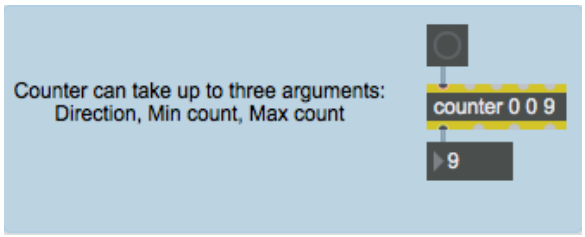
2- Quels sont les attributs?

Les ***attributs*** sont une nouvelle façon de spécifier le comportement des objets Max. La plupart des objets Jitter utilisent des attributs pour les différentes variables qui composent leur état interne actuel.



Le bon vieil objet *counter* de Max

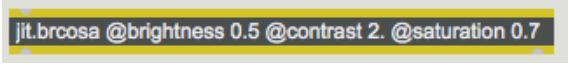
De nombreux objets Max, tels que l'objet *counter* présenté ci-dessus, prennent un certain nombre ***d'arguments*** pour déterminer leur comportement. L'ordre de ces arguments après le nom de l'objet comment l'objet les interprète. Dans l'exemple ci-dessus, le premier argument de *counter* définit la direction dans laquelle l'objet compte; les deuxième et troisième arguments déterminent les valeurs minimale et maximale entre lesquelles l'objet compte. Comme ces valeurs sont simplement attribuées à l'objet sous forme de nombres, leur ordre est important. Avec certains objets Max (*counter* en est un), le nombre d'arguments que vous donnez a une incidence sur la manière dont ils sont interprétés. Si vous fournissez à *counter* seulement deux arguments, ils seront compris par l'objet comme le compte minimum et maximum, et ***non*** comme le sens et le compte minimum. Comme la position et le nombre d'arguments sont cruciaux, il est impossible, par exemple, de créer un objet *counter* avec une direction et un nombre maximum prédéfinis en utilisant seulement deux arguments.

Les arguments d'un objet sont souvent considérés comme des valeurs ***initiales*** et les objets Max ont généralement des moyens pour modifier ces valeurs par le biais d'entrées supplémentaires dans l'objet ou de messages spéciaux que vous envoyez à l'objet. Vous pouvez modifier le sens et le nombre maximum d'un objet *counter* en envoyant des entiers dans sa deuxième et sa cinquième entrée, respectivement. Ces valeurs remplaceront les valeurs par défaut fournies par les arguments. De même, vous pouvez modifier le nombre minimum de l'objet en envoyant le message min suivi d'un nombre entier dans l'entrée gauche.

Bien que ce système fonctionne bien lorsqu'un objet Max ne comporte que deux ou trois variables qui définissent son comportement, les objets Jitter ont souvent beaucoup, beaucoup plus de variables (parfois des dizaines). Si toutes ces variables dépendaient de l'ordre des entrées et des arguments des objets, vous passeriez toute la journée à chercher dans le manuel de référence et vous n'auriez jamais le temps de travailler avec Jitter!

Définition des attributs

Les objets Jitter peuvent être indiqués pour comment se comporter en utilisant des ***attributs***. Vous pouvez taper des attributs dans une boîte d'objet avec le nom de l'objet Jitter, ou vous pouvez définir (et récupérer) des attributs par des messages Max après la création de l'objet.



Un objet Jitter avec des attributs après le nom de l'objet

L'objet Jitter présenté ci-dessus, appelé *jit.brcosa*, possède trois attributs saisis. Les attributs saisis sont définis dans les boîtes d'objet en utilisant le symbole @ suivi du nom de l'attribut et d'un ou plusieurs arguments (qui peuvent être n'importe quel type de données Max: ints, float, symbols ou lists). Vous pouvez saisir autant d'attributs que l'objet reconnaît, dans n'importe quel ordre, après le nom de l'objet. Bien que vous ne sachiez pas encore ce que fait l'objet *jit.brcosa*, vous pouvez en déduire quelques informations sur l'objet en vous basant sur les noms des attributs et sur le type de données qu'ils leur sont assignés.

> *Important*: il n'y a pas d'espace entre le signe @ et le nom de l'attribut saisi que vous voulez définir. Le caractère @ indique à l'objet Jitter d'interpréter le mot qui lui est attaché comme un nom d'attribut au lieu d'une valeur d'argument pour un attribut précédent.
>
> *Important également*: les objets Jittere peuvent avoir à la fois des attributs saisi set des arguments saisis. Voir la section **Arguments d'objet Jitter** ci-dessous pour plus de détails.

Comme pour les objets Max, les informations que vous fournissez à un objet Jitter pour définir son comportement initial sont généralement quelque chose que vous pouvez modifier après la création de l'objet. Les attributs peuvent être modifiés à tout moment en envoyant des messages à l'objet, comme indiqué ci-dessous.

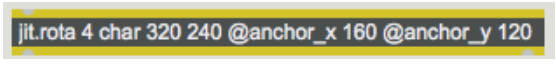
Les attributs peuvent être modifiés avec des messages Max

Cet objet *jit.brcosa* a son attribut de **brightness** fixé à **0.5.1** (saisi dans la boîte de l'objet comme "@brightness 0.5"), mais nous pouvons le changer en envoyant le message **brightness [float]** dans l'entrée gauche de l'objet. Vous pouvez modifier pratiquement n'importe quel attribut en envoyant un message contenant le nom de l'attribut, suivi des arguments pertinents, dans l'entrée gauche d'un objet Jitter.

Comme pour les objets Max, les objets Jitter ont des valeurs par défaut pour leurs paramètres. L'objet *jit.brcosa* ci-dessus n'a que des attributs saisis initialisant sa valeur de **brightness**, mais les autres attributs sont réglés sur leurs valeurs par défaut. Nous vous montrerons plus loin comment découvrir les attributs utilisés par un objet. Dans l'exemple ci-dessus, nous pouvons modifier les valeurs des attributs de **contraste** et de **saturation** de l'objet à l'aide de messages, en remplaçant les valeurs par défaut fournies par l'objet.

Arguments des objets Jitter

Il y a quatre informations que la plupart des objets Jitter utilisent et qui peuvent être entrées soit comme attributs saisis soit comme d'arguments saisis. En fait, ce sont toujours des attributs, mais les objets Jitter les traitent automatiquement et correctement lorsqu'ils sont utilisés comme arguments.



Les objets Jitter peuvent aussi avoir des arguments!

L'objet *jit.rota*, présenté ci-dessus, a clairement deux attributs initialisés: **anchor_x** et **anchor_y**. Mais que signifie le reste?
Si vous fournissez des arguments pour un objet Jitter qui traite les données de matrice Jitter (comme c'est le cas pour la plupart des objets Jitter), les arguments sont interprétés comme:

1.Le nombre de plans (**plancount**) de la matrice de sortie.
2. Le **type** de la matrice de sortie.
3.La taille ou «liste de dimensions» (**dim**) de la matrice de sortie.

Maintenant que nous savons cela, nous pouvons déterminer que l'objet *jit.rota* ci-dessus produira une matrice composée de 4 plans de données **char** (nombre entier de 8 bits) avec deux dimensions de **320** par **240** cellules.

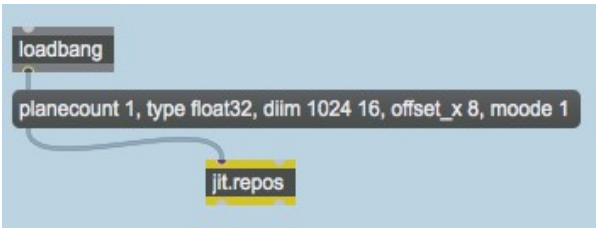
> *Important*: les arguments de l'objet Jitter, s'ils sont utilisés, doivent apparaître ***avant*** que les attributs ne soient définis. Sinon, l'objet Jitter interprétera mal les arguments comme des valeurs pour l'attribut et non des arguments pour l'objet.

Ces arguments peuvent également être définis par des attributs qui sont cohérents pour tous les objets Jitter qui produisent des données matricielles: **planecount**, **type** et **dim**. Ils peuvent être définis en tant qu'attributs saisis non ordonnés ou modifiés par des messages. Les trois objets ci-dessous, par exemple, sont identiques.

Arguments ou attributs? Vous décidez.

Les attributs de la matrice de sortie du premier objet sont définis à l'aide d'arguments **type**. Le deuxième objet a **planecount** et **type** définis à l'aide d'arguments de saisie, mais il utilise un attribut de saisie pour la liste de dimensions. Le troisième objet utilise un attribut typed-in pour tout définir.

Si vous préférez, vous pouvez initialiser les attributs d'un objet à l'aide des messages déclenchés à par un objet *loadbang*, comme indiqué ci-dessous.

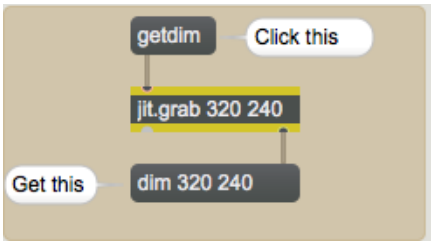


Encore une autre façon d'initialiser vos attributs

Interrogation des attributs et de l'état des objets

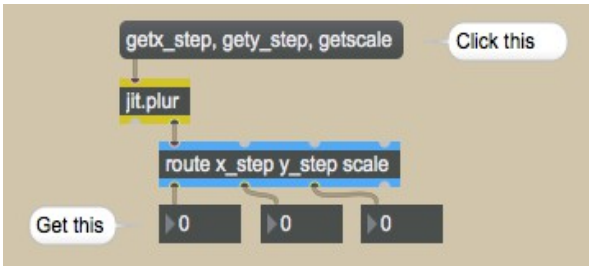
Le moyen le plus rapide de connaître les paramètres des attributs d'un objet est de consulter la fenêtre de ***l'inspecteur***. Vous pouvez y accéder en sélectionnant l'objet et en tapant commande/alt ou en cliquant sur l'icône i dans la barre d'outils de droite. Vous trouverez les paramètres d'attributs actuels au bas de la fenêtre.

Une caractéristique supplémentaire (et très utile) des attributs est que vous pouvez demander à un objet Jitter de vous dire quelle valeur il a actuellement stockée pour un attribut donné. Pour ce faire, vous ***interrogez*** l'attribut avec le message Max **get** et suivi (***sans espace***) du nom de l'attribut que vous souhaitez connaître. La valeur résultante est émise par l'objet Jitter sous la forme d'un message (commençant par le nom de l'attribut), envoyé par la sortie droite de l'objet.



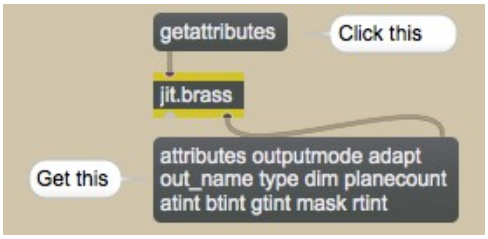
Interrogation d'un attribut pour un objet Jitter

L'utilisation de **get** pour trouver les valeurs des attributs vous permet de découvrir la valeur actuelle d'un attribut, même si vous n'avez jamais défini l'attribut en premier lieu. Par exemple, le patch ci-dessous découvre certaines des valeurs par défaut de l'objet *jit.plur*. L'objet *route* de Max vous permet de séparer facilement les valeurs de chacun des attributs.



Connaître les valeurs par défaut des attributs d'objet

Deux messages que vous pouvez envoyer à n'importe quel objet Jitter, **getattributes** et **getstate**, affichent tous les attributs utilisés par l'objet.

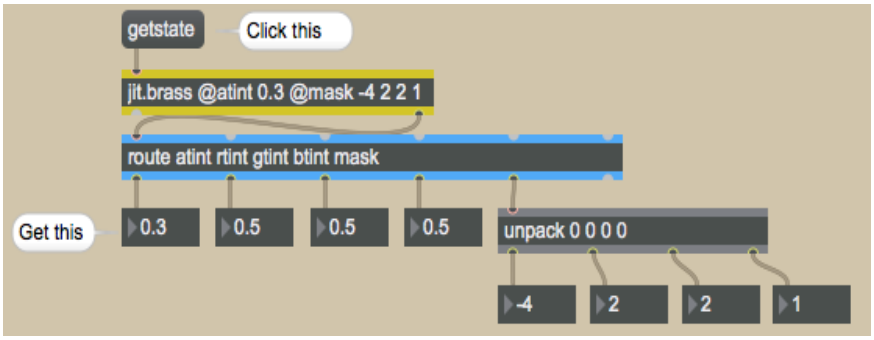


Connaître vos options…

Le message **getattributes** amène l'objet Jitter à sortir les **attributs** du message, suivis d'une liste de tous les symboles d'attributs que l'objet Jitter peut comprendre. En expérimentant avec quelques objets Jitter, vous verrez rapidement que beaucoup d'entre eux, comme **outputmode**, **type** et **dim**,

sont assez standards. D'autres (tels que *mask* dans l'objet *jit.brass*) auront une signification particulière pour l'objet qui les utilise.

Le message **getstate** déverse tous les attributs de l'objet Jitter comme si toutes les requêtes d'attributs imaginables avait été effectuées en une seule fois.



Trouver l'état d'un objet

Vous pouvez ensuite utiliser les objets *route*, *unpack* et autres objets Max pour extraire les attributs selon vos besoins. Plus loin dans les tutoriels, vous rencontrerez plusieurs objets Jitter dont les attributs changent en fonction de calculs effectués sur la matrice d'entrée (ou d'un fichier qui vient d'être ouvert par l'objet). L'interrogation des attributs pertinents permet de connaître le résultat du calcul de l'objet.

Attrui

Vous pouvez également lire les valeurs des attributs avec l'objet *attrui*. Il s'agit d'un widget d'interface utilisateur qui semble être connecté à l'entrée d'un objet mais qui, en fait, a une communication bidirectionnelle. L'objet *attrui* comporte deux sections - la section de gauche est un menu déroulant dans lequel sont disponibles tous les attributs de l'objet connecté. Une fois qu'un attribut a été choisi, la partie droite affiche la valeur actuelle de l'attribut et vous permet de la modifier.

Sommaire

Les attributs de Jitter sont un outil puissant pour gérer les paramètres d'objets complexes. Vous pouvez utiliser des attributs pour initialiser, modifier et découvrir les valeurs actuelles stockées dans les objets Jitter, et l'attachement de chaque valeur à un nom d'attribut fixe élimine le besoin de se souvenir de l'ordre des arguments saisis ou le rôle de chaque entrée dans un objet complexe.