sur une fonctionnalité appelée [Isolation de Site \[EN\]](#) qui permet d'atténuer considérablement l'exploitation de ces types de vulnérabilités. L'isolation du site est plus efficace lorsque les types MIME sont corrects.

La solution : configurer un en-tête HTTP "X-Content-Type-Options"

Ajoutez **dans les réponses de chacune de vos ressources** l'en-tête HTTP "X-Content-Type-Options", associé à la valeur "nosniff". Cela vous permettra de vous prémunir du risque d'une mauvaise interprétation de vos ressources.

 **L'en-tête HTTP "X-Content-Type-Options" peut se configurer avec Apache. Assurez vous tout d'abord que le module [mod_headers](#) est bien activé. Ensuite, vous pouvez indiquer l'en-tête dans votre configuration (fichier .htaccess par exemple), comme sur l'exemple ci-dessous :**

```
<IfModule mod_headers.c>
Header always set X-Content-Type-Options "nosniff"
</IfModule>
```

Exemple

Sur cette page, **vous devriez configurer les ressources suivantes**, qui risquent d'être mal interprétées :

Ressources provenant de "todolistme"

- http://todolistme.net/css/style_g.css
- <http://todolistme.net/javascript/lib.js>
- http://todolistme.net/javascript/javascript_e.js
- <http://todolistme.net/javascript/lists.js>

Ressources hébergées par un parti tiers

Les ressources suivantes sont hébergées par un parti tiers, il se peut donc qu'elles ne soient pas sous votre responsabilité. Vous devriez cependant considérer toute alternative possible à ces fichiers pour rester en accord avec la bonne pratique.

- <http://code.jquery.com/ui/1.10.3/themes/smoothness/jquery-ui.css>
- <http://platform.twitter.com/widgets.js>
- <http://code.jquery.com/jquery-2.2.4.min.js>
- <http://code.jquery.com/ui/1.12.1/jquery-ui.js>
- <https://platform.twitter.com/js/button.5573c974dc31bbdab5ea7923a0bd5cf3.js>

Vos principaux axes d'optimisation

59/100

#2436

! 1 image est redimensionnée côté navigateur

Si vos images sont plus grandes que leur zone d'affichage, le navigateur téléchargera des données inutiles et procédera à des redimensionnements disgracieux.

Éviter le redimensionnement d'images côté navigateur

Lorsqu'il souhaite afficher une image dans votre page, le navigateur fait tout son possible pour l'adapter à sa surface de rendu. Si l'image est trop grande, il la réduit.

Fournissez directement vos images aux dimensions d'affichage utilisées sur votre site. Vous évitez ainsi l'envoi de données inutiles sur le réseau, ce qui diminue le temps de chargement de la page.

Comme les algorithmes embarqués dans les navigateurs ne sont pas aussi bons que ceux des outils dédiés à la manipulation d'images, vous obtiendrez un résultat visuel plus satisfaisant en redimensionnant vos images en amont, plutôt qu'en laissant le navigateur le faire.

Utiliser des Images Adaptatives (Responsive)

Plusieurs méthodes existent, permettant de servir des images adaptées au navigateur quelle que soit la résolution de l'écran ou la densité de pixels du matériel. Nous vous conseillons de lire les ressources suivantes :

- "Images adaptatives", sur le [Mozilla Developer Network \(MDN\)](#)
- [Picturefill](#), permet d'utiliser l'élément <picture> dans des navigateurs qui ne le supportent pas encore (EN)
- [RICG, groupe de travail sur les images responsive](#) (EN)

Ne redimensionnez pas l'image suivante :

- <http://todolistme.net/images/tick.png> (taille d'affichage : 41x35)

60/100

#2388

! Vous pouvez réduire certains fichiers JavaScript (minification)

En compressant votre code JavaScript, vous pouvez libérer de nombreux octets de données et réduire les délais de téléchargement, d'analyse et d'exécution.

[Réduisez la taille des ressources JavaScript](#) suivantes afin de gagner 46.0 Ko (réduction de 38%).

Les ressources suivantes sont hébergées par un parti tiers, il se peut donc qu'elles ne soient pas sous votre responsabilité. Vous devriez cependant considérer toute alternative possible à ces fichiers pour rester en accord avec la bonne pratique.

- Une réduction de la taille de code.jquery.com/ui/.../ui.js pourrait libérer 46.0 Ko (réduction de 38%).

De nombreux outils existent pour minifier des fichiers JavaScript. C'est le cas de [YUI Compressor](#) ou [JSMIn](#), recommandés par Google.

0/100

#2490

! Allégez la favicon

La favicon se doit d'être la plus légère possible.

La favicon sur le web

Cette petite image permet de représenter votre page via une petite icône, sur l'onglet du navigateur par exemple.

Que faire ?

Une favicon doit idéalement être < 10ko.

La favicon (<http://todolistme.net/favicon.ico>) dispose d'un poids trop important (17ko).

Vos principaux axes d'optimisation

0/100

#2348

Vous devriez utiliser Universal Analytics

Depuis avril 2014, Google a mis à disposition de ses utilisateurs une nouvelle version de Google Analytics, nommée Universal Analytics. Cet outil permet entre autre de mesurer les interactions des clients à travers les différents périphériques et plates-formes utilisés.

Vous n'utilisez pas encore ces [nouvelles propriétés](#). Suivez [les instructions suivantes](#) afin de mettre à jour Google Analytics.

Notez que si besoin, vous pouvez mettre en place à la fois l'ancienne (ga.js/dc.js) et la nouvelle version (analytics.js) de Google Analytics, celles-ci pouvant très bien cohabiter.

Vos principaux axes d'optimisation

0/100

#2555

Envisagez d'utiliser jQuery 1.12

Votre page web utilise jQuery 1.9.1. Vous devriez migrer vers la dernière version de la branche 1.x : jQuery 1.12, qui contient plusieurs correctifs de bugs et de sécurité.

Dois-je migrer vers la dernière version de jQuery ?

La migration d'une version 1. x de jQuery vers la dernière version (3.x) peut avoir de nombreux impacts inattendus et implique une perte de compatibilité avec les navigateurs plus anciens. Vous ne devriez envisager d'abandonner jQuery 1.x que dans le cadre d'une refonte complète de votre infrastructure Front-End. Lors de la réflexion préalable à cette refonte, vous découvrirez peut-être que [vous n'avez pas besoin de jQuery](#).