Html5 : intégrer une vidéo compatible avec tous les navigateurs

Author : pierrehirel

La version 5 du langage html a apporté le support natif de la vidéo, qui n'était auparavant possible qu'à travers un plugin (le plus répandu étant Adobe Flash). Ce billet propose une méthode pour intégrer une vidéo dans une page Web, qui puisse être affichée par le plus grand nombre de navigateurs possible.

Html5, balise <video> et codecs

Le langage html, mis au point au début des années 1990 par Tim Berners-Lee, est un langage structuré par des balises. Une balise peut indiquer par exemple si une partie du texte est un lien vers une autre page Web (lien hypertexte), un titre, un paragraphe, un tableau, etc. Au départ ce langage était essentiellement concentré sur le texte, puis est apparue la possibilité d'intégrer des images fixes. La démocratisation d'Internet et l'augmentation des débits des lignes des abonnés à la fin des années 1990, ainsi que l'amélioration des algorithmes de compression audio et vidéo, ont permis de diffuser des vidéos par Internet, et éventuellement de les intégrer dans des pages Web. Comme le langage html ne proposait pas de balise standard pour intégrer des vidéos, celles-ci étaient lues à travers le programme Adobe Flash. Ceci requiert une application installée sur le serveur pour diffuser la vidéo, et un plug-in pour navigateur que les utilisateurs devaient installer pour recevoir et lire la vidéo. La plupart des sites Web proposant des vidéos ont été conçus autour de Flash, comme Youtube, Dailymotion, Vimeo, etc.

Le html5 a apporté, entre autres, le support natif de la vidéo. Autrement dit il est désormais possible d'intégrer une vidéo grâce à la balise <video>, sans avoir besoin de faire appel à un plug-in. En revanche, si la syntaxe de la balise en elle-même est bien définie, rien n'est exigé pour ce qui concerne le format de fichier audio/vidéo. La raison est compréhensible : les codecs audio et vidéo évoluent avec le temps (en l'espace de dix ans on est passé du mpeg au mpeg4, du wav non compressé au mp3), figer un codec particulier dans le standard du Web risquerait de freiner l'évolution de ces codecs. Ce qui est moins compréhensible, c'est que si les différents navigateurs Web (Internet Explorer, Firefox, Safari, Chrome, Opera…) supportent bien la balise <video>, en revanche ils ne supportent pas les mêmes codecs. Le tableau ci-dessous résume la situation : il n'existe aucun codec qui garantit qu'une vidéo sera bien lue par tous les navigateurs. Le format h264, propriété du groupe MPEG-LA, est soumis au paiement de royalties que tous les éditeurs de navigateurs ne sont pas enclins à payer. On peut noter que Google supportait ce format dans les versions 4.0 à 16.0 de Chrome, mais l'a retiré dans les versions ultérieures. Ogg/Theora et WebM sont tous deux des formats vidéo libres, reconnus par Firefox, Chrome et Opera, mais non supportés par Internet Explorer et Safari.

Navigateur	H.264/MP4	OGG Theora	WebM
Mozilla Firefox	**non**	**3.5**	**4.0**
Opera	**non**	**10.5**	**10.6**
Microsoft Internet Explorer	**9.0**	**non**	**oui si codecs installés**
Google Chrome	**4.0 à 16.0**	**4.0**	**6.0**
Apple Safari	**5.0**	**non**	**non**

Dès lors, une question se pose au développeur : comment concevoir une page Web qui propose une vidéo qui s'affiche correctement sur tous les navigateurs ? Il n'existe malheureusement pas de réponse simple, il est nécessaire de proposer la vidéo dans plusieurs formats.

1. Conversion de la vidéo

Commençons donc par convertir la vidéo dans les différents formats. Nous utiliserons ici le programme libre ffmpeg, disponible gratuitement pour Windows, MacOS et Linux. Supposons que le fichier vidéo s'intitule « ma_video.avi ». Pour transcoder la vidéo au format x264, supporté par Internet Explorer et Apple Safari :

client:~$ ffmpeg -i ma_video.avi -vcodec libx264 ma_video.mp4

Convertissons-la ensuite au format libre Ogg/Theora, reconnu par Firefox, Opera et Chrome :

client:~$ ffmpeg -i ma_video.avi -vcodec libtheora ma_video.ogv

Enfin convertissons-la au format WebM, format développé par Google et préféré par Chrome, mais également lisible par Firefox et Opera :

client:~$ ffmpeg -i ma_video.avi -b 1000k ma_video.webm

Notez que les options de ffmpeg utilisées ici ne sont que des exemples. Vous pouvez vous référer à la documentation de ffmpeg, ou bien utiliser un autre logiciel pour encoder votre vidéo -ce n'est pas le but de ce billet.

Les formats Ogg/Theora et WebM sont quelque peu redondants puisqu'ils sont supportés par les mêmes navigateurs. Le tableau ci-dessus montre cependant que la compatibilité n'est pas complète avec les anciennes versions des navigateurs : Ogg/Theora est supporté depuis Firefox 3.5 et Chrome 4.0, tandis que WebM est supporté depuis Firefox 4.0 et Chrome 6.0. De plus, si les deux formats sont proposés alors Chrome préférera utiliser le format WebM alors que Firefox préférera l'Ogg/Theora. À vous de décider quel(s) flux proposer en fonction de la qualité de la vidéo, de la bande passante utilisée, etc.

2. Intégration dans la balise <video> dans la page Web

La balise <video> est conçue pour être souple, et pour proposer plusieurs flux vidéos. Chaque navigateur, en arrivant sur la page, choisira le format qu'il préfère. Le contenu de la balise se présente de la manière suivante :

page_video.html

```
<video poster="ma_video.jpg" controls preload >
<source src="ma_video.mp4" type="video/mp4" />
<source src="ma_video.webm" type="video/webm" />
<source src="ma_video.ogv" type="video/ogg" />
</video>
```

L'attribut « poster » dans la balise <video> est optionnel, il peut être utilisé pour afficher une image le temps que le navigateur charge le lecteur vidéo. Si cet attribut est omis, un cadre blanc ou gris apparaîtra. L'attribut « control » autorise le contrôle de la vidéo : bouton play/pause, navigation, mode plein écran. Enfin l'attribut « preload » indique au navigateur qu'il peut précharger la vidéo afin qu'elle démarre plus rapidement lorsque l'utilisateur appuie sur le bouton « play ». Cet attribut est recommandé s'il y a de grandes chances pour que

l'utilisateur regarde effectivement la vidéo ; si au contraire on s'attend à ce que la plupart des visiteurs du site ne regardent pas la vidéo, cet attribut peut être omis pour économiser la bande passante.

Les trois lignes « source » suivantes indiquent les emplacements des trois vidéos et leurs formats. Chaque navigateur choisira son format préféré pour afficher la vidéo.

3. Assurer la compatibilité avec les vieux navigateurs ne supportant pas la balise <video>

Il est possible qu'un certain nombre de visiteurs du site Web utilisent encore une vieille version de leur navigateur, incapable d'interpréter la balise <video>. Il est possible de proposer à ces utilisateurs une solution « à l'ancienne », en employant un player Flash.

Pour proposer la vidéo au format Flash, il faut d'abord l'encoder dans ce format (.flv) :

client:~$ ffmpeg -i ma_video.avi -vcodec flv ma_video.flv

Ensuite, comme expliqué dans l'introduction, le serveur doit disposer d'une application pour diffuser la vidéo. Utilisons le player proposé gratuitement par Alsacreations : téléchargez-le et placez le fichier « dewtube.swf » sur votre site Web.

L'intégration de ce player Flash dans la page Web se fait via la balise <object>, que l'on va inclure ici à l'intérieur de la balise <video> conçue précédemment :

page_video.html

```
<video poster="ma_video.jpg" controls preload >
<source src="ma_video.mp4" type="video/mp4" />
<source src="ma_video.webm" type="video/webm" />
<source src="ma_video.ogv" type="video/ogg" />
<object type="application/x-shockwave-flash" data="dewtube.swf?movie=ma_video.flv" width="400"
height="300">
<param name="movie" value="dewtube.swf?movie=ma_video.flv" />
<img src="ma_video.jpg" alt="Ma vidéo" title="Ma vidéo" class="figure" />
<p>Ici devrait se trouver ma vidéo, mais votre navigateur ne peut pas l'afficher.</p>
</object>
</video>
```

L'attribut « data » commence par pointer vers le player (dewtube.swf) directement suivi d'une option (?movie=) indiquant le nom du fichier vidéo flv à lire. Il est également possible de spécifier une taille à ce player, afin par exemple qu'il occupe la même place que celle qu'occuperait la balise <video>.

Il est également possible que le visiteur ait un vieux navigateur qui ne dispose pas de Flash. Dans ce cas il sera impossible d'afficher une vidéo, mais il est possible de la remplacer par une image statique, par exemple une image tirée de la vidéo. C'est ce qui a été fait dans l'exemple ci-dessus grâce à l'ajout d'une balise <img /> à l'intérieur de la balise <object>, qui affichera une image au format jpeg.

Enfin il est aussi possible que le visiteur utilise un navigateur qui ne supporte pas la balise <video>, n'ait pas Flash, et soit incapable d'afficher des images -c'est le cas par exemple des navigateurs en ligne de commande comme Lynx. Il est aussi possible que vos fichiers vidéo (ainsi que l'image) soient inaccessibles pour une raison ou une autre (mauvais nom de fichier, fichier corrompu…). Pour parer à ces possibilités il est recommandé d'afficher un texte à l'endroit où devrait se trouver la vidéo : c'est ce qui a été fait dans l'exemple ci-dessus avec la dernière balise <p>. Selon les cas, il peut aussi être de bon ton d'afficher un lien proposant de télécharger la vidéo.

Conclusions

À l'heure où les entreprises et associations regroupées dans les W3C (Microsoft, Apple, Google, Mozilla, Opera…) développent ensemble des standards pour que le contenu du Web soit accessible à tous de la même façon, à l'heure où les navigateurs se multiplient et respectent de mieux en mieux ces standards, on pourrait s'attendre à la fin des hacks spécifiques à tel ou tel navigateur, qui étaient nécessaires il y a dix ans tant les standards étaient mal respectés ou inexistants. Cela est en partie vrai avec html5 et CSS3, mais le problème s'est déplacé vers le support des formats audio/vidéo : les grands acteurs n'ont pas réussi à s'accorder pour supporter un codec donné, il est donc nécessaire d'utiliser des « hacks » si l'on souhaite qu'une vidéo soit lue par la plupart des visiteurs d'un site Web. Espérons que cette situation ne soit que temporaire.