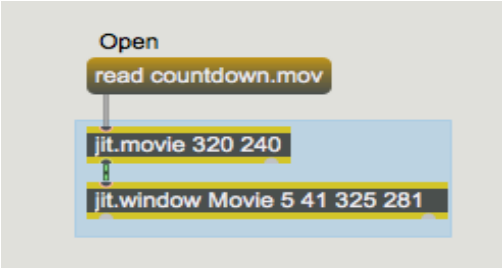
1-Lecture d'un film

Ce premier didacticiel présente l'une des tâches les plus simples et pourtant les plus utiles que vous pouvez effectuer avec Jitter: lire un film dans une fenêtre.

Il y a deux objets Jitter: dans ce patch: *jit.movie* et *jit.window*. L'objet *jit.window* ouvre automatiquement une fenêtre sur l'écran de votre ordinateur. L'objet *jit.movie* vous permet d'ouvrir un film existant et commence à le lire.

• Cliquez sur la boîte de *message* contenant le message **read countdown.mov**. L"objet *jit.movie* ouvre alors le fichier vidéo ***countdown.mov*** et commencer à le lire.



Le message **read** ouvre un fichier vidéo.

Par défaut, *jit.movie* commence à lire un film dès qu'il l'ouvre. (Vous pouvez aussi modifier ce comportement en envoyant à l'objet *jit.movie* un message **autostart 0** avant d'ouvrir le fichier, mais pour l'instant le comportement par défaut convient.) Remarquez, cependant, que même si nous avons dit que l'objet *jit.movie* lit le film, le film n'est pas affiché dans la fenêtre Movie. Voici pourquoi:

Chaque objet Jitter effectue une tâche particulière. Cette tâche peut être très simple ou plutôt compliquée. Ce que nous considérons comme "lire un film" est en fait décomposé par Jitter en deux tâches:

1. Lire chaque image de données de film dans la RAM à partir du fichier sur le disque dur

2. Récupérer les données qui sont dans la RAM et les afficher sous la forme de pixels colorés sur l'écran.

La première tâche est effectuée par l'objet *jit.movie* et la seconde par l'objet *jit.window*. Mais pour que l'objet *jit.window* sache quoi afficher, ces deux objets doivent communiquer.

Comment les objets Jitter communiquent

> *Concept important*: La chose la plus importante que les objets Jitter communiquent entre eux est un nom, faisant référence à une matrice - un endroit en mémoire où des données sont stockées. (Nous expliquerons la signification du mot "matrice" plus en détail dans le prochain chapitre du didacticiel.) Les objets Jitter émettent un message que seuls les autres objets Jitter comprennent. Ce message est le mot **jit_matrix** suivi d'un espace et du nom d'une matrice où les données sont stockées. Ce message est communiqué d'un objet Jitter à un autre par le biais d'un cordon de raccordement de la manière normale dans Max. (Mais, tout comme les cordons de patch des objets MSP ont un aspect différent des autres cordons de patch Max, les cordons de raccordement des sorties des objets Jitter qui envoient le message **jit_matrix** ont leur propre aspect.) L'objet Jitter récepteur reçoit le message dans

son entrée (le plus souvent l'entrée gauche), récupère les données de l'emplacement spécifié en mémoire, modifie les données d'une certaine manière, et envoie le nom des données modifiées par sa sortie gauche à tous les objets Jitter connectés. De cette façon, les tâches sont effectuées par chaque objet sans nécessairement savoir ce que font les autres objets, et chaque objet obtient les données dont il a besoin en consultant l'emplacement approprié en mémoire. La plupart des objets Jitter ne font rien jusqu'à ce qu'ils reçoivent un message **jit_matrix** d'un autre objet Jitter, leur indiquant de consulter cette matrice et de faire quelque chose avec les données qui s'y trouvent.

Dans de nombreux cas, un objet Jitter génére lui-même un nom unique pour sa matrice. Dans d'autres cas, il est possible (et même souhaitable) de dire à un objet quel nom utiliser pour une matrice. En nommant explicitement une matrice, nous pouvons faire en sorte que les objets utilisent le même espace mémoire. Vous en verrez des exemples dans les prochains chapitres du didacticiel.

Cause de l'action des objets Jitter

Pourquoi un objet Jitter envoie-t-il un message **jit_matrix** à un autre objet? La plupart des objets Jitter envoient un message **jit_matrix** lorsqu'ils reçoivent le message **outputmatrix** ou **bang**. (Ces deux messages ont le même effet dans la plupart des objets Jitter.) L'autre cas où un objet envoie un message **jit_matrix**, c'est lorsqu'il a lui-même reçu un tel message et qu'il a modifié les données d'une manière ou d'une autre. il envoie ensuite automatiquement un message **jit_matrix** pour informer les autres objets du nom de la matrice contenant les nouvelles données.

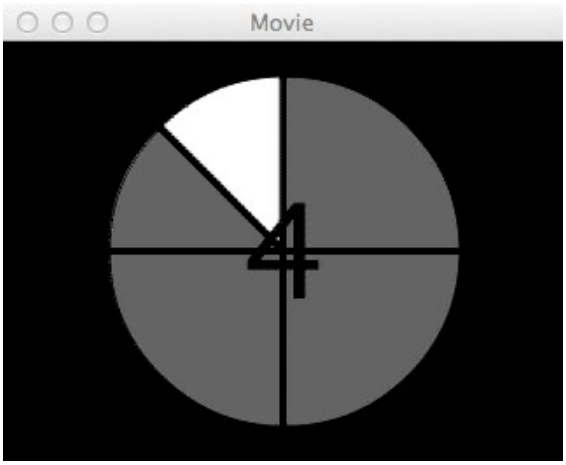
Pour reformuler le paragraphe précédent, lorsqu'un objet reçoit un message **jit_matrix**, il agit et envoie lui-même un message **jit_matrix**. Lorsqu'un objet reçoit **outputmatrix** ou **bang**, il envoie un message **jit_matrix** sans rien faire d'autre.

Ainsi, dans notre exemple de patch, l'objet *jit.movie* "lit" le film, stockant constamment l'image de la vidéo actuelle, mais l'objet *jit.window* n'affichera quelque chose que lorsqu'il recevra un message **jit_matrix** de l'objet *jit.movie*. Et cela ne se produira que lorsque *jit.movie* recevra le message **bang** (ou **outputmatrix**). À ce moment-là, *jit.window* affichera l'image vidéo en cours de lecture dans le film (c'est-à-dire l'image actuellement stockée par *jit.movie*).

Pour que *jit.window* mette à jour son affichage à la vitesse souhaitée pour montrer une vidéo en progression continue, nous devons envoyer le message **bang** à *jit.movie* à cette vitesse.

Le film est en lecture dans *jit.movie*,
mais nous devons lui envoyer un **bang** chaque fois que nous voulons afficher une image.

• Cliquez sur l'objet *toggle* marqué "Play" pour démarrer l'objet *metro*. Cela enverra des **bangs** au rythme de 25 fois par seconde (toutes les 40 millisecondes). Cela devrait être assez rapide pour afficher chaque image de cette vidéo. Tant que les messages **bang** continuent, vous verrez le film affiché dans la fenêtre Movie.



jit.window affiche le contenu d'une matrice : dans ce cas, une image de film

• Cliquez sur le *toggle* pour arrêter le *metro*. L'objet *jit.window* cesse de mettre à jour la fenêtre Movie, de sorte que vous ne verrez donc plus qu'une image fixe de la dernière image affichée. Le film est toujours en cours de lecture - et *jit.movie* met toujours à jour sa mémoire image par image -, mais *jit.window* est désormais invisible, car *jit.movie* n'envoie plus de messages.
• Vous pouvez vérifier que le film progresse toujours en cliquant sur l'objet *button* en dessous du *metro*. Cela fera en sorte que *jit.movie* envoie un message **jit_matrix** à *jit.window* par *jit.movie*, qui mettra à jour la fenêtre Movie avec l'image actuelle. Si vous faites cela plusieurs fois, vous verrez que le film a progressé entre deux clics de souris. (Le film est un compte à rebours de dix secondes et se déroule en boucle.).

En résumé *jit.movie* lit continuellement une image vidéo du film, image par image, à la vitesse normale du film. Lorsque *jit.movie* reçoit un **bang**, il communique l'emplacement de ces données (cette seule image vidéo) à *jit.window,* de sorte que l'image que *jit.movie* contient lorsqu'il reçoit un **bang**, est la donnée qui sera affichée par *jit.window*.

Arguments dans les objets

L'objet *jit.movie* de ce patch de tutoriel a deux arguments saisis: **320 240**. Ces nombres spécifient les dimensions horizontales et verticales (largeur et hauteur) que l'objet utilisera afin de conserver une seule image vidéo en mémoire. Il demandera suffisamment de RAM pour stocker une image avec ces dimensions. Ainsi, dans le cas le plus simple, il est logique de saisir les dimensions de la vidéo que vous souhaitez lire avec le message **read**. Dans ce cas (puisque nous avons nous-mêmes créé le film en question), nous savons que les dimensions du film ***countdown.mov*** sont 320x240.

Si nous saisissons des arguments de dimension plus petits que les dimensions du film que nous avons lu, *jit.movie* n'aura pas réclamé assez d'espace mémoire et sera obligé d'ignorer certains des pixels de chaque image du film. Inversement, si nous entrons des arguments de dimension plus grands que les dimensions du film que nous lisons, il n'y aura pas assez de pixels dans chaque image du film pour remplir tout l'espace mémoire qui a été alloué., donc *jit.movie* distribuera les données qu'il contient de manière égale et remplira sa mémoire supplémentaire avec des données en double.

L'objet *jit.window* possède cinq arguments saisis: **Movie 5 41 325 281**. Le premier argument est un nom qui sera donné à la matrice de données que *jit.window* affiche. Ce nom apparaîtra également dans la barre de titre de la fenêtre du film. Il peut s'agir de n'importe quel mot, ou de plus d'un mot si le nom complet est placé entre guillemets. Les deux arguments suivants indiquent les coordonnées d'écran x, y du coin supérieur gauche de la région d'affichage de la fenêtre de film et les deux derniers arguments fournissent les coordonnées x, y du coin inférieur droit de la région d'affichage. (Une autre façon de penser à ces quatre nombres est de les mémoriser en tant que coordonnées signifiant "gauche", "haut", "droite" et "bas".) Nous avons choisi ces nombres particuliers car ***a)*** ils décrivent une région d'affichage de 320x240 pixels, la même taille que la vidéo que nous avons l'intention d'afficher, et ***b)*** lorsque nous prenons en compte les dimensions des bords de la fenêtre, de la barre de titre et de la barre de menus imposées par le système d'exploitation, la fenêtre entière sera soigneusement placée dans le coin supérieur gauche de notre bureau utilisable. (Il est possible de faire disparaître les bords de la fenêtre et la barre de titre avec un message **border 0** à *jit.window*, mais les bords par défaut sont correctes pour le moment.)

Nous avons saisi la valeur **40** en tant qu'argument de *metro* pour qu'il envoie 25 messages **bang** par seconde. Le film a en fait une fréquence d'images d'exactement 24 images par seconde, donc ce *metro* déclenchera l'objet *jit.movie* assez fréquemment pour s'assurer que chaque image soit mise à la disposition de *jit.window* et que nous pourrons voir chaque image.

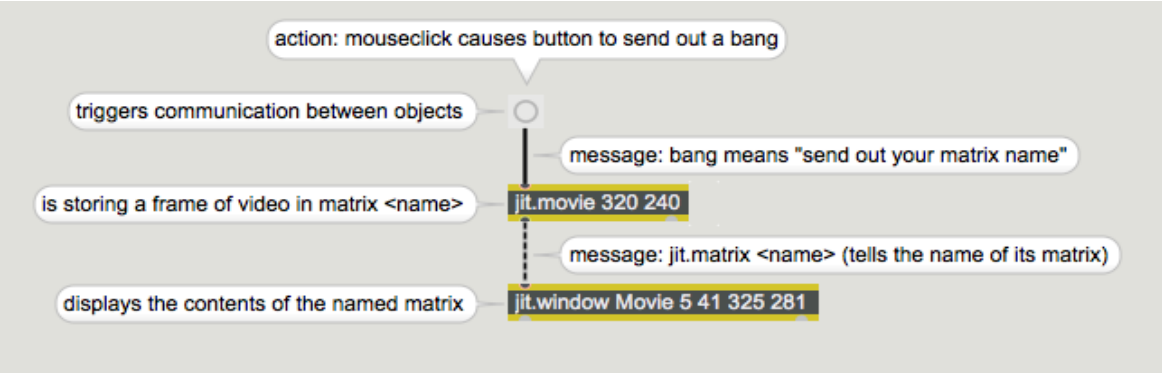
L'objet *jit.movie* comprend en réalité bien d'autres messages qu'un simple **bang** (bien trop nombreux pour être expliqués ici). Dans le coin supérieur droit de la fenêtre du Patcher, nous avons inclus un exemple de message supplémentaire, simplement pour démontrer que la progression du film peut être contrôlée dans *jit.movie* indépendamment de la vitesse à laquelle le *metro* lui envoie des messages **bang**. Le message **Time**, suivi d'un nombre, amène *jit.movie* à sauter immédiatement à un moment précis du film.

• Cliquez sur l'objet *button* intitulé "**Restart**". Cela envoie un message **Time 0** à *jit.movie,* le faisant sauter au début du film, puis envoie un message **1** au *toggle* pour démarrer le *metro* et commencer à afficher le film.

Pour lire un film, utilisez l'objet *jit.movie* pour ouvrir le fichier et lire les images successives de la vidéo dans la RAM, puis utilisez l'objet *jit.window* pour afficher le film dans une fenêtre séparée. Utilisez des arguments saisis pour spécifier les dimensions du film et les coordonnées précises de la zone d'affichage sur votre écran.

Sommaire

Les objets Jitter communiquent les informations relatives à une image vidéo particulière en s'envoyant mutuellement le nom d'une matrice - un emplacement en mémoire où se trouvent ces informations. Lorsqu'un objet Jitter reçoit le nom d'une matrice, il effectue sa tâche désignée en utilisant les données de cet emplacement, puis envoie le nom des données modifiées aux autres objets Jitter. Presque tous les objets Jitter envoient un nom (dans un message **jit_matrix**) lorsqu'ils reçoivent le message **bang** (ou **outputmatrix**). Ainsi, pour afficher les images successives d'une vidéo, il faut envoyer des messages **bang** à la fréquence souhaitée à un objet *jit.movie* connecté à un objet *jit.window*.



Traçage des messages et des rôles de chaque objet