

Utilisation de la librairie ming et création d'un lecteur flash vidéo

par maxime.ohayon ([Site Perso](#))

Date de publication : 11/12/2007

Dernière mise à jour : 11/12/2007

La bibliothèque ming permet de créer des animations flash à partir d'objet PHP, cet article montre la création d'un lecteur flash vidéo supportant le streaming.

- I - Ming c'est quoi ?
 - I-A - Description
 - I-B - Mais pour quoi faire ?
 - I-C - But de l'article
- II - Avec un Exemple
 - II-A - Pré-requis
 - II-B - Animation Simple
 - II-C - Avec des Boutons
 - II-D - Utilisation d'un Stream
 - II-E - L'Action Script
 - II-E-1 - Interface
 - II-E-2 - Vidéo et Flux
 - II-E-3 - Gestion du déplacement de la souris sur la barre de progression
 - II-F - Conclusion de l'exemple
- III - Intégration & Outils
 - III-A - Intégration dans une page HTML
 - III-B - Spécification Vidéo
 - III-B-1 - Utilisation ffmpeg
 - III-B-2 - Utilisation de flvmdi
- IV - Conclusion

I - Ming c'est quoi ?

I-A - Description

Ming est une bibliothèque logicielle open-source sous licence GPL qui permet de créer des animations Flash (.swf) sous Windows et Unix. Ming reprend toutes les fonctionnalités de Flash 6 tels que les formes, les boutons, les images, les textes, les actions, le streaming et MP3 et vidéo. De plus cette librairie est utilisée le plus souvent en PHP mais est utilisable depuis d'autre langage tels que C++, Perl, Python ou Ruby.

I-B - Mais pour quoi faire ?

Le gros avantage d'une telle bibliothèque c'est sa gratuité, sa légèreté et sa portabilité. En effet, rares sont les alternatives libre au logiciel Flash de Macromedia, la librairie est multi portable et simple d'utilisation. Cependant, une bonne connaissance du langage Flash et requis pour l'utiliser car tout est dans un script PHP.

I-C - But de l'article

Cet article a pour but de vous faire découvrir quelques possibilités de la bibliothèque logiciel ming au travers d'un exemple : un lecteur de vidéo qui gère le streaming.

Il est destiné à des personnes ayant un niveau de programmation avancé, en effet certains concepts vus sont assez abstraits. De plus une bonne connaissance en Action Script est requise pour comprendre chaque ligne.

Cependant l'article n'est pas destiné à rebuter les débutants, il est même intéressant qu'il utilisent le projet sans se plonger dans tout le codes sources, c'est pour cela que les sources sont disponible à la fin de l'article.

II - Avec un Exemple

II-A - Pré-requis

Pour pouvoir utiliser Ming il faut l'activer au près de votre serveur Apache sous Windows (Wamp, EasyPHP, ...). Ou compiler les sources sous linux (plus d'information sur [site officiel de Ming.](#))

II-B - Animation Simple

La bibliothèque implémente pas moins de 15 objets et environ 150 méthodes. Voici un exemple d'une première animation flash :

FirstFlash.php

```
<?php
$movie = new SWFMovie(7);
$movie->setDimension(320,270);
$movie->setBackground(0,0,0);
$movie->setRate(8);
$movie->save("player.swf");
?>
```

Construction d'un objet Ming SWFMovie avec le mot clé new.

La méthode setDimension() donne deux dimensions à notre animation.

La méthode setBackground() modifie la couleur de fond (0,0,0 correspond au noir).

La méthode setRate() définit le débit des frames de l'animation.

La méthode save("fichier") sauvegarde l'animation dans le fichier passé en paramètre.

Il est aussi possible d'utiliser la méthode output() pour afficher directement l'animation flash dans la page, cependant il ne faudra pas oublier de mettre de le header adéquate. L'animation ainsi créée est n'est pas statique c'est à dire que contrairement à l'exemple précédent elle n'est pas "en dure" sur le serveur.

MethodeOutput.php

```
<?php
header('Content-type: application/x-shockwave-flash');
$movie->output();
?>
```

II-C - Avec des Boutons

On va maintenant ajouter des boutons à notre animations. Pour cela on fabrique une méthode générique en de création de bouton:

createButton

```
<?php
```

createButton

```
function createButton($movie, $name, $loc, $script)
{
    $button = new SWFButton();
    $button->addShape(createImage("control_" . $name . ".png"), SWFBUTTON_UP);
    $button->addShape(createImage("control_" . $name . "_blue.png"), SWFBUTTON_DOWN | SWFBUTTON_HIT |
    SWFBUTTON_OVER);
    $button->addAction(new SWFAction($script), SWFBUTTON_HIT);
    $item=$movie->add($button);
    $item->moveto($loc,248);
}
?>
```

Cette fonction créer un bouton, lui ajoute des images (de fond et lors du survol), associe une action, l'ajoute à \$movie et le place sur la scène. Voici la fonction create image qui permet de fabriquer un objet image.

createImage

```
<?php
function createImage($img)
{
    $shape = new SWFShape();
    $shape->setRightFill($shape->addFill(new SWFBitmap(fopen($img, "rb"))));
    $shape->drawLine(16,0);
    $shape->drawLine(0,16);
    $shape->drawLine(-16,0);
    $shape->drawLine(0,-16);
    return $shape;
}
?>
```

Puis on appelle la fonction createButton dans notre script principale pour ajouter les boutons de lecture de notre player.

Avec des boutons

```
<?php
createButton($movie, "start", 10, "");
createButton($movie, "pause", 40, "");
createButton($movie, "play", 70, "");
?>
```

A noter que les actions correspondantes ne sont pas encore implémentées et les images de fonds des boutons sont dans le même dossier que notre script PHP.

Voici le résultat avec les boutons.

II-D - Utilisation d'un Stream

Notre projet consiste à réaliser un lecteur vidéo en flash par le principe du streaming, c'est à dire que l'on regarde les frames au fur et à mesure qu'on les télécharge. Ce procédé n'est possible qu'avec des vidéos FLV (flash vidéos) vous trouverez toutes les informations nécessaires dans le chapitre III pour convertir vos vidéos.

Stream

```
<?php
$stream = new SWFVideoStream();
$stream->setDimension(320, 240);
?>
```

Ici on vient d'instancier un objet SWFVideoStream et on lui donne la dimension de 320 par 240 pixels.

Il faut maintenant ajouter notre flux à la scène.

Ajout du flux dans la scène

```
<?php
$item=$movie->add($stream);
$item->setName("videoStreamItem");
$item->moveto(0,5);
$movie->add(new SWFAction($strAction));
$movie->nextFrame();
?>
```

C'est la méthode "add" de l'objet \$movie qui ajoute le flux à la scène, cette méthode retourne un "SWFDisplayItem" ce qui nous permettra d'accéder à l'objet "\$stream" et de modifier ces propriétés.

La méthode "setName" permet de le nommer pour ensuite pouvoir l'appeler dans un script.

La méthode "moveto" déplace l'objet dans la scène de 0 pixel en abscisse et de 5 en ordonnées.

Maintenant on va se préoccuper des actions effectués par les boutons mais aussi par le flux, le langage qui permet de définir ces actions s'appelle l' Action Script.

C'est la méthode "add" et du constructeur "SWFAction" qui ajoute l'Action Script à l'objet "\$movie" contenu dans la variable "\$strAction".

Enfin la méthode "nextFrame()" déplace la tête de lecture sur la prochaine frame de l'animation.

II-E - L'Action Script

L'Action Script va nous permettre de définir le téléchargement du flux, l'action correspondante à chaque bouton, la lecture de la vidéo et sa barre de défilement.

Tout le code suivant est contenu dans la variable PHP "\$strAction". J'ai fait le choix de ne pas trop rentrer dans les détails concernant certaines méthodes car il s'agit d'Action Script et cet article ne vise pas à faire de l'Action Script.

II-E-1 - Interface

On veut afficher un message 'Appuyer sur play' pour avertir l'utilisateur que la vidéo ne se lancera que lorsqu'il aura appuyer sur le bouton play, cependant le téléchargement aura déjà commencé.

```
<?php
$strAction ='this.createTextField(\'video_txt\', 999, 120,120, 100, 100);
video_txt.autoSize = 'left';
video_txt.multiline = true;
video_txt.textColor = 0xeeeeee;
video_txt.text=\'Appuyer sur play ...\' ;
stop();';
?>
```

Ici on déclare une zone de texte et l'on met 'Appuyer sur play ...' dedans avec un fond noir et la possibilité d'écrire sur plusieurs ligne. La fonction "stop()" va permettre d'arrêter la tête de lecture sur cette frame et de ne pas afficher les suivantes.

Maintenant on souhaite insérer une barre de progression qui nous donnera deux informations, la quantité d'information téléchargé et la position de la tête de lecture par rapport à la vidéo. La barre de progression est une simple ligne, la quantité d'information téléchargé est représenté par une ligne un peu plus épaisse superposé sur la ligne précédente. Et enfin la position de la tête de lecture est représentée par un rectangle au dessus de la ligne. Ce rectangle doit pouvoir se déplacer sur la ligne et doit être déplaçable par l'utilisateur.

Barre de défilement et de téléchargement

```
<?php
$strAction+='this.createEmptyMovieClip("progressBar_mc", this.getNextHighestDepth());
progressBar_mc.createEmptyMovieClip("bar_mc", progressBar_mc.getNextHighestDepth());
with (progressBar_mc.bar_mc) {
    _xscale=100;
}

progressBar_mc.createEmptyMovieClip("bar1_mc", progressBar_mc.getNextHighestDepth());
with (progressBar_mc.bar1_mc) {
    lineStyle(1,0xFFFFFFFF,50,true, "none", "round", "miter", 1);
    moveTo(110,256);
    lineTo(311,256);
}

progressBar_mc.createEmptyMovieClip("stroke_mc", progressBar_mc.getNextHighestDepth());
with (progressBar_mc.stroke_mc) {
    lineStyle(0, 0xFFFFFFFF);
    beginFill(0xFFFFFFFF);
    moveTo(110,251);
    lineTo(114,251);
    lineTo(114,261);
    lineTo(110,261);
    lineTo(110,251);
    endFill();
};
?>
```

Au début on créer un "clip" appelé "progressBar_mc" et on rajoute dedans un clip qui sera la barre de progression.

Ensuite on créer une seconde barre, celle de téléchargement que l'on met par-dessus et que l'on définit avec une épaisseur plus importante.

Pour terminer on créer la rectangle qui gèrera le défilement de la tête de lecture.

Avant de s'occuper du déplacement du rectangle, on va s'intéresser au téléchargement de la vidéo.

II-E-2 - Vidéo et Flux

Tout d'abord on va avoir besoin de deux variables locales qui vont stocker la durée de la vidéo et si le rectangle de la barre de progression s'est déplacé.

variables globales

variables globales

```
<?php
$strAction+='var duree=0;
var mvt = false;
videoStream.onMetaData = function(infoObject) {
    duree =infoObject[\'duration\'];
};';
?>
```

La variable durée récupère la durée de la vidéo qui est contenu dans une méta-information du flux (voir chapitre III pour les méta données des vidéos et juste après pour le flux). Une vidéo en streaming est un objet "NetStream", son constructeur prend en paramètre un objet "NetConnection" qui permet de lire un flux vidéo à partir d'une adresse local ou http.

flux et connection

```
<?php
$strAction+='nc=new NetConnection();
nc.connect(null);
videoStream=new NetStream(nc);
videoStreamItem.attachVideo(videoStream);
videoStream.setBufferTime(3);
videoStream.play(_level0.video_path);
videoStream.pause();';
?>
```

Ici on déclare un objet "NetConnection" que l'on met à "null" on créer un flux et on attache ce flux à la scène. C'est la ligne "videoStream.play(_level0.video_path);" qui dis à l'animation flash quelle vidéo jouer, le chemin passer en paramètre correspond à un chemin fictif que l'on pourra définir lors de l'appel de l'animation via les balise HTML.

Au début de l'article on avait créer des boutons mais on ne leurs avaient pas assignés d'actions.

Actions des Boutons

```
<?php
createButton($movie, "start", 10, "_root.videoStream.seek(0);");
createButton($movie, "pause", 40, "_root.videoStream.pause(true);");
createButton($movie, "play", 70, "_root.videoStream.pause(false);video_txt.text='';
nc.connect(null);");
?>
```

Le bouton "start" déplace la tête de lecture au début de la vidéo et lance la vidéo alors que le bouton "pause" demande à la tête de lecture de s'arrêter mais de rester à sa position.

Le bouton "play" arrête la pause s'il y en avait une, il enlève la bannière où il y avait écrit "Appuyer sur play ..." et remet la connexion à "null".

On va maintenant s'intéresser au déplacement du curseur de lecteur (rectangle) en fonction de la tête de lecture.

déplacement du curseur de lecteur

```
<?php
$strAction+='var loaded_interval2 = setInterval(checkVideoPlay, 10, videoStream);
function checkVideoPlay(my_ns){
    var seconds = my_ns.time;
    var pctLu = Math.round(seconds/duree*100);
```

déplacement du curseur de lecteur

```

if(duree!=0)
{
    progressBar_mc.stroke_mc.clear();
    progressBar_mc.stroke_mc.lineStyle(0, 0xFFFFFF);
    progressBar_mc.stroke_mc.beginFill(0xFFFFFF);
        progressBar_mc.stroke_mc.moveTo(110+2*pctLu, 251);
    progressBar_mc.stroke_mc.lineTo(114+2*pctLu, 251);
    progressBar_mc.stroke_mc.lineTo(114+2*pctLu, 261);
    progressBar_mc.stroke_mc.lineTo(110+2*pctLu, 261);
    progressBar_mc.stroke_mc.lineTo(110+2*pctLu, 251);
    progressBar_mc.stroke_mc.endFill();
}
}';
?>
    
```

On définit une variable qui est associée à une fonction qui sera appelée à des intervalles périodiques durant l'exécution de l'animation. Cette "intervalle" va vérifier la position de la tête de lecture est adapté la position du curseur sur la ligne qui représente la durée totale de la vidéo.

On peut faire de même pour faire évoluer la barre de téléchargement:

barre de téléchargement

```

<?php
$strAction+='var loaded_interval = setInterval(checkBytesLoaded, 500, videoStream);
function checkBytesLoaded(my_ns) {
    var pctLoaded = Math.round(my_ns.bytesLoaded/my_ns.bytesTotal*100);
    progressBar_mc.bar_mc.clear();
    progressBar_mc.bar_mc.beginFill(0xFFFFFF, 50);
    progressBar_mc.bar_mc.moveTo(110, 255);
    progressBar_mc.bar_mc.lineTo(110+2*pctLoaded, 255);
    progressBar_mc.bar_mc.lineTo(110+2*pctLoaded, 257);
    progressBar_mc.bar_mc.lineTo(110, 257);
    progressBar_mc.bar_mc.lineTo(110, 255);
    progressBar_mc.bar_mc.endFill();

    if (pctLoaded>=100) {
        clearInterval(loaded_interval);
    }
}';
?>
    
```

II-E-3 - Gestion du déplacement de la souris sur la barre de progression

L'utilisateur doit pouvoir via le rectangle placé sur la barre de progression déplacer la tête de lecteur, c'est pour cela qu'il faut ajouter des événements souris et des fonctions qui gèrent la position par rapport à la barre.

Gestion de la souris

```

<?php
$strAction+='progressBar_mc.stroke_mc.onMouseDown = function () {
    if(_xmouse>=110 && _xmouse<=310 && _ymouse>=251 && _ymouse<=261)
    {
        var seconds = my_ns.time;
        var pctLu = Math.round(seconds/duree*100);
        var point = 110+2*pctLu;
        if(_xmouse>=point && _xmouse<=point+4)
        {
            mvt = true;
        }
    }
}';
?>
    
```

Gestion de la souris

```
    }
    else
    {
        var pos = (_xmouse-110)/2;
        videoStream.seek(pos*duree/100);
    }
};

progressBar_mc.stroke_mc.onMouseDown = function () {
    if(mvt==true)
    {
        var pos = (_xmouse-110)/2;
        videoStream.seek(pos*duree/100);
        mvt=false;
    }
};

progressBar_mc.stroke_mc.onMouseMove = function () {
    if(mvt==true)
    {
        var pos = (_xmouse-110)/2;
        videoStream.seek(pos*duree/100);
    }
};

Mouse.addListener(progressBar_mc.stroke_mc);';
?>
```

L'événement "onMouseDown" est associé à une fonction qui va détecter le fait que le bouton de la souris soit cliqué sur le rectangle (variable "mvt" passe à vrai) .

L'événement "onMouseUp" est associé à une fonction qui va détecter le fait que le bouton de la souris à été relâché, à ce moment là on déplace la tête de lecture.

L'événement "onMouseMove" est associé à une fonction qui va détecter le fait que l'on bouge après avoir cliqué sur le rectangle, à ce moment là on déplace la tête de lecture

Le rectangle est dessiné en fonction de la position de la tête de lecture dans la fonction associé à "loaded_interval2".

A la fin on associe les événements souris à la barre.

II-F - Conclusion de l'exemple

Cet exemple n'est pas du tout trivial c'est pour cela que l'on peut récupérer directement les sources **ici** et les images des boutons **ici**

lorsque l'on exécutera le script cela fabriquera une animation player.swf, voir le chapitre suivant pour ensuite plonger l'animation dans du HTML.

III - Intégration & Outils

Nous verrons dans ce chapitre l'intégration de l'animation flash dans une page html ainsi que les spécifications concernant le format de la vidéo.

III-A - Intégration dans une page HTML

C'est lors de l'intégration dans la page Html que l'on spécifie la vidéo à jouer.

```
<object type="application/x-shockwave-flash"
codebase="http://fpdownload.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#version=8,0,0,0"
width="550" height="400" data="player.swf?video_path=test.flv">
<param name="movie" value="player.swf?video_path=test.flv" />
<param name="quality" value="high" />
</object>
```

C'est la vidéo test.flv qui sera jouée par notre lecteur (voir chapitre suivant sur les spécification des vidéos), on lui indique par le '?video_path=' qui est considéré comme un paramètre de l'animation.

III-B - Spécification Vidéo

Seulement les vidéos au format Flash Vidéo (flv) supporte le streaming, et notre lecteur ne peut lire que ce format de vidéo. Ce format est basé sur un codec similaire au H.263 ou VP6 et propose une qualité correcte. Cependant le problème reste de convertir nos vidéos préférées en Flash Vidéos.

Pour cela il existe plusieurs solution (Ripp-it, Riva Flv Encoder ...). La plupart des solutions sont basées sur ffmpeg, il s'agit d'une collection de logiciels libres dédiés au traitement d'un flux numérique. L'avantage de cette solution c'est qu'elle est présente sous Linux et Windows. De plus il existe un projet appelé ffmpeg-php qui sert à convertir des flux numériques via un script PHP ; cependant cette bibliothèque est disponible uniquement sous Linux.

ffmpeg est accessible [ici](#), cette version est une version très complète (compilé avec la plupart des options et modules disponibles et nécessite une bibliothèque dll : pthreadGC2.dll présente aussi dans le zip).

III-B-1 - Utilisation ffmpeg

L'utilisation de ffmpeg se fait en ligne de commande, la ligne suivante permet de convertir une vidéo en Flash Vidéo dans une qualité correct.

```
ffmpeg.exe -i ma_video.avi -acodec mp3 -ab 128 -f flv -s 800x600 -ar 44100 -aspect 4:3 -b 1000 -r
15 ma_video.flv
```

III-B-2 - Utilisation de flvmdi

La vidéo ainsi produite est bien lisible par notre lecteur, mais cependant elle manque d'informations, des métas-informations (ou tags). En effet notre lecteur connait la durée de la vidéos car elle est stockée dans les premiers bits de notre vidéos.

Il est maintenant important de rajouter ces méta-informations à notre vidéo. Pour cela nous allons utiliser sous Windows flvmdi, malheureusement je n'ai pas trouvé de solution intéressante sous Linux, vous pouvez toujours essayer de l'utiliser sous wine.

```
flvmdi.exe ma_video.flv ma_video2.flv
```

flvmdi est disponible [ici](#) et notre vidéo est fin prête.

IV - Conclusion

La bibliothèque logicielle Ming est donc une alternative très intéressante aux solutions payantes pour réaliser des animations Flash. De plus cette bibliothèque comble les quelques lacunes de Flash, elle permet d'intégrer des animations plus utiles, plus réactives et totalement dynamiques.

