



TMS – Guide d'implémentation

v 1
06/05/2014

SOMMAIRE

1. Présentation de TagCommander.....	4
1.1 Qu'est-ce que TagCommander ?	4
2. Structure du tag.....	5
2.1 La version script.....	5
2.2 Cas spécifique : le noscript (optionnel)	5
2.3 Cas spécifique : Implémentation via DOM Scraping ou via plug-in webanalytics.....	6
3. Implémentation du tag.....	7
3.1 Création du Datalayer	7
3.1.1 Description	7
3.1.2 Position du Datalayer dans la page.....	7
3.1.3 Syntaxe	7
3.1.4 Cas spécifiques	8
3.2 Implémentation des conteneurs	9
3.2.1 Description	9
3.2.2 Position des conteneurs dans la page.....	10
3.2.3 Syntaxe	10
3.2.4 Cas spécifiques	10
3.3 Déclaration des events	12
3.3.1 Description	12
3.3.2 Syntaxe	12
3.4 Implémentation sans rechargement de page (AJAX).....	13
3.4.1 Option 1 : via process interne	13
3.4.2 Option 2 : utilisation du jQuery	14
3.4.3 Option 3: Utiliser les événements TagCommander	14
4. Recette de l'implémentation	15
4.1 Tester le Datalayer	15
4.1.1 Démarche à suivre	15
4.1.2 Débogage	16
4.2 Tester les conteneurs	16
4.2.1 Démarche à suivre	16
4.2.2 Débogage	17
4.3 Tester les événements.....	18
4.3.1 Démarche à suivre	18

4.3.2 Débogage	18
5. Mode de synchronisation du conteneur	19
5.1 Le fonctionnement	19
5.1.1 Description	19
5.2 Les différents modes	19
5.2.1 Mise à jour manuelle (téléchargement direct ou envoi par email)	19
5.2.2 Mise à jour semi-automatisée	20
5.2.3 Mise à jour automatisée	20
5.2.4 Cas particulier : le CDN (content delivery network)	20
Supports et contacts	21

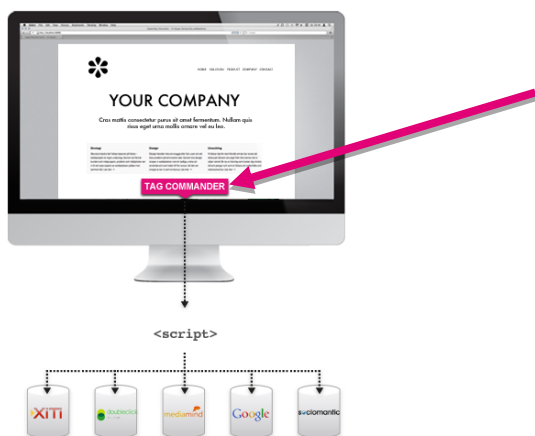
1. Présentation de TagCommander

1.1 Qu'est-ce que TagCommander ?

TagCommander propose un système de taggage universel via un conteneur de tags : l'Universal Tag Container.

Ce dernier permet à des utilisateurs non techniques de piloter les solutions utilisées sur un site via une interface web. Grâce à cela, un utilisateur lambda est capable d'ajouter, supprimer et d'interagir directement avec n'importe quel tag sans avoir à toucher au code.

Pour fonctionner, une implémentation de notre conteneur de tag dans le code en dur est nécessaire.



2. Structure du tag

2.1 La version script

Le tag TagCommander peut-être divisé en deux parties :

- le « Datalayer » qui consiste en une librairie de variables renseignée par l'intégrateur.
- le « Conteneur » qui est un fichier JavaScript permettant de faire fonctionner TagCommander. Ce dernier est hébergé sur vos serveurs ou par TagCommander (CDN).

Illustration



2.3 Cas spécifique : Implémentation via DOM Scraping ou via plug-in webanalytics

Dans ces deux cas spécifiques, le datalayer est créé par les équipes TagCommander ou réutilise **le code déjà présent dans le code source du site**.

Dans ce type de configuration, il vous suffit d'ajouter les appels au conteneur TagCommander dans le code source. Vous pouvez donc passer directement à l'**implémentation des conteneurs** (section 3.B).

3. Implémentation du tag

3.1 Création du Datalayer

3.1.1 Description

Le DataLayer est un objet JavaScript composé d'une liste de variables permettant à la fois aux solutions tierces embarquées dans TagCommander de recevoir des informations sur le site et les visiteurs, et à l'utilisateur final de la solution de créer des règles servant à déclencher les tags selon des conditions particulières (pages produits, homepage, visiteurs loggés etc.).

Le Datalayer devra être renseigné par le prestataire technique du client et intégré directement en dur dans le code du site embarquant TagCommander.

La liste des variables à intégrer dans le Datalayer varie généralement d'un projet à l'autre (se renseigner auprès de son consultant TagCommander) et n'est pas limitée en nombre.

3.1.2 Position du Datalayer dans la page

Le Datalayer **doit être implémenté avant** les conteneurs TagCommander. Dans le cas contraire, tous les conteneurs présents avant le chargement du Datalayer ne seront pas en mesure d'utiliser son contenu.

3.1.3 Syntaxe

Afin de ne pas interagir avec le code déjà présent sur le site, TagCommander préconise d'utiliser **l'objet littéral** suivant :

```
tc_vars = {} ;
```

Une fois la déclaration faite, l'ajout de variables se fait en suivant cette syntaxe :

```
Key : "value"
```

Key correspond au nom de la variable et **value** à sa valeur. Ce qui donne par exemple :

```
tc_vars = {  
  env template : "homepage"  
}
```

Pour séparer les variables, il suffit de rajouter une virgule après la valeur, de la façon suivante :

```
tc_vars = {
```

```
env_template : "homepage" ,
page name : "the basket page"
}
```



Attention :

Il ne faut pas mettre de virgule supplémentaire avant le « } » après la valeur de la dernière variable sinon cela génèrera une erreur.

Dans l'exemple suivant, la syntaxe est donc incorrecte :

```
tc vars = {
  env template : "homepage",
  page_name : "the_basket_page" ,
}
```

Les quotes à l'intérieur des variables doivent être échappées :

```
tc vars = {
  env template : "homepage",
  page_name : "L'histoire du \"datalayer\""
}
```

3.1.4 Cas spécifiques

Intégrer un tableau (array) dans le datalayer

Pour intégrer un sous-tableau dans le datalayer (ex : order_product, list_product,...), il suffit de suivre la syntaxe ci-dessous :

```
tc vars = {
order product : [{
  price: "123",
  id: "k9odsqo"
}]
}
```

Pour chaque produit ajouté, le contenu du tableau devra être répété et séparé par une virgule :

```
tc vars = {
order_product : [{
  price: "123",
  id: "k9odsqo"
} , {
  price: "12",
  id: "Pmdslql"
}
```



```
} ]
}
```



Attention :

Ne pas mettre de virgule supplémentaire avant le « } » après la valeur de la dernière variable sinon cela génèrera une erreur.

Dans l'exemple suivant, la syntaxe est donc incorrecte :

```
tc vars = {
  order product : [{
    price: "123",
    id: "k9odsqo"
  }, {
    price: "12",
    id: "Pmdslql"
  } , ]
}
```

Ne pas utiliser le signe « égal » pour déclarer un tableau.

Dans l'exemple suivant, la syntaxe est donc incorrecte :

```
tc vars = {
  order product = [{
    price: "123",
    id: "k9odsqo"
  }, {
    price: "12",
    id: "Pmdslql"
  } ]
}
```

Pour la recette, se référer au [chapitre 4.A.a](#)

Gestion des caractères spéciaux et autres alphabets

TagCommander utilise le format UTF-8, et est donc compatible avec tous les alphabets et caractères spéciaux.

3.2 Implémentation des conteneurs

3.2.1 Description

Le conteneur de tags est un fichier JavaScript qui a deux objectifs :

- Faire fonctionner TagCommander
- Exécuter les solutions (tags) partenaires embarquées dans TagCommander

Il est possible, et c'est souvent le cas, d'avoir plusieurs conteneurs sur un même site, voir une même page.

3.2.2 Position des conteneurs dans la page

Les conteneurs peuvent être placés à différents endroits de la page selon leurs utilités :

Type de conteneur	Emplacement dans la page
AB Test (optionnel)	Dans le <head>
Générique (tags de conversion, retargeting, webanalytics, etc)	Avant la balise fermante </body>
Bannière publicitaire (optionnel)	Dans la <div> de la bannière
Vidéo (optionnel)	A définir avec le consultant



Attention :

- Il n'est pas obligatoire de placer l'appel aux conteneurs (donc l'appel aux fichiers JavaScript) juste à la suite de l'appel au Datalayer. En revanche les conteneurs doivent toujours être appelés après le Datalayer dans la page.
- L'appel au script doit se faire depuis la racine de votre document et non depuis un élément enfant : <div>, <p> etc.
- Pour la recette, se référer au [chapitre 4.B.a](#)

3.2.3 Syntaxe

La syntaxe à respecter est la suivante :

```
<script type="text/javascript" src="mon-conteneur.js"></script>
```

La partie « **mon-conteneur.js** » doit être remplacée par le véritable emplacement du conteneur sur votre serveur (ou sur le nôtre).

3.2.4 Cas spécifiques

Empêcher la mise en cache du conteneur par le navigateur : le versioning

A chaque modification faite dans l'interface TagCommander, l'utilisateur devra mettre à jour le conteneur en production en remplaçant la version existante par la nouvelle version disponible dans l'interface (ex : « mon-conteneur.js »).

Afin d'éviter des problèmes de mise en cache par le navigateur (le cache serveur pouvant être géré directement lors du déploiement du fichier depuis l'interface ou par les équipes techniques du client), il est recommandé de faire suivre le nom du conteneur d'un numéro de version, comme suit :

```
<script type="text/javascript" src="mon-conteneur.js?v17.11"></script>
```



Attention :

Veillez à bien ajouter un point d'interrogation entre le nom du fichier et le numéro de versioning (le numéro ajouté). Dans le cas contraire, le navigateur interprètera le numéro comme faisant partie intégrante du nom du conteneur.

```
<script type="text/javascript" src="mon-conteneur.js?v17.11"></script>
```

Chargement du conteneur en Synchrone ou Asynchrone ?

Les conteneurs TagCommander **doivent être intégrés de manière synchrone**. En effet certaines solutions susceptibles d'être implémentés dans les conteneurs ne sont pas compatibles avec le fait d'être chargées en asynchrone.

Conteneur bannière

Avant de commencer toute implémentation d'un conteneur de bannière, merci de contacter votre consultant TagCommander.

L'implémentation de conteneurs de bannières publicitaires peut se faire de deux manières distinctes :

- En implémentant un seul conteneur sur la page : avec cette méthode, l'appel des tags dans les différents emplacements publicitaires est géré directement depuis le conteneur dans l'interface TagCommander.
- En implémentant un conteneur spécifique par emplacement publicitaire (il y aura donc plusieurs conteneurs sur la page) :

```
<div class="top adcontainer" id="EAS 134">
<script src="mon-conteneur-banniere"></script>
</div>
```



Attention :

Certaines solutions ne sont pas compatibles avec la première méthode (méthode d'« injection depuis le conteneur principal »).

Conteneur vidéo

L'implémentation des conteneurs vidéo varie en fonction du lecteur utilisé. Nous vous recommandons de [contacter votre consultant TagCommander](#).


3.3 Déclaration des events

3.3.1 Description

TagCommander permet de gérer des clics et des événements. Pour cela il faut appeler la fonction prédéfinie `tc_events_XXX()` avec un événement onclick comme lien html, ou n'importe quel autre événement JavaScript.

Le `tc_events_XXX` appelle le conteneur XXX et est alimenté avec les paramètres destinés à répondre aux besoins des solutions embarquées dans le conteneur.

Le XXX est à remplacer par le numéro du conteneur concerné. Ce dernier vous sera communiqué par votre consultant TagCommander mais peut aussi être trouvé dans l'interface via l'URL :



3.3.2 Syntaxe

La syntaxe à utiliser est la suivante :

```
<a href="http://www.link.com" target=" blank" onclick="javascript:return
tc events XXX (this,"CLICK",{
'VARIABLE 1':'VALEUR 1',
'VARIABLE 2':'VALEUR 2',
...,
'VARIABLE N':'VALEUR N'
});"> mon nom de lien </a>
```

- **XXX** : doit-être remplacé par le numéro de conteneur (fourni par votre consultant)
- **CLICK** : Il s'agit de l'identifiant de l'événement. Il peut être remplacé par n'importe quel texte ou nombre.
- `{'VARIABLE_1':'VALEUR_1',...}` : Ce sont des variables complémentaires de l'événement, nécessaires pour le bon fonctionnement des tags embarqués dans TagCommander :
 - Les **VARIABLES** doivent être remplacés par le nom des variables
 - Les **VALEURS** doivent être remplacées par leurs valeurs

Exemple :

```
<a href="http://www.link.com" target=" blank" onclick="javascript:return tc events 3
(this,"add_to_cart",{
```

```
'product id':'EC654',
'product name':'robe bleue',
'emplacement':'fiche produit'
});"> mon nom de lien </a>
```



Attention :

Ne pas mettre de virgule supplémentaire avant le « } » après la valeur de la dernière variable sinon cela génèrera une erreur.

Dans l'exemple suivant, la syntaxe est donc incorrecte :

```
<a href="http://www.link.com" target=" blank" onclick="javascript:return
tc events 3 (this,"add to cart",{
'product id':'EC654',
'product name':'robe bleu',
'emplacement':'fiche produit'
});"> mon nom de lien </a>
```

Il est fortement recommandé de laisser l'état « **return** ». Certains outils en ont besoin pour fonctionner correctement.

```
<a href="http://www.link.com" target=" blank" onclick="javascript:return
tc events 3 (this,"add to cart",{
'product id':'EC654',
'product name':'robe bleu',
'emplacement':'fiche produit'
});"> mon nom de lien </a>
```

Il est possible de passer autant de variables que nécessaire dans la fonction tc_events.

3.4 Implémentation sans rechargement de page (AJAX)

Par défaut les tags embarqués dans TagCommander sont configurés pour être exécutés au chargement des pages.

Sur des sites utilisant de l'ajax tout au long de la navigation (ou même uniquement pour des ajouts panier ou des filtres de recherche par exemple), une implémentation classique du conteneur ne peut pas fonctionner car elle ne permet pas le rechargement du conteneur à chaque action ou rechargement de page. Les tags ne peuvent ainsi plus être déclenchés sur les pages sur lesquelles ils doivent être appelés.

L'implémentation de TagCommander sur un site en ajax (partiellement ou full ajax) requiert par conséquent une attention toute particulière. TagCommander propose 3 solutions d'implémentation pour les sites en ajax. Le principe consiste à chaque fois à recharger le conteneur et à ré-initialiser le data layer.

3.4.1 Option 1 : via process interne

Le conteneur doit être rechargé et l'objet tc_vars réinitialisé à chaque requête Ajax conformément aux technologies et contraintes propres à l'architecture client.

3.4.2 Option 2 : utilisation du jQuery

Si le site contient une librairie jQuery, il est alors possible d'ajouter d'une fonction afin de rappeler le datalayer et les conteneurs sur les événements à traquer :

```
$("#buttonnextsteps").click(function() {
'VARIABLE 1': 'VALEUR 1',
'VARIABLE 2': 'VALEUR 2',
'VARIABLE 3': 'VALEUR 3'
...
$.getScript("mon-conteneur.js");
});
```

3.4.3 Option 3: Utiliser les événements TagCommander

Il est également possible d'appeler la fonction tc_events_XXX à chaque rechargement d'éléments sur la page :

```
<a href="http://www.link.com" target=" blank" onclick="javascript:return
tc events xxx (this,"CLICK",{
'VARIABLE 1': 'VALEUR 1',
'VARIABLE 2': 'VALEUR 2',
...
'VARIABLE N': 'VALEUR N'
});"> mon nom de lien </a>
```

Voir le paragraphe sur la description des [événements TagCommander](#).

4. Recette de l'implémentation

Afin de pouvoir recetter l'implémentation de TagCommander, le conteneur ou les conteneurs qui vous ont été livrés contiennent des tags de recette (console.log) qui seront ensuite supprimés lors du passage en production. Ces tags permettent d'afficher les informations relatives au datalayer lors du chargement des pages ainsi que celles relatives aux fonctions tc_events à chaque activation d'événement.

4.1 Tester le Datalayer

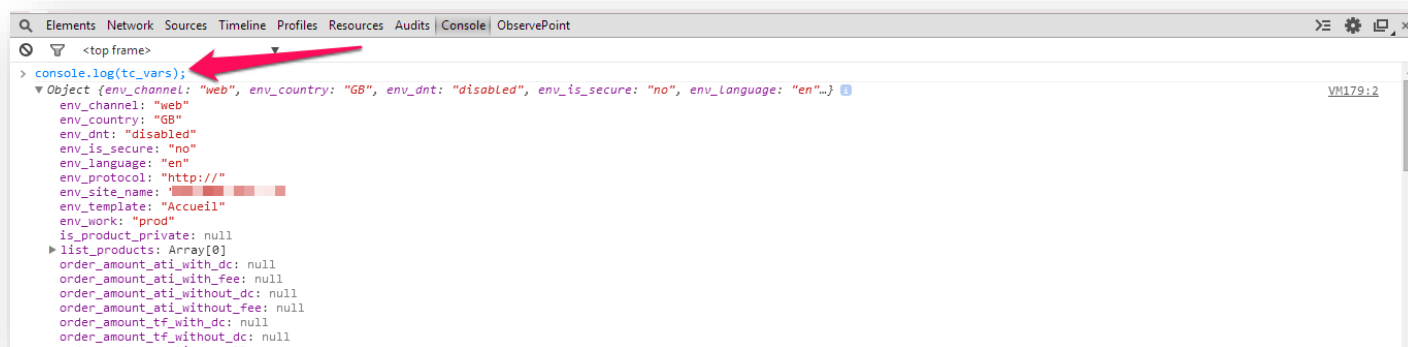
4.1.1 Démarche à suivre

La méthode la plus sûre pour vérifier la bonne déclaration des variables dans le navigateur est d'utiliser la console d'un débogueur (Firefox/Chrome de préférence). Pour cela :

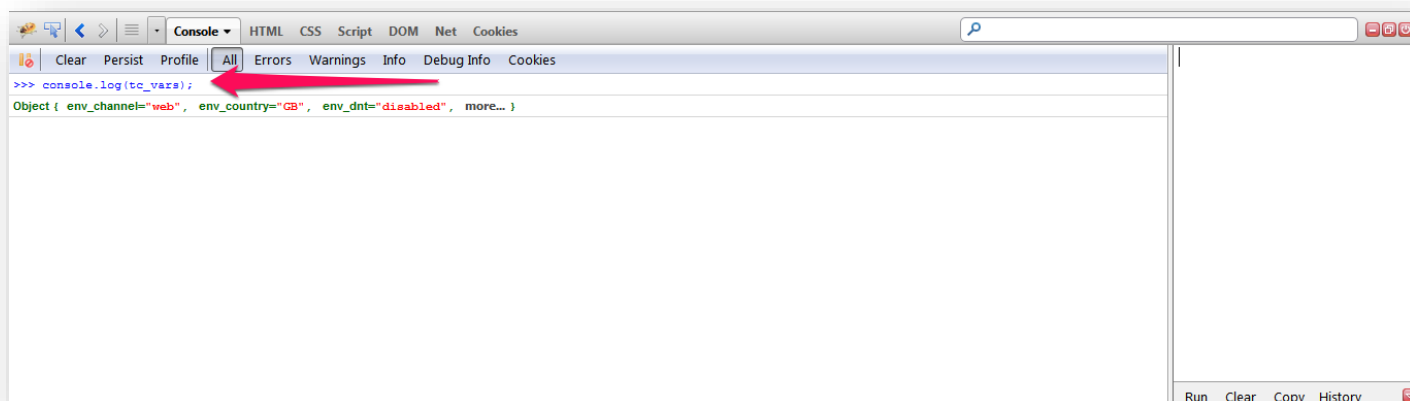
- Assurez-vous d'avoir un débogueur installé et prêt à l'emploi (Firebug,...)
- Ouvrez le débogueur
- Allez dans l'onglet « Console »
- Rechargez la page

Vous devriez voir apparaître les informations relatives au datalayer grâce au tag de recette.

Chrome :



- Firefox



4.1.2 Débogage

Si ce n'est pas le cas :

- Veillez à recharger la page une fois le débogueur lancé (parfois nécessaire pour prendre en compte la page)
- Vérifiez que le Datalayer est bien présent dans le code source
- Vérifiez que la syntaxe est conforme au point 3.A :
 - L'objet littéral `tc_vars = {}` est-il bien déclaré ?
 - Les variables suivent-elles bien la syntaxe suivante : « key : "value" »
 - N'il-y-a-t-il pas de virgule en trop à la fin du datalayer ?
 - N'il-y-a-t-il pas de caractères spéciaux non encodés/échappés qui casseraient le code ? (Parfois une simple apostrophe non échappée suffit à casser le datalayer).

4.2 Tester les conteneurs

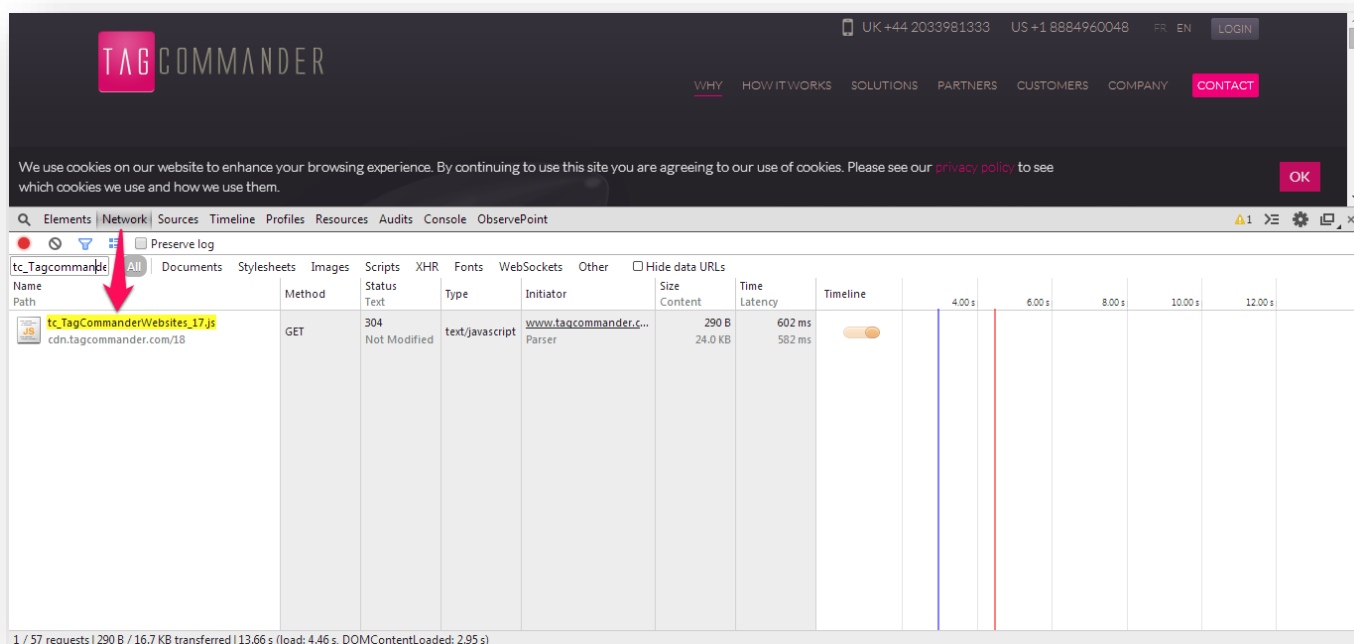
4.2.1 Démarche à suivre

La méthode la plus sûre est de vérifier dans la console que les conteneurs soient bien chargés. Pour cela :

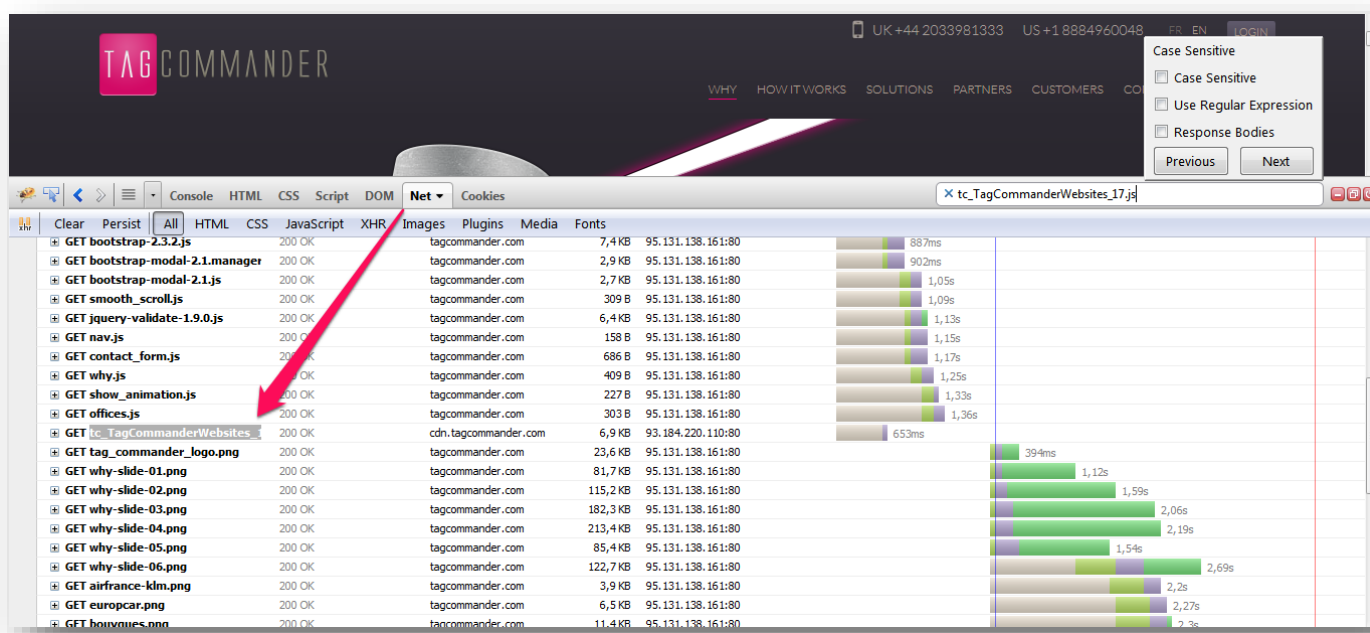
- Assurez-vous d'avoir un débogueur installé et prêt à l'emploi (Firebug,...)
- Ouvrez le débogueur et rechargez la page
- Cliquez sur l'onglet « Network » ou « Net » en fonction du navigateur.
- Recherchez le nom du conteneur appelé

Si ce dernier est bien appelé, il devrait apparaître dans la console :

- Chrome :



- Firefox :



4.2.2 Débogage

Si ce n'est pas le cas :

- Veuillez à recharger la page une fois le débogueur lancé (parfois nécessaire pour prendre en compte la page)
- Vérifiez que le conteneur est bien présent dans le code source de la page
- Vérifiez la syntaxe conformément au point 3.B
- Assurez-vous que le chemin vers le conteneur ou le nom du conteneur est correct.

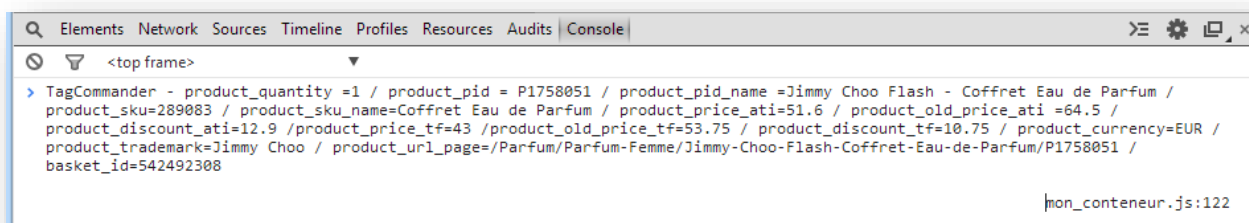
4.3 Tester les événements

4.3.1 Démarche à suivre

La méthode la plus sûre est de vérifier dans la console que les événements se déclenchent bien à chaque clic/action. Pour cela :

- Assurez-vous d'avoir un débogueur installé et prêt à l'emploi (Firebug,...)
- Ouvrez le débogueur
- Cliquez sur l'onglet « console » du navigateur.
- Cliquez sur l'événement à tester

Si ce dernier est bien appelé les informations liées à l'événement devraient apparaître dans la console :



4.3.2 Débogage

Si ce n'est pas le cas :

- Veillez à recharger la page une fois le débogueur lancé (parfois nécessaire pour prendre en compte la page)
- Vérifiez que l'évènement est bien présent dans le code source de la page
- Vérifiez la syntaxe conformément au point 3.C :
 - Le nom de la fonction est-il bien « tc_event*s* » et non « tc_event » ?
 - Les variables suivent-elles bien la syntaxe suivante : « key : "value" »
 - N'il-y-a-t-il pas de virgule en trop à la fin de l'événement ?
 - N'il-y-a-t-il pas de caractères spéciaux non encodés/échappés qui casseraient le code ? (Parfois une simple apostrophe non échappée suffit à casser l'événement).
- Assurez-vous que le numéro du conteneur appelé dans l'événement est bien le bon

5. Mode de synchronisation du conteneur

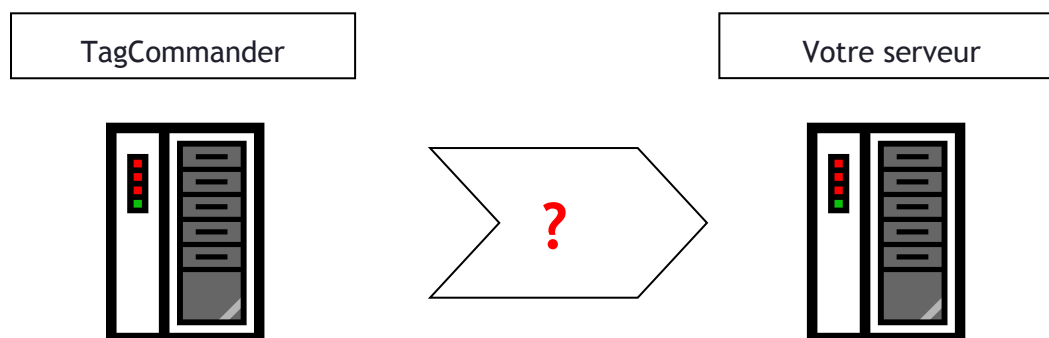
5.1 Le fonctionnement

5.1.1 Description

Une des forces de TagCommander est de fusionner tags et intelligence embarquée en les combinant en un seul fichier JavaScript.

Pour garder le contrôle sur la délivrabilité du fichier JavaScript, nous vous recommandons de l'héberger sur vos serveurs. Une passerelle entre ce fichier JavaScript et le fichier maître hébergé chez TagCommander devra alors être mise en place afin de vous assurer d'utiliser la version la plus récente de ce fichier (version générée après chaque modification dans l'interface utilisateur de TagCommander).

Plusieurs méthodes sont disponibles afin de mettre à jour le fichier JS.



5.2 Les différents modes

5.2.1 Mise à jour manuelle (téléchargement direct ou envoi par email)

Pour chaque conteneur de tags, vous pouvez :

- Accéder au fichier JS depuis l'interface TagCommander pour télécharger manuellement la version à déployer
- Vous faire envoyer le fichier par email à chaque déploiement.

Cela sous-entend un dépôt manuel sur votre serveur ce qui complexifie la procédure de déploiement et peut réduire la fréquence de mise à jour.

5.2.2 Mise à jour semi-automatisée

Afin de simplifier la tâche, nous pouvons paramétrer des comptes FTP/sFTP/SSH depuis l'interface TagCommander afin de contrôler la mise à jour des conteneurs via l'interface.

Une fois ces paramètres définis, vous devenez autonome dans le déploiement puisque la mise à jour se fait alors en un clic depuis l'interface.

Pour se faire, vous devez ouvrir un accès serveur au consultant TagCommander afin de pouvoir y déposer les conteneurs à chaque mise à jour.

Voici les informations demandés :

Information requise	Exemple
Host	192.168.1.254
Login	tagcommander
Port	21
Path	/live/js/tagcommander/
Password	p4ssw0rd

5.2.3 Mise à jour automatisée

Le moyen le plus sûr et le plus rapide pour mettre à jour le fichier est de créer un programme (batch/crontab) sur vos serveurs. Son objectif sera de télécharger la dernière version du conteneur de tags à fréquence régulière.

Pour se faire, nous pouvons vous fournir une URL fixe pour chaque conteneur fournissant toujours la dernière version du conteneur.

5.2.4 Cas particulier : le CDN (content delivery network)

Si vous utilisez nos CDN, il n'y a rien à faire côté client pour mettre à jour les conteneurs.

Supports et contacts

Customer Support

support@tagcommander.com

Tel: +33 (0) 1.43.12.33.89

TAGCOMMANDER . <http://www.tagcommander.com>
12 rue Vignon . 75009 Paris . France