7 La vidéo HD en F4V

Introduction

Depuis la version 10 (CS4), Flash accepte le codec H-264 à travers le format vidéo F4V. Ce codec offre une compression d'une excellente qualité où les artefacts demeurent presque imperceptibles. Cette prouesse est telle que l'on peut désormais déployer une vidéo dans un format étendu (haute résolution) sans perdre en qualité d'affichage.

Cependant, à la différence du format FLV, le format F4V ne gère pas la couche transparente (canal alpha). Le F4V convient donc plus particulièrement à la diffusion de séquences vidéo isolées et non composites (extraits de films, introductions, transitions, décors d'arrière-plan animés, interviews, infographies animées).

Dans ce chapitre, nous allons voir comment encoder au format F4V et distribuer des vidéos de haute qualité vers Flash, y compris vers des versions du lecteur *a priori* incompatibles avec ce format. Nous aborderons également quelques astuces qui permettent de compresser confortablement une vidéo de grande taille sans perdre sur la qualité du rendu.

Pour l'ensemble de ces exemples, nous utilisons des créations originales réalisées par la société gKaster, spécialisée dans le Motion Design, et mises amicalement à notre disposition pour cette démonstration (**www.gKaster.com**).

À l'issue de ce chapitre, vous serez en mesure d'intégrer des vidéos de grande qualité au sein de documents Flash, y compris d'anciennes générations.

Encoder en F4V avec Adobe Media Encoder

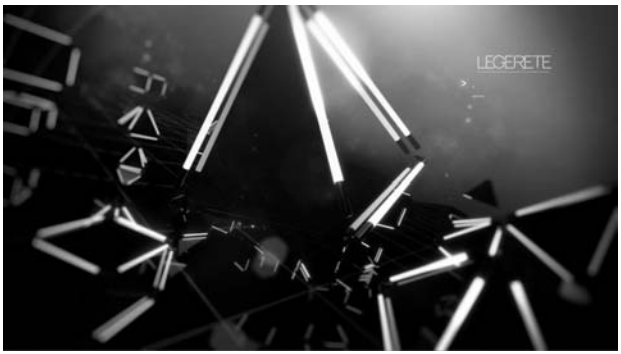
Pour créer un fichier F4V, vous pouvez l'échantillonner avec Adobe Media Encoder. Dans cette section, nous abordons uniquement les réglages spécifiques au format F4V qu'apporte ce logiciel. Si vous voulez découvrir les réglages de base pour la compression d'une vidéo pour Flash, communs au format FLV et F4V, reportez-vous au chapitre précédent, à la section "Échantilloner la vidéo pour Flash".

Exemples > gKaster > gKaster-amusement.mov

Dans cette section, nous détaillons les options d'encodage pour le format F4V, avec le codec H-264.

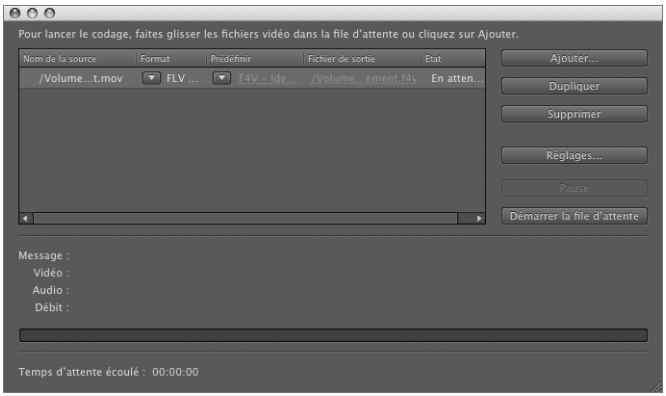
1. Lancez l'application Adobe Media Encoder.
2. Ajoutez, dans la file de rendu, le fichier nommé gKaster-amusement.mov, disponible dans le dossier gKaster des exemples du livre (voir Figure 7.1).

Figure 7.1
Aperçu de la vidéo gKaster-amusement.mov.



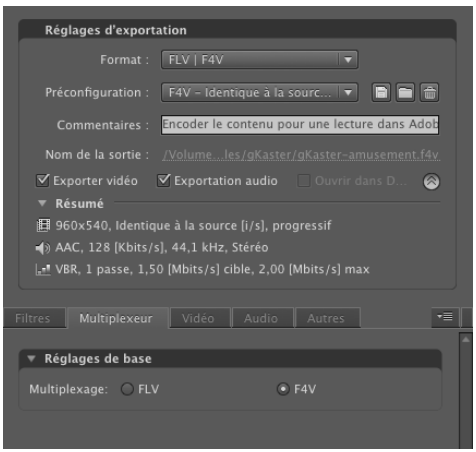
3. Dans la fenêtre de l'encodeur Adobe, cliquez directement sur le lien jaune de la colonne Prédéfinir pour accéder aux réglages personnalisés (voir Figure 7.2).

Figure 7.2
Fenêtre Adobe Media Encoder.



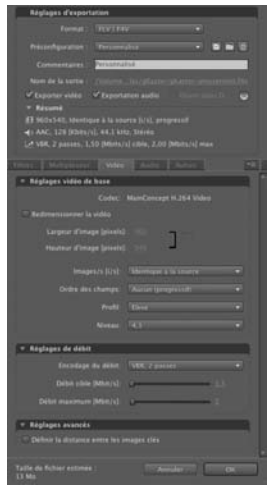
4. La fenêtre de réglages s'ouvre. Dans la partie droite de la fenêtre, activez l'onglet Multiplexeur pour définir le type d'encodage Flash (voir Figure 7.3).

Figure 7.3
Fenêtre Réglages d'exportation, onglet Multiplexeur.



5. Sélectionnez l'option F4V si celle-ci n'est pas déjà active.
6. Puis, cliquez sur l'onglet Vidéo pour définir les réglages relatifs au format F4V (voir Figure 7.4).

Figure 7.4
Onglet Vidéo.



Dans la catégorie Vidéo, nous découvrons des options légèrement différentes de celles affichées pour le format FLV.

Onglet Vidéo

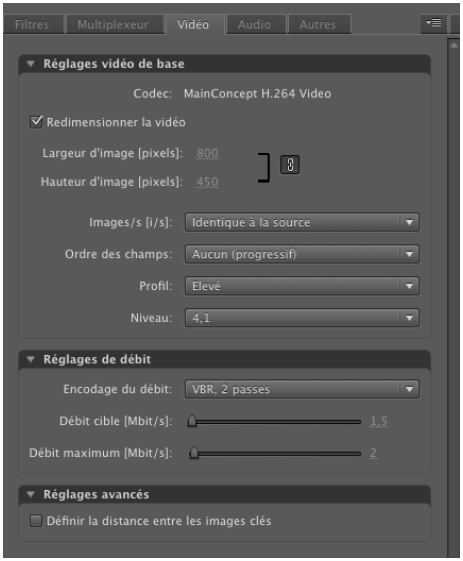
Le document Flash mesure 800 pixels de large. La vidéo gKaster en fait 960. Nous allons donc commencer par redimensionner cette vidéo pour la contenir dans notre document.

1. Cochez d'abord l'option de redimensionnement homothétique située à droite des champs Largeur et Hauteur.
2. Puis, inscrivez la valeur 800, dans le champ relatif à la largeur. La valeur Hauteur s'adapte automatiquement (voir Figure 7.5).

Dans la partie Réglages vidéo de base, vous trouverez de nouveaux paramètres, spécifiques au format F4V.

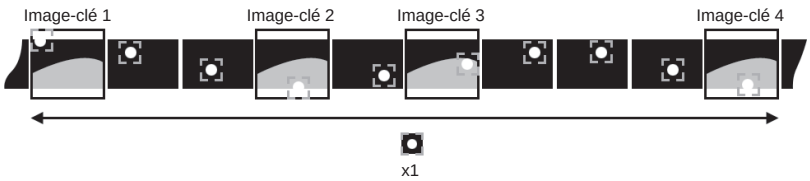
Le format F4V utilise le codec H-264 (ou MPEG-4 Partie 10/AVC). Ce codec est développé en commun par le VCEG (*Video Coding Expert Group* ou groupe d'experts en encodage vidéo), l'ITU-T (*International Telecommunications Union, Telecommunications Standardization Sector* ou le département standardisation de l'union internationale des télécommunications) et le MPEG (*Movie Picture Expert Group* ou le groupe d'experts en images animées). Il se présente comme une amélioration du codec MPEG-2 et MPEG-4 et repose sur le principe suivant.

Figure 7.5
Réglages vidéo de base (F4V).



Lorsqu'une compression classique n'utilise qu'une image-clé de référence pour définir les zones qui changent d'une image sur l'autre (codage différentiel), le H-264 peut utiliser jusqu'à quatre images de référence. Cette logique lui permet de détecter le mouvement global, sur la durée, de groupes de pixels entiers, et de ne coder que l'emplacement de ces groupes et non la teneur colorimétrique de chaque pixel qui compose chaque groupe. Ainsi, quand une scène se répète à l'intérieur d'une vidéo, ou si plusieurs événements graphiques identiques sont reproduits ou en mouvement sur différentes images-clés, ces événements ne sont codés qu'une fois (voir Figure 7.6). Cela permet de compacter largement les vidéos longues aux effets répétitifs, ou dont la nature des images reprend toujours le même type d'informations, qu'elles soient mobiles ou non (un sujet frontal sur un fond blanc, même en mouvement, sera peu gourmand en poids, par exemple). Une compression H-264 appliquée à une vidéo perd ainsi environ 80 % de son poids lorsqu'une vidéo compressée en MPEG-4 Partie 2 n'en perd que 50 %, et sans différence de qualité entre les deux modes de compression.

Figure 7.6
Représentation du principe de l'algorithme H-264.



Les blocs, qui fondent la base de l'algorithme H-264, sont constitués de 16 pixels (4×4). Outre l'avantage observé sur le poids du fichier, cette technique permet également de faciliter et stabiliser la lecture du flux vidéo surtout lorsqu'elle est diffusée en ligne. Car chaque paquet peut ainsi être lu indépendamment des autres et contribue à remplir automatiquement les trous résiduels éventuellement obtenus en cas de rupture de flux.

Ces blocs sont utilisés pour définir les images-clés de référence. Si une portion de l'image se *répète* ailleurs, l'algorithme identifie les nombreuses occurrences *identiques* de la zone

de base et la reproduit virtuellement, même dans une image-clé. Cela allège donc aussi le poids des images-clés.

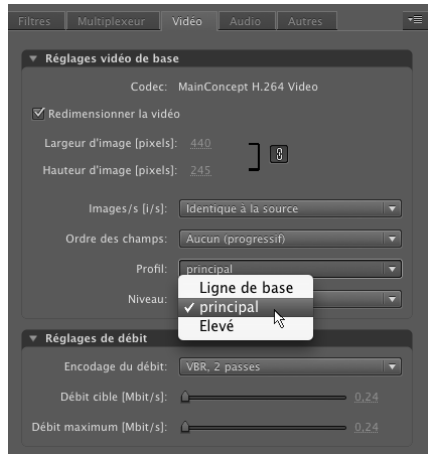
Le H-264 offre enfin un éventail de compression suffisamment large pour être employé pour des flux de haute résolution, avec une fréquence d'image élevée (diffusion HD), aussi bien que pour des flux très réduits et de faible définition (appareils mobiles).

De nombreuses options de compression sont possibles pour le H-264. Deux options essentielles sont disponibles dans l'encodeur Adobe : le profil et le niveau.

Profil

Le codec H-264 possède initialement sept profils de compression permettant de coder les images de faible résolution aux images à haute résolution. Les profils disponibles sont de qualité croissante et à sélectionner en fonction de la dimension de la vidéo. Parmi ces sept profils propres au H-264, Adobe Media Encoder en propose trois : ligne de base, principal et élevé (voir Figure 7.7).

Figure 7.7
Sélection d'un profil.

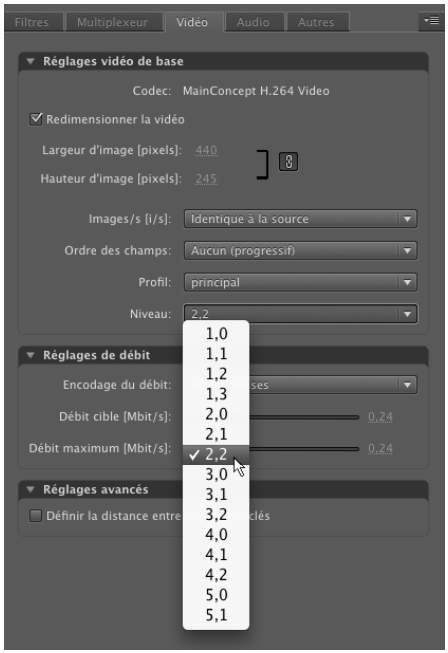


- **Ligne de base.** Ce profil est adapté aux contenus vidéos à destination d'appareils à faibles ressources (appareils nomades, mobiles, visio-conférence, caméras réseau). Le temps de latence de ce profil (temps requis pour compresser et décompresser le signal) est faible. Ce profil est donc tout particulièrement adapté pour l'encodage en direct de mouvements rapides (zoom, inclinaisons, vues panoramiques).
- **Principal.** Ce profil est adapté aux contenus à destination du Web, de la diffusion en vrai et faux streaming de contenus déjà enregistrés. Ce profil possède des capacités de robustesse à la perte de données qui en fait l'option idéale pour la vidéo diffusée en ligne.
- **Élevé.** Ce profil est utilisé pour la diffusion et le stockage sur disque, l'encodage Blue-Ray, HD-DVD et la télévision haute définition française. Il convient également à la diffusion de contenu en ligne de très haute qualité, pour lesquels on prévoit un chargement de la vidéo avant d'en activer la lecture, pour la V.O.D. par exemple.

Niveau

Selon les dimensions de la vidéo, nous déterminons un niveau (*Level*) sur une échelle de 1 à 5.1. Plus l'image à encoder est grande, plus le niveau doit être élevé (voir Figure 7.8).

Figure 7.8
Sélection d'un niveau



Dans le Tableau 7.1, la valeur de niveau à sélectionner est définie en fonction de la résolution de l'image et sa cadence. Pour notre exemple, conservez les valeurs par défaut.

Tableau 7.1 : Niveaux en H-264 (extrait de la page http://fr.wikipedia.org/wiki/H.264#Levels)

Niveau	Nombre maximum de macro-blocs lus par seconde	Débit maximum en bits pour les profils Ligne de base et Principal	Débit maximum en bits pour le profil Élevé	Exemples de définition et d'images par seconde par niveau
1	1485	64 Kbit/s	80 Kbit/s	128 × 96/30.9 176 × 144/15.0
1b	1485	128 Kbit/s	160 Kbit/s	128 × 96/30.9 176 × 144/15.0
1.1	3000	192 Kbit/s	240 Kbit/s	176 × 144/30.3 176 × 240/10.0
1.2	6000	384 Kbit/s	480 Kbit/s	176 × 144/60.6 320 × 240/20.0 352 × 288/15.2
1.3	11880	768 Kbit/s	960 Kbit/s	352 × 288/30.0
2	11880	2 Mbit/s	2.5 Mbit/s	352 × 288/30.0

Tableau 7.1 : Niveaux en H-264 (extrait de la page http://fr.wikipedia.org/wiki/H.264#Levels) *(suite)*

Niveau	Nombre maximum de macro-blocs lus par seconde	Débit maximum en bits pour les profils Ligne de base et Principal	Débit maximum en bits pour le profil Élevé	Exemples de définition et d'images par seconde par niveau
2.1	19800	4 Mbit/s	5 Mbit/s	352 × 480/30.0 352 × 576/25.0
2.2	20250	4 Mbit/s	5 Mbit/s	720 × 480/15.0 352 × 576/25.6
3	40500	10 Mbit/s	12.5 Mbit/s	720 × 480/30.0 720 × 576/25.0
3.1	108000	14 Mbit/s	17.5 Mbit/s	1 280 × 720/30.0 720 × 576/66.7
3.2	216000	20 Mbit/s	25 Mbit/s	1 280 × 720/60.0
4	245760	20 Mbit/s	25 Mbit/s	1 920 × 1 080/30.1 2 048 × 1 024/30.0
4.1	245760	50 Mbit/s	62.5 Mbit/s	1 920 × 1 080/30.1 2 048 × 1 024/30.0
4.2	522240	50 Mbit/s	62.5 Mbit/s	1 920 × 1 080/64.0 2 048 × 1 088/60.0
5	589824	135 Mbit/s	168.75 Mbit/s	1 920 × 1 080/72.3 2 560 × 1 920/30.7
5.1	983040	240 Mbit/s	300 Mbit/s	1 920 × 1 080/120.5 4 096 × 2 048/30.0

Onglet Audio

Le format F4V offre une compression audio de bien meilleure qualité que le FLV. Le codec AAC, supérieur à la qualité du MP3, propose une profondeur de son plus étendue, proche de celle d'un CD audio, mais naturellement, de poids légèrement plus élevé. C'est aussi le codec employé par Apple et RealNetWorks pour la diffusion de leurs contenus audios. Pour contrôler la compression audio, nous considérons véritablement deux paramètres : la fréquence et le débit (voir Figure 7.9).

Fréquence. La fréquence indique le nombre de fois qu'une ponction d'échantillons sonores est effectuée par seconde (à ne pas confondre avec la longueur d'onde qui représente la sonorité elle-même). Plus la fréquence est élevée, plus l'on peut distinguer les variations du son sur une durée donnée. Mais plus cela augmente nécessairement le poids du fichier.

Débit. Le débit consiste à synthétiser les informations en les arrondissant aux valeurs numériques les plus proches. Plus la compression du débit est forte, plus la représentation du son est grossière et perd en nuances, même avec une fréquence élevée.

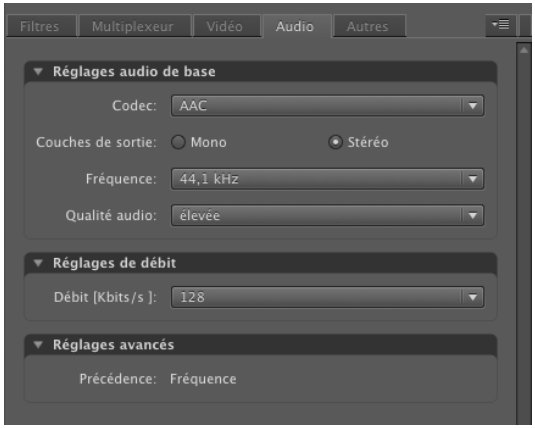


Figure 7.9
Réglages Audio (F4V).

Les paramètres de réglage affichés sont ceux définis par défaut pour le préréglage à partir duquel nous effectuons cette compression. Ces préréglages sont ceux affichés dans le menu Préconfiguration, au sommet de la fenêtre d'encodage, ou bien dans la liste Prédéfinir, de la file d'attente de rendu. Selon le type de préréglage choisi, vous n'observerez pas, par conséquent, les mêmes taux de compression.

1. Conservez pour cet exemple les valeurs par défaut.
2. Puis, refermez la boîte de dialogue des réglages de compression en cliquant sur OK.
3. Lancez le rendu en activant le bouton Démarrer la file d'attente.
4. Puis, quittez l'application.

Seuls des tests de rendu effectués en réel vous donneront une idée du résultat obtenu. N'hésitez pas à dupliquer le réglage dans la file d'attente de rendu et à y appliquer différents paramètres de compression audio pour ne retenir que celui qui vous convient le mieux.

À retenir

- Le format F4V offre une grande souplesse de compression qui permet d'adapter la vidéo à tout type de support, d'une projection HD à une image réduite pour appareil mobile, sans perte de qualité.
- Le format F4V ne permet pas la gestion de la transarence (couche alpha).
- L'encodage audio, en F4V, est supérieur en qualité au codage MP3 du format FLV.
- Si nous intégrons la vidéo avec un composant `FLVPlayback`, le format F4V requiert un lecteur Flash récent.

Encodage F4V avec Quick Time

Cela fait déjà longtemps qu'il est possible d'exporter avec une compression H-264 depuis Quick Time. Le format vidéo F4V de Flash qui repose sur le codec H-264, peut donc aussi être généré depuis un fichier Quick Time et être intégré à un composant Flash `FLVPlay-Back`, il sera normalement interprété. Pour éviter les messages possibles d'avertissement et de sécurité de Flash, lors de l'intégration de la vidéo, songez à substituer l'extension

obtenue avec Quick Time par `.f4v`. Mais, même sans en modifier l'extension, le fichier pourra néanmoins être lu comme un fichier F4V.

Exemples > *gKaster* > *gKaster-amusement.mov*

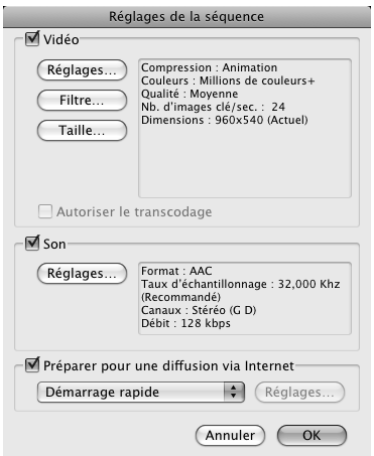
Dans cette section, nous détaillons le moyen d'encoder une vidéo, à partir de Quick Time, au format F4V, pour Flash.

1. Pour exporter la vidéo au format H-264 avec Quick Time, dans Quick Time, faites Fichier > Exporter.
2. Dans la boîte de dialogue d'enregistrement, cliquez sur le bouton options (voir Figure 7.10).

Figure 7.10
Fenêtre d'exportation Quick Time.

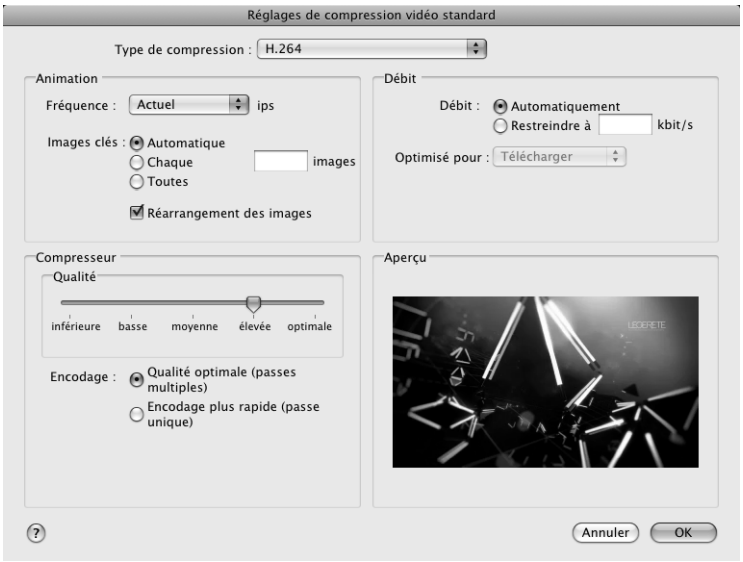
3. Dans la boîte de réglages, cliquez sur Réglages de la catégorie Vidéo (voir Figure 7.11).

Figure 7.11
Réglages de la séquence.



4. Puis, dans la fenêtre d'encodage, dans le menu Type de compression, sélectionnez H-264.
5. Dans les options de réglages pour le type H-264, choisissez Passes multiples et une qualité élevée, ainsi qu'un nombre d'image-clés automatique (voir Figure 7.12). Puis, validez.

Figure 7.12
Compression H-264.



Puisque le document Flash dont nous disposons mesure 800 pixels de large, nous pouvons également redimensionner ici la vidéo en activant l'option taille :

1. Cliquez sur le bouton Taille.
2. Dans la nouvelle boîte de dialogue, spécifiez une taille personnalisée. Puis, dans les champs situés à droite, inscrivez manuellement les valeurs 800 et 450.

Les options de proportions permettent de rogner et déformer ou non la vidéo afin qu'elle s'adapte aux nouvelles dimensions. L'option Letterbox ajoute du noir de part et d'autre si les proportions ne sont pas préservées. Dans notre cas, elles le sont. Cette option n'aura donc aucun effet. Vous pouvez donc valider (voir Figure 7.13).

Figure 7.13
Compression H-264.

La boîte de dialogue d'options affiche maintenant les informations en rapport avec la compression H-264 (voir Figure 7.14).

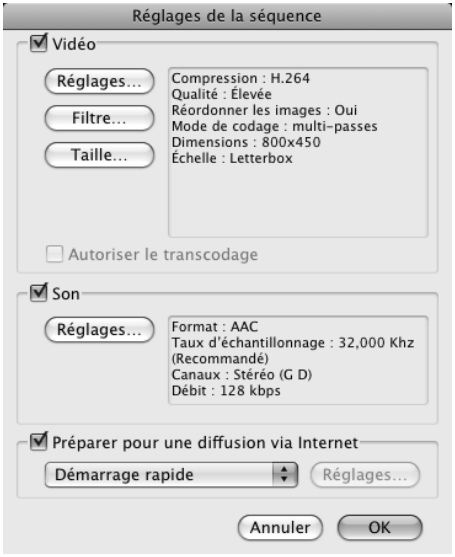


Figure 7.14
Compression H-264.

Attention, Quick Time conserve par défaut les précédents réglages lors de chaque nouvelle compression. Lorsque vous encoderez un nouveau document, pensez à initialiser le redimensionnement pour éviter d'affecter votre nouveau fichier.

Validez également la fenêtre d'options. Puis, renommez le fichier "gKaster-amusement-h264.mov" et confirmez l'enregistrement. L'encodage se termine. La vidéo peut être appelée directement depuis Flash, *via* un composant `FLVPlayBack` (voir Figure 7.15).

Figure 7.15
Lecture d'une vidéo Quick Time H-264 dans un document Flash.

Info

Les formats pris en charge par le composant `FLVPlayBack` (publié dans un document configuré pour ActionScript 3) sont : FLV, F4V, MP4, M4A, MOV, MP4V, 3GP et 3G2.

À retenir

- Quick Time permet d'exporter facilement une vidéo avec un encodage H-264 directement lisible par le composant `FLVPlayback` de Flash CS4.

Créer un lecteur vidéo personnalisé

Dans Flash, en utilisant le composant vidéo en vue d'intégrer une vidéo FLV ou F4V, une console de lecture est disponible par défaut. Elle permet de contrôler la vidéo sans programmation spécifique. Différents types de consoles sont disponibles à travers le paramètre Skin accessible dans l'Inspecteur de composants de chaque vidéo active.

Si vous utilisez une Skin prédéfinie, n'oubliez pas que Flash génère un fichier SWF pour chacune d'elle et qu'il ne faudra pas manquer de placer ce fichier sur le serveur d'hébergement, en même temps que votre document SWF de base et votre vidéo, sans quoi la console n'apparaîtrait pas.

Info

Lorsque vous utilisez une Skin prédéfinie, Flash importe en réalité un fichier SWF qu'il génère à la volée, selon le type de Skin sélectionné dans la fenêtre Inspecteur de composants. Ces boutons sont standardisés. Mais vous pouvez modifier la forme intrinsèque des boutons déjà encapsulés dans ces Skins prédéfinies. Pour ce faire, dans le moteur de recherche de votre système, saisissez le nom de la Skin générée par Flash à la publication (par exemple `SkinOverAll`). Puis, repérez l'emplacement du fichier FLA, natif, ayant permis à Flash de générer ce fichier. Il suffit d'ouvrir ce FLA dans Flash, de le modifier et de publier un nouveau fichier en lieu et place du précédent. La console accessible depuis votre composant est instantanément mise à jour.

Pour personnaliser la forme des boutons, nous pouvons utiliser de simples symboles créés manuellement ou bien recourir à des composants séparés qui contrôlent la lecture de la vidéo. Ces composants sont également disponibles depuis la fenêtre des composants. L'avantage de ces éléments est qu'ils disposent déjà d'une structure animée (effet "rollOver" intégré) et d'actions de contrôles préprogrammés.

Dans cette section, nous présentons uniquement ces boutons préprogrammés. Nous revenons sur les actions gérées manuellement en ActionScript, au Chapitre 8.

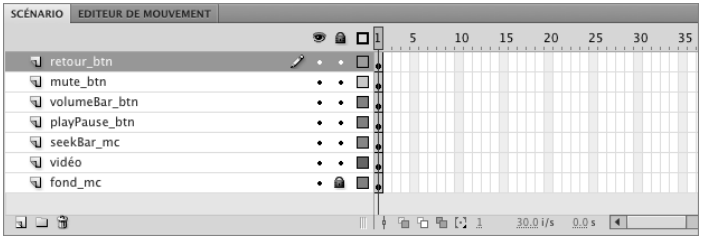
Exemples > ch7_videoHD_3.fla

Dans le document "ch7_videoHD_3.fla", sur la scène, un composant vidéo apparaît au-dessus des boutons de contrôle personnalisés (voir Figure 7.16).

Dans le scénario, au-dessus du calque `fond_mc`, chaque symbole ou composant est réparti vers un calque distinct (voir Figure 7.17). Ils possèdent des noms d'occurrence, mais cela est facultatif puisqu'aucune action n'est associée à ces objets sinon celles déjà encapsulées dans les composants boutons eux-mêmes.

Figure 7.16
Composant `FLVPlayBack` avec boutons personnalisés.

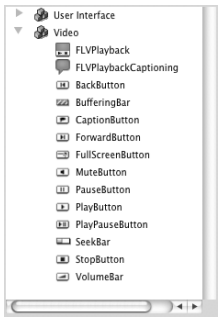
Figure 7.17
Fenêtre de scénario.



Pour insérer des boutons prédéfinis, prêts à utiliser, vous devez placer des éléments de la fenêtre Composants, sur la même scène que celle où figure le composant vidéo `FLVPlayBack`. Flash, au moment de la publication, va automatiquement établir une liaison entre le composant vidéo et les boutons qui s'y réfèrent.

Pour placer des boutons préprogrammés sur la scène, affichez la fenêtre des composants *via* Fenêtre > Composants (voir Figure 7.18). Puis, dans la catégorie Vidéo, sélectionnez le bouton de votre choix. Puis, glisser-déposez les boutons individuellement sur la scène où se trouve le composant `FLVPlayBack`.

Figure 7.18
Fenêtre Composants.



Double-cliquez ensuite, dans la scène, sur chacun d'entre eux jusqu'à atteindre le graphisme modifiable pour éventuellement les personnaliser. Enregistrez et publiez. Les boutons insérés contrôlent la vidéo (voir Figure 7.19).

Figure 7.19
Aperçu du document après publication.

Vidéo

Définition des composants associés à `FLVPlayBack`

Voici les commandes disponibles sous la forme de boutons prédéfinis, depuis la fenêtre Composants et leur définition :

- `BackButton`. Bouton qui permet théoriquement de rembobiner le flux vidéo. En réalité, il renvoie à la première image du flux vidéo (en progressif).
- `BufferingBar`. Barre de progression de mise en cache de la vidéo. Elle avance à mesure que la vidéo est chargée dans le cache du navigateur et signale à l'utilisateur la possibilité de naviguer à l'intérieur du flux déjà mis en cache.
- `CaptionButton`. Utilisé avec le composant `FLVPlayBack Captionning` pour réaliser des sous-titrages (voir Chapitre 8).
- `ForwardButton`. Bouton d'accélération de la lecture de la vidéo. Attention, en flux progressif, seule la partie déjà chargée peut être atteinte par l'utilisateur.
- `FullScreenButton`. Active l'Affichage en mode plein écran du contenu vidéo (voir aussi le Chapitre 15 pour la gestion du mode d'affichage en plein écran).
- `MuteButton`. Permet de stopper l'audio et de le réactiver, sur le même bouton.
- `PauseButton`. Arrête la lecture du flux vidéo, sans interrompre la connexion au serveur. La mise en cache se poursuit de manière transparente.
- `PlayButton`. Lit ou reprend la lecture d'un flux vidéo.
- `PlayPauseButton`. Propose sur le même bouton, une fonctionnalité de lecture de la vidéo lorsque celle-ci est en pause, et de mise en pause lorsqu'elle est en lecture.

- `SeekBar`. Barre de progression de la lecture de la vidéo. À la différence du composant `BufferingBar`, la barre affiche la position courante de la vidéo dans la scène. L'utilisateur peut déplacer la tête de lecture pour atteindre un autre extrait de la bande vidéo, dans la mesure, toujours, où le contenu appelé en flux progressif est déjà chargé par le navigateur.
- `StopButton`. Arrête la lecture vidéo et interrompt aussi son chargement. Cela libère le canal et les ressources graphiques sollicitées pendant la lecture de la vidéo.
- `VolumeBar`. Modifie le volume sonore de l'audio inclus dans la vidéo.

Info

Il se peut que vous cherchiez à intégrer une vidéo dans un fichier SWF, lui-même importé dans un autre document SWF. Le fait de supprimer le SWF chargé dans la liste d'affichage (avec `removeChild`) ne permet pas, dans ce cas, d'interrompre la lecture de la vidéo. Pour ce faire, il convient d'associer, à la vidéo, des contrôles supplémentaires en ActionScript. Nous détaillons ces contrôles au chapitre suivant.

À retenir

- Le composant `FLVPlayBack` met à disposition des consoles de lecture de la vidéo, personnalisables si l'on édite le FLA utilisé pour les générer, à partir du document source disponible dans le dossier de l'application Flash.
- Des boutons composants sont utilisables individuellement et peuvent être personnalisées directement depuis l'interface auteur, dans le scnario. Il ne requièrent en outre aucune programmation spécifique pour fonctionner.

Créer un lecteur vidéo H-264 pour Flash 6 et plus

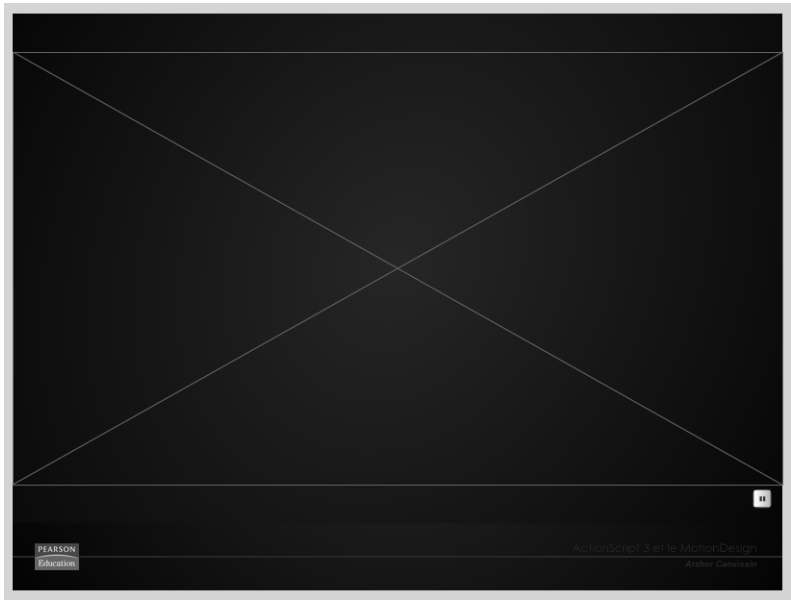
Si l'utilisation du composant offre une plus grande souplesse de manipulation pour des formats standards comme le FLV et le F4V, il ne permet pas de lire des fichiers vidéo codés en H-264 avec des versions de documents antérieures à la version 10 (CS4), car le codec H-264 n'est implémenté dans le composant `FLVPlayBack` que depuis cette version.

Pour permettre de lire un flux HD (H-264) dans des versions de Flash antérieures à Flash 10, nous créons un lecteur vidéo manuellement, en ActionScript, à l'aide de la classe `NetStream`. Cette classe étant apparue avec Flash 6, nous pouvons programmer l'affichage de la vidéo en haute définition à partir de Flash 6 et pour les versions ultérieures.

Exemples > ch7_videoHD_4.fla

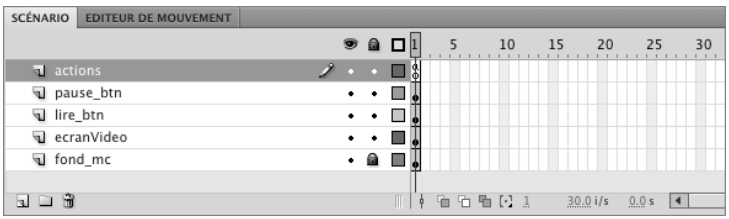
Dans le document "ch7_videoHD_4.fla", sur la scène, apparaît une vidéo vide et transparente. C'est un objet vidéo importé de la bibliothèque et utilisé avec la classe `NetStream` (voir Figure 7.20). À droite et en bas, deux boutons `lire_btn` et `pause_btn` sont superposés.

Figure 7.20
Aperçu de la scène principale.



Dans la fenêtre de scénario, au-dessus du calque `fond_mc`, un calque affiche l'objet vidéo. Deux calques distribuent respectivement les deux boutons `lire_btn` et `pause_btn`. Un autre calque affiche les actions (voir Figure 7.21).

Figure 7.21
Aperçu du scénario de la scène principale.



Les propriétés du document affichent enfin un format d'exportation Flash 6 et ActionScript en version 2 (voir Figure 7.22).

Les objets vidéo sont nativement disponibles dans tous les documents Flash, depuis la bibliothèque. L'objet qui figure dans ce document est donc extrait de la bibliothèque. Pour extraire un objet vidéo de la bibliothèque, procédez comme suit :

1. Affichez la bibliothèque (Fenêtre > Bibliothèque ou Cmd+L sur Mac ou Ctrl+L sur Windows).
2. Pour insérer un objet vidéo, supprimez éventuellement l'objet déjà en place sur la scène et dans la bibliothèque.
3. Dans le menu contextuel de la bibliothèque, sélectionnez l'option Nouvelle Vidéo (voir Figure 7.23).
4. Dans la boîte de dialogue, attribuez un nom d'objet, par exemple `ecranVideo` (voir Figure 7.24). Puis validez.

Figure 7.22
Aperçu des propriétés de la scène principale.

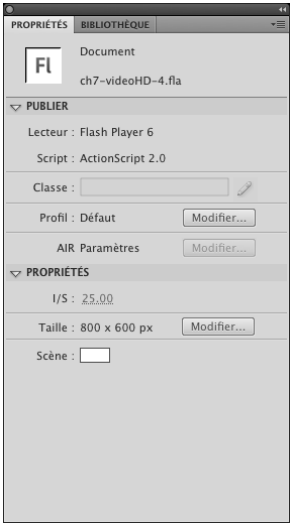


Figure 7.23
Création d'un nouvel objet Vidéo.

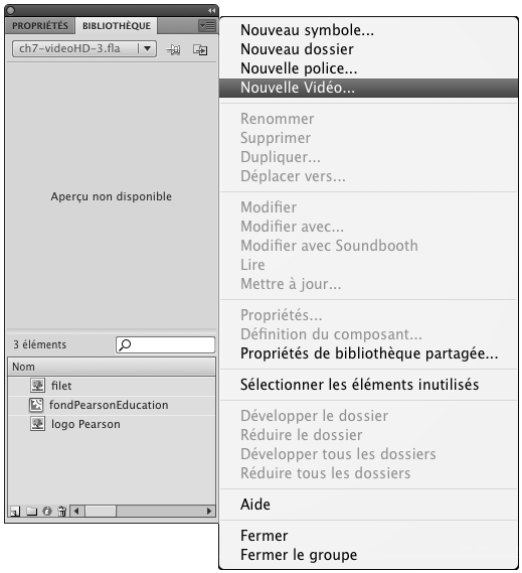
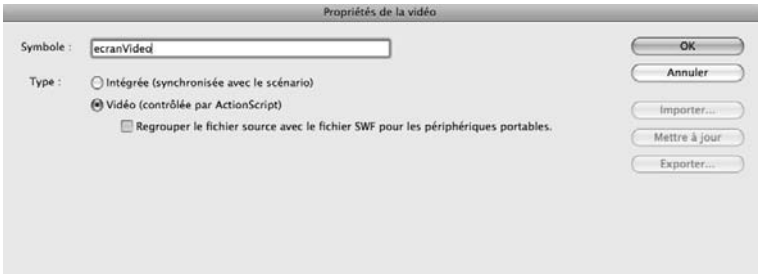


Figure 7.24
Propriétés de la vidéo.

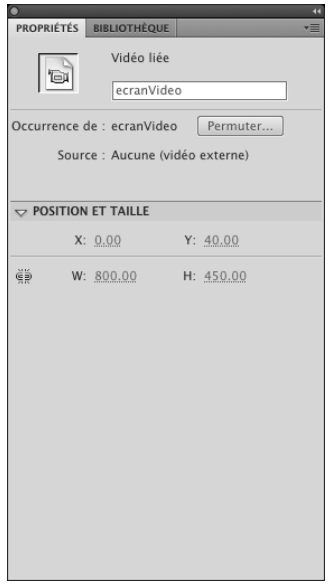


5. Le nouvel objet apparaît dans la bibliothèque (voir Figure 7.25).

Figure 7.25
Fenêtre Bibliothèque.

6. Glisser-déposez cet objet sur la scène et, depuis l'Inspecteur de propriétés, donnez-lui un nom d'occurrence, par exemple : `ecranVideo`. Puis, redimensionnez-le à la taille du flux vidéo à importer, par exemple 800 × 450 pixels (voir Figure 7.26).

Figure 7.26
Propriétés de l'occurrence de l'objet vidéo placé sur la scène.



Dans la fenêtre Actions, nous pouvons lire le code suivant, en ActionScript 2 :

```
// Lecteur vidéo Flash 6
connexion = new NetConnection();
connexion.connect(null);
chargementContinu = new NetStream(connexion);
ecranVideo.attachVideo(chargementContinu);
chargementContinu.play("gKaster/gKaster-amusement.f4v");

// gestion du bouton Play/Pause

lire_btn._visible=false;
pause_btn._visible=true;

lire_btn.onPress=function  () {
   chargementContinu.pause();
   lire_btn._visible=false;
   pause_btn._visible=true;
}
pause_btn.onPress=function  () {
   chargementContinu.pause();
   lire_btn._visible=true;
   pause_btn._visible=false;
}
```

Ce code est structuré en deux parties. La première gère l'affichage de la vidéo. La seconde organise les boutons `lire_btn` et `pause_btn` qui contrôlent la vidéo.

Dans la première partie, nous spécifions d'abord, sans typage spécifique (AS2 autorise), une variable intitulée connexion qui désigne l'activation d'un flux Internet distant :

```
connexion = new NetConnection();
```

Plus loin, nous initialisons cette connexion :

```
connexion.connect(null);
```

La valeur `null`, renseignée en paramètre, spécifie que l'on se connecte en relatif sur le même emplacement que la source SWF. Sinon il serait également possible d'user de http:// ou de rtmp:// pour désigner une référence absolue.

Puis, nous créons une deuxième variable qui active le transfert d'un flux à chargement progressif (faux *streaming*). Nous précisons, en paramètre de la méthode `NetStream` que ce flux concerne la connexion préalablement définie :

```
chargementContinu = new NetStream(connexion);
```

Nous attachons ensuite ce flux progressif à l'occurrence vidéo qui figure sur la scène (`ecranVideo`) :

```
ecranVideo.attachVideo(chargementContinu);
```

Enfin, nous spécifions que ce flux doit lire le fichier localisé dans le chemin défini entre les guillemets :

```
chargementContinu.play("gKaster/gKaster-amusement.f4v");
```

Même si le format F4V, comme nous l'avons précisé, n'est pas reconnu dans les anciennes versions de Flash, ce qui est appelé ici est bien un fichier encodé en H-264. Flash, en réalité,

fait abstraction de l'extension lors de la lecture de vidéos. C'est pourquoi nous pouvons appeler directement le fichier nommé F4V. Le composant `FLVPlayBack` des anciennes versions ne sait pas interpréter le F4V, mais la classe `NetStream`, elle, le fait.

Dans la seconde partie du code, deux actions contrôlent chaque bouton individuellement. Le premier, `lire_btn`, se masque lui-même et affiche le bouton `pause_btn`. Simultanément, il reprend la lecture là où elle était arrêtée grâce à la méthode `pause()` :

```
lire_btn.onPress=function  () {
   chargementContinu.pause();
   lire_btn._visible=false;
   pause_btn._visible=true;
}
```

De même, le bouton `pause_btn` se fait disparaître et réaffiche le bouton `lire_btn` tout en interrompant la lecture en cours :

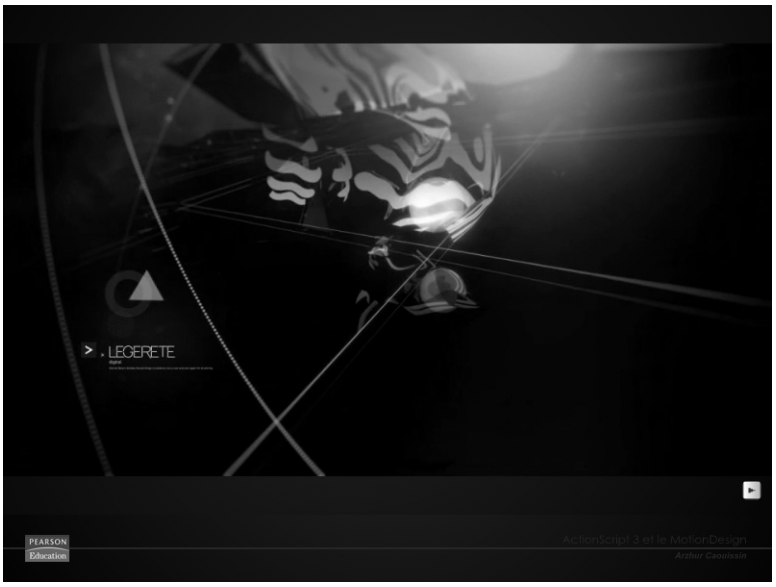
```
pause_btn.onPress=function  () {
   chargementContinu.pause();
   lire_btn._visible=true;
   pause_btn._visible=false;
}
```

À l'initialisation, puisque la vidéo est lue dès le chargement, juste avant les actions attachées aux deux boutons, nous spécifions que le bouton `pause_btn` est visible et que `lire_btn` est masqué :

```
lire_btn._visible=false;
pause_btn._visible=true;
```

En publiant le document, la vidéo F4V joue instantanément en haute définition dans un document Flash 6 codé en ActionScript 2. Les boutons lire et pause contrôlent la vidéo (voir Figure 7.27).

Figure 7.27
Lecture d'une vidéo Quick Time H-264 dans un document Flash.



À retenir

- Il est possible de publier une vidéo en haute définition pour un lecteur Flash 6 ou de version ultérieure. Pour cela, nous utilisons la classe `NetStream`.
- Pour contrôler la lecture du flux vidéo HD avec `NetStream`, nous codons et exportons en ActionScript 2.

Agrandir une vidéo sans perte

La difficulté de la vidéo pour le Web réside dans le bon compromis entre une compression avec une contenu accessible et une qualité qui, elle, valorise le contenu au détriment de l'accessibilité. Avec des vidéos traitées en haute définition, le compromis devient presque impossible à résoudre dans des contextes où l'image est très riche (présence de bruit continu, images toutes différentes, audio avec une belle amplitude, animation de durée conséquente, etc). Pour permettre une compression forte tout en conservant l'illusion d'une image de qualité, il existe une astuce. Comme dans l'édition papier, et dans la presse quotidienne en particulier, pour compenser la perte engendrée par une image allégée, nous lui appliquons une trame.

Vous pouvez appliquer une trame dans Flash au moyen d'une image tramée et transparente placée dans un `MovieClip` auquel vous appliquez une propriété d'affichage de type Incrustation ou Produit. Mais, cela sollicite de manière importante la carte vidéo de l'utilisateur. Nous préférons alors intégrer cette trame directement dans le flux vidéo :

1. Dans un logiciel graphique, comme Photoshop, dessinez cette trame en reproduisant sur une image de dimensions identiques à celles de la vidéo, un motif (un filet, un damier, des points, des croix ou un effet trame de demi-teinte, par exemple) – voir Figure 7.28.

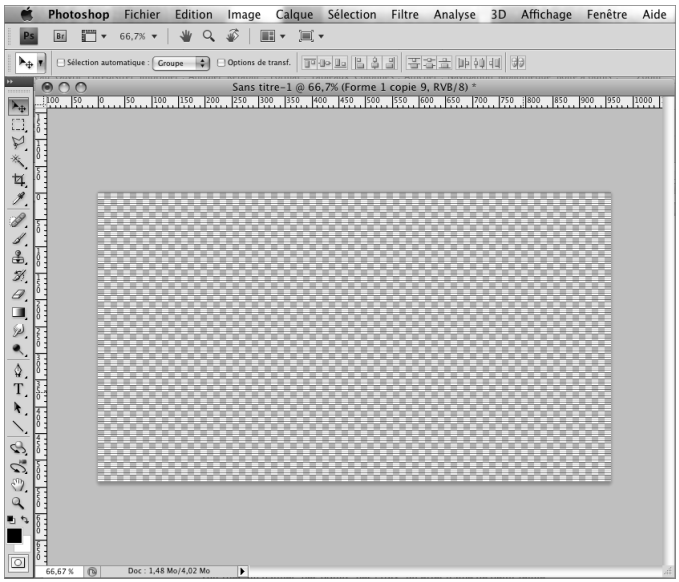
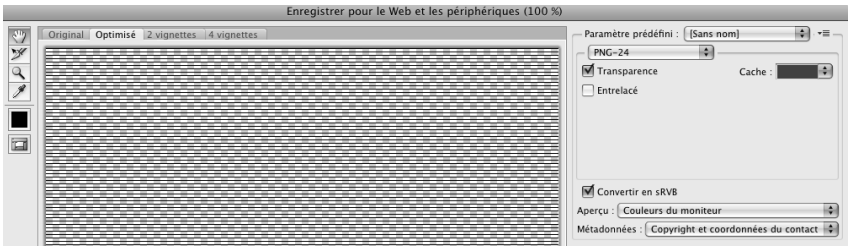


Figure 7.28
Création dans Photoshop d'une trame de type filaire.

2. Puis, exportez cette image en PNG-24, afin d'en préserver la transparence, en faisant Fichier > Exporter pour le Web et les périphériques.
3. Dans la boîte de dialogue d'enregistrement, à droite, sélectionnez l'option PNG-24 et spécifiez bien une transparence en cochant cette option (voir Figure 7.29).

Figure 7.29
Exporter au format PNG-24, avec la transparence.



4. Dans After Effects par exemple, importez l'image en faisant Ctrl+I (Windows) ou Cmd+I (Mac).
5. Puis, glissez l'image sur la vidéo déjà en place dans la scène (voir Figure 7.30). Au besoin, ajustez le paramètre Mode disponible sur le calque dans le scénario, en choisissant un mode de type Produit, Incrustation ou autre.

Figure 7.30
Appliquer la trame sur la vidéo, dans After Effects.

6. Puis, encodez la vidéo.

Pour le logiciel Motion, procédez de même.

1. Depuis la fenêtre Bibliothèque, glisser-déposez l'image réalisée dans Photoshop sur votre scène.
2. Puis, exportez la vidéo au format Quick Time.

À retenir

- Pour agrandir une vidéo, nous conservons une compression forte, mais compensons l'affichage par l'ajout d'un tramage.
- Il est préférable d'intégrer ce tramage dans le flux du signal vidéo plutôt que dans le document Flash afin d'optimiser les ressources graphiques de l'utilisateur.

Synthèse

Dans ce chapitre, vous avez appris à gérer l'affichage de flux vidéo en haute définition, codés en H-264, avec Adobe Media Encoder ou tout autre type de source et notamment avec les solutions Apple. Vous avez également appris à intégrer une vidéo HD pour les anciennes versions de Flash, jusqu'à la version 6. Vous avez appris à personnaliser les boutons de contrôle de lecture d'une vidéo et à contourner les contraintes de poids liées à une forte compression grâce à des astuces inspirées de l'univers de la presse. Vous êtes à mesure de réaliser désormais des sites contenant des vidéos de qualité professionnelle et haut de gamme, compatibles avec toutes les configurations d'utilisateur.