



Guide de l'utilisateur

Serge Rosmorduc

Table des matières

Introduction	5
Présentation de l'interface de JSesh	6
Éditer des textes.....	7
Édition avec la souris.....	7
Positionner le curseur	7
Insérer des signes avec les menus et la palette.....	7
Sélectionner une zone.....	10
Ajouter un signe sélectionné depuis un menu	10
Grouper des signes.....	10
Ligatures.....	11
Édition à l'aide du clavier	12
Saisie des hiéroglyphes.....	12
Codes phonétiques	13
Signes groupés	14
Exemple.....	14
Saisie directe de codes du Manuel de codage.....	14
Éditeur de glossaire.....	15
Ombrer/Hachurer un texte	16
Ombrer une partie de cadrat	18
Options courantes pour réaliser des ombrages difficiles	19
Présentation des documents.....	21
Préférences de dessin	21
Préférences d'exportation.....	21
Exportation des graphiques.....	22
Copier/coller.....	22
Choix du format	22
Réglages fins	23
Limitations du copier-coller.....	24
Menu "Copier" avancé.....	24
Possibilité de coller le texte hiéroglyphique dans JSesh	24
Les éditeurs Mellel, Nissus et le PDF	25
NeoOffice (OpenOffice) et texte RTF	26
Autres.....	28
Impression	28
Export en PDF	28
Export en JPEG ou PNG.....	29

Export en RTF.....	29
Problèmes spécifiques aux traitements de texte	29
Choix de format pour les applications sur Macintosh	29
Choix de format pour les applications Windows	30
Choix de format pour les applications Linux	30
Trucs et astuces sur l'encodage de textes hiéroglyphiques	31
Encodage de textes hiératiques.....	32
Disposition des signes	32
Translittération.....	34
Utiliser une police unicode.....	34
Problèmes actuels avec les polices Unicode	35
Ne pas utiliser unicode.....	36
Méthode simple avec l'ancienne police	36
Utilisation de vos propres polices compatibles MdC.....	37
Explication des menus de JSesh.....	38
Fichier	38
Édition.....	40
Manipulation de groupes	42
Signes	44
Fenêtre.....	45
Aide	46
Extension de la liste des signes	47
Présentation	47
Importation de nouveaux glyphes	47
Choisir un dossier pour vos signes.....	47
Insertion d'un nouveau signe	48
Importation de dessins	49
Affiner vos signes.....	49
Attacher des codes aux dessins et les insérer dans JSesh	49
Système alternatif pour l'insertion de signe	51
Création d'un signe avec Inkscape	52
Présentation	52
Quelques notions sur le dessin vectoriel	52
Création d'une image de fond	52
Dessiner le contour.....	53
Ajuster le contour.....	53
Travail sur les détails	55
Choisir la bonne largeur de ligne	59

Unir tout	59
Ajout d'autres détails	60
Commentaires finaux	60
Informations avancées dans Inkscape	61
Présentation	61
Zones de ligature	61
Gravité des zones de ligature	62
Parties de SVG comprises par JSesh	63
Annexe A. Le système actuel de description des signes.....	64
Démarrage de l'éditeur de description de signes	64
Modification des descriptions des signes.....	64
La fenêtre principale de l'éditeur d'informations sur les signes	65
Translittération	66
La fenêtre de création de tags	70
Les menus.....	70
Contribution de votre description de signes à JSesh.....	71
Annexe B : Informations sur le fichier de description des signes	72
DTD des descriptions de signes	73
Annexe C : Guide du développeur	77
Obtenir le code	77
Compilation du code	77
Comment faire.....	78
Pour ajouter champ d'édition hiéroglyphique dans interface SWING. 78	
Pour produire une image bitmap à partir d'un texte MDC.....	79

Introduction

Vous utilisez JSesh, qui est à la fois un éditeur de textes hiéroglyphiques et un ensemble d'outils dédiés à la manipulation des textes hiéroglyphiques en Java. En tant qu'utilisateur, ce sont sans doute les capacités d'édition et d'impression de JSesh qui vous intéresseront davantage.

JSesh couvre la plupart des spécifications du *Manuel de codage*¹ et peut lire des fichiers provenant d'un grand nombre d'autres logiciels d'édition hiéroglyphiques, tels que WinGlyph² et tksesh. La compatibilité avec MacScribe³ n'a pas encore été vérifiée, et n'est sans doute pas totale.

Le *Manuel de codage* est un standard créé en 1984 pour décrire les textes hiéroglyphiques en ASCII. Il est un peu ancien aujourd'hui, et il existe de nombreuses suggestions pour l'améliorer (voire le remplacer). JSesh proposera plusieurs extensions. JSesh vous permet d'éditer des textes hiéroglyphiques soit en tapant les codes du Manuel de Codage, soit par un système de menus plus intuitif. Certains éléments du Manuel du Codage ne sont pas encore disponibles via des menus, mais vous avez toujours la possibilité de saisir le code directement.

JSesh a de nombreuses possibilités de sortie. Il peut imprimer un fichier ou bien le sauvegarder dans des formats graphiques aussi nombreux qu'intéressants : pdf, jpg ou même en tant qu'ensemble de fichiers html. Parmi les nombreux formats de sortie, on peut noter le WMF (Windows MetaFile). Les *metafiles* sont des images vectorielles, ce qui signifie qu'ils conviennent bien à l'impression. Les fichiers WMF peuvent être lus par la plupart des logiciels de traitement de texte.

JSesh, contrairement à tksesh, n'est pas en premier lieu une base de données de textes. La fonctionnalité de base de données textuelle de tksesh sera ajoutée à JSesh ultérieurement, mais l'option préférable de fournir un bel éditeur d'abord, de façon à satisfaire le plus grand nombre.

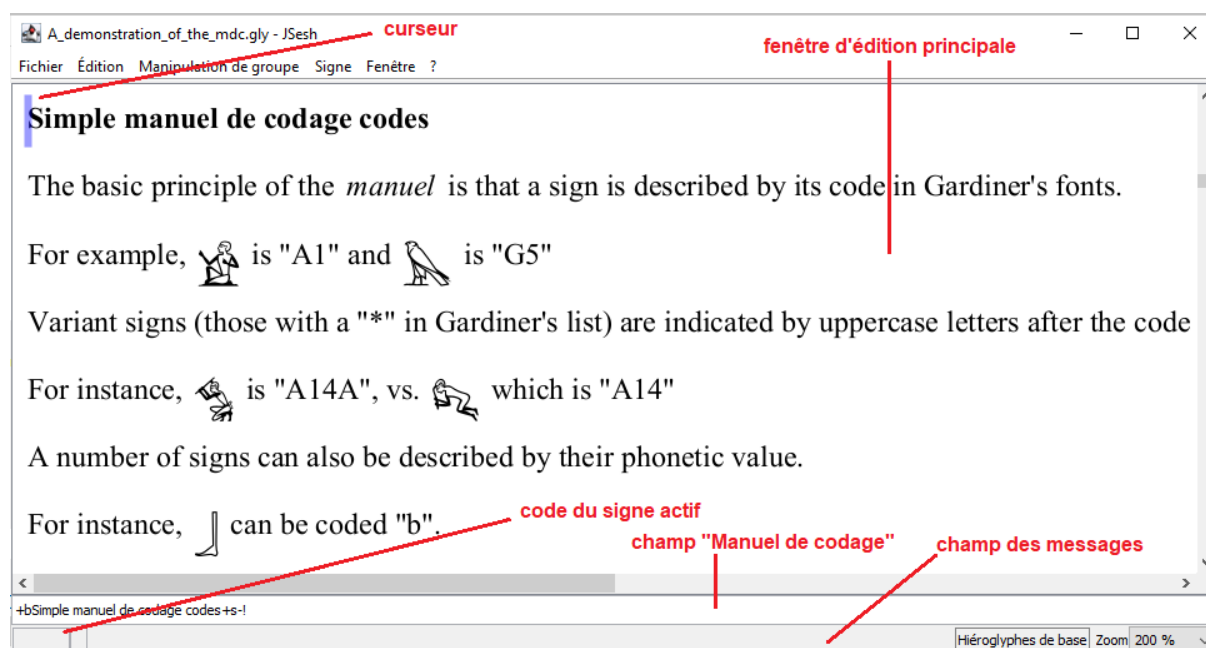
1

https://fr.wikipedia.org/wiki/Manuel_de_codage_des_textes_hiéroglyphiques_en_vue_de_leur_saisie_informatique (toutes les notes sont de l'auteur du présent fichier PDF).

² Logiciel obsolète, ne fonctionne plus avec Windows 10, et son éditeur, le CCER, a cessé ses activités en 2010 (voir <http://ancientworldonline.blogspot.com/2010/09/news-from-centre-for-computer-aided.html>).

³ Fin de vie en 2017.

Présentation de l'interface de JSesh



L'espace de travail de JSesh (version 7.5.5 Windows)

Actuellement, l'espace de travail de JSesh contient une *barre de menu*, une *fenêtre d'édition principale* (avec la *palette des signes* par-dessus), et un certain nombre de champs.

Le champ intitulé *code du signe actif* est utilisé dans l'édition interactive pour indiquer le code pour le texte en cours d'édition.

Le champ intitulé *Manuel de Codage* contient le code de la ligne de texte active (celle sur laquelle se situe le curseur), au format du Manuel de Codage. Vous pouvez modifier directement ce code.

Le champ immédiatement à droite de ce dernier est un champ de messages. Il n'a pas vraiment d'utilité pour le moment.

Éditer des textes

Édition avec la souris

Éditer avec la souris est un processus simple, mais lent. En général, il sera utilisé conjointement avec l'un des deux autres modes d'édition.

Positionner le curseur

Pour positionner le curseur, il suffit de cliquer dans la fenêtre d'édition principale.

Insérer des signes avec les menus et la palette

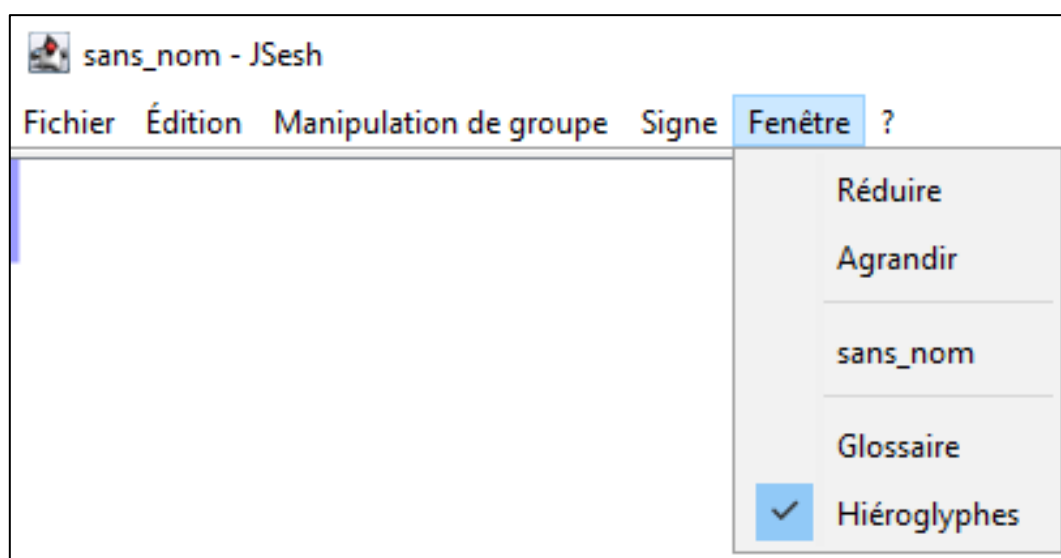
La méthode la plus rapide d'insérer des signes est d'utiliser le clavier (voir la section *Édition au clavier*), et d'en saisir soit le code au format du Manuel de Codage, soit la translittération. Toutefois, il est possible de sélectionner des signes depuis un menu accessible par un bouton ou depuis la palette des signes.

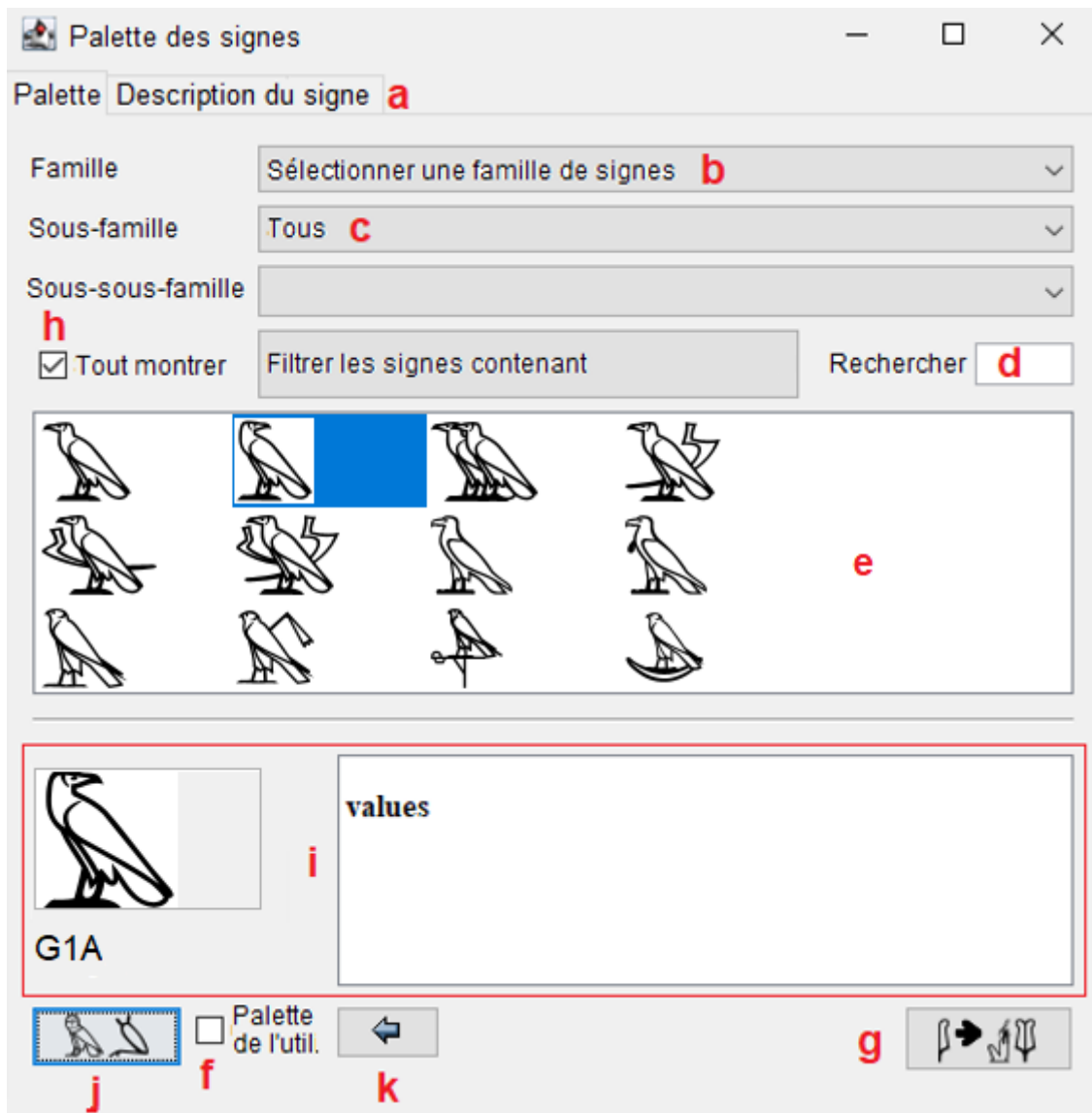
Le bouton **Hiéroglyphes de base**

Le menu du bouton **Hiéroglyphes de base** liste seulement les signes basiques à partir des catégories Gardiner classiques (auxquelles s'ajoutent la catégorie Ff des signes issus du hiératique - cette catégorie étant utilisée dans la fonte Gardiner originelle). Le bouton se trouve en bas à droite de la fenêtre de JSesh.

La Palette des signes

La palette des signes a été créée pour fournir une meilleure solution que le menu du bouton **Hiéroglyphes de base**. Elle peut afficher tous les signes disponibles sans problème et comporte des options de recherche avancée. La palette des signes s'affiche ou s'efface en utilisant le menu **Fenêtre > Hiéroglyphes**.





Palette des signes (toutes les copies d'écran sont de la version 7.5.5 pour Windows)

L'utilisation basique de la palette est simple : sélectionner la famille de signes désirée en utilisant la liste déroulante (b) puis double-cliquer dans le panneau (e) sur le signe que l'on souhaite insérer dans la fenêtre d'édition de JSesh.

Tous les signes ne sont pas affichés (par défaut, la palette n'affiche pas les variantes de signes). Pour afficher tous les signes d'une famille, cocher la case *Tout montrer* (h). Cliquer sur un signe sélectionne celui-ci. Les informations au sujet de ce signe (dessin agrandi, code, valeurs, etc.) apparaissent dans le panneau (i). Plus d'informations sont éventuellement disponibles en sélectionnant l'onglet *Description du signe* (a).

N.B. : La plupart des informations dans JSesh sont indiquées à des fins de recherche. En particulier, les détails concernant la signification des valeurs ne sont pas exhaustivement affichés. Ainsi, nous utilisons "bin" comme valeur pour G37, mais seulement parce qu'il s'agit d'un moyen mnémotechnique commode. En réalité, le format interne de JSesh permet de stocker davantage d'informations sur ce que sont les valeurs. Un autre exemple est Y1 ; pour ce signe, nous avons conservé le moyen

mnémotechnique "sš", et ajouterons le probablement meilleur "sh3". Toute information plus scientifique sur le signe est du ressort de l'onglet *Description du signe* (qui est plutôt vide pour le moment).

Contrôles avancés

- **b** : Le sélecteur de familles contient également deux familles spéciales : *0. Derniers signes utilisés* et *1. Palette de l'utilisateur*. Voir ci-dessous pour plus d'informations sur la palette de l'utilisateur. *Derniers signes utilisés* contient tous les signes utilisés par l'utilisateur au cours de la session. Ceci est sensé être utile si les mêmes signes apparaissent fréquemment.
- **c** : Le contrôle *Sous-famille* permet de restreindre la liste aux signes qui partagent certaines caractéristiques. Par exemple, si la famille "God" (dieux) est sélectionnée, la recherche peut être restreinte aux divinités à tête de faucon ("hawk-headed").
- **d** : Le filtre de translittération permet de rechercher des signes à partir de leur translittération. La translittération peut être la valeur phonétique d'un signe (pour les phonogrammes et les idéogrammes), ou peut être une valeur typique d'un mot entier ou d'un mot dans lequel le signe apparaît. Si la case *Tout montrer* est cochée, toutes les valeurs connues de JSesh seront utilisées.
- **f** : Sélecteur de la palette de l'utilisateur. Si cette case est cochée, le signe en cours de sélection sera ajouté à la palette de l'utilisateur.
- **g** : En cliquant sur le bouton *Partie de*, JSesh affichera tous les signes qui contiennent le signe en cours de sélection (du moins, s'il les connaît). À l'avenir, il pourrait utiliser la boîte *Tout sélectionner* pour choisir jusqu'à quel point aller. Chaque clic supplémentaire étendra l'ensemble des signes affichés (le suivant listera les parties, puis les parties de parties, etc.).
- **j** : Sélectionne des variantes connues du signe courant. Le terme *variante* est employé ici au sens large. Il peut désigner une réelle variante linguistique (ainsi Z7 est une variante de G43), tout comme il peut signifier *signes qui sont graphiquement basés sur un autre*. Par exemple, A17A est une variante de A17 quant à sa signification, même si les utilisations linguistiques présentent quelque différence. Chaque clic supplémentaire étend l'ensemble, un second clic intégrant les variantes de variantes, etc.

La palette de l'utilisateur

La palette de l'utilisateur permet à tout utilisateur de composer sa propre liste de signes favoris. Ajouter un signe à la palette est simple : sélectionner le signe et cocher la case *Palette de l'utilisateur* (f). Le contenu de la palette de l'utilisateur est automatiquement sauvegardé, ce qui fait que la palette est disponible à chaque nouveau lancement de JSesh. Enlever un signe de

la palette de l'utilisateur est tout aussi simple : il suffit de sélectionner le signe concerné et de décocher la case *Palette de l'utilisateur*.

Tous les signes sélectionnés pour une inclusion dans la palette de l'utilisateur sont affichés lorsque la famille spéciale *Palette de l'utilisateur* est sélectionnée.

Vous pouvez aider

Vous trouverez une description des fichiers utilisés par la palette dans un appendice à cette documentation. Si vous avez une bonne connaissance des hiéroglyphes, vous pouvez aider à améliorer JSesh en étendant les informations que la palette utilise. Pour plus de renseignements, envoyez un courriel à l'auteur (serge.rosmorduc AT qenherkhopeshef.org).

Sélectionner une zone

La sélection se situe entre le curseur et ce qui est appelé la *marque*. L'ensemble de la sélection est surligné en bleu clair. Certaines opérations ne sont possibles que si une zone de texte est sélectionnée.

Il existe divers moyens de sélectionner une zone de texte :

- façon Mac : shift-clic à un endroit du texte : l'ensemble du texte entre cet endroit et le curseur sera sélectionné ;
- façon Unix : idem, mais en utilisant le clic-droit. Cela pourrait changer si je décide un jour d'utiliser les menus contextuels ;
- sélection au clavier : shift + flèche gauche ou droite pour déplacer l'étendue de la sélection ;
- Mac, Unix et Windows : cliquer-glisser à l'aide de la souris.

Ajouter un signe sélectionné depuis un menu

Le menu du bouton *Hiéroglyphes de base* donne accès aux signes de la liste Gardiner standard. Sélectionner un signe l'ajoute à la position du curseur.

Grouper des signes


Cela est fait à partir du menu *Manipulation de groupe*. Il est possible de grouper les signes en groupes verticaux ou horizontaux, en sélectionnant les signes (ou cadrats), puis en choisissant soit *Grouper horizontalement* soit *Grouper verticalement*. N.B. : Il est possible de dégroupier les signes avec le menu *Séparer*. Tous les types de groupes peuvent ainsi être disjoints.




Grouper horizontalement a pour raccourci clavier "Ctrl-H".

Grouper verticalement a pour raccourci clavier "Ctrl-G".

Ligatures




JSesh connaît un ensemble de groupes particuliers, appelés *ligatures*, qui sont au-delà des capacités des cadrats. Par exemple, "w" et "t" seront le

plus souvent disposés ainsi : . Pour réaliser une telle mise en forme, sélectionner les signes à ligaturer et utiliser le menu **Manipulation de groupe > Ligaturer**.


Toutefois, JSesh ne sait pas ligaturer tous les groupes, même s'il connaît quelques graphies sportives : ligaturer   donne .

Ligatures complexes

Certaines ligatures peuvent être considérées comme un signe et un groupe.

Prenons par exemple le groupe . Il peut être interprété comme une ligature entre le signe  et le groupe . De telles ligatures peuvent être réalisées à l'aide du menu **Manipulation de groupe > Ligaturer groupe et hiéroglyphe** ou **Manipulation de groupe > Ligaturer hiéroglyphe et groupe**. Dans le premier cas, le groupe se situe à l'avant du hiéroglyphe, dans le second cas (qui correspond à notre exemple), le hiéroglyphe se trouve à l'avant du groupe.

Pour chaque signe, JSesh essaie de trouver deux zones. Une pour les ligatures *antérieures*, une pour les ligatures *postérieures*. Certains signes ont leur ligatures prédéfinies. Pour d'autres signes, celles-ci sont calculées automatiquement. En vérité, JSesh essaie de voir s'il peut adapter un rectangle dans la zone inférieure gauche des signes (pour l'orientation de gauche à droite). Cette zone serait la zone de départ. Pour la zone de fin, deux endroits sont examinés. Tout d'abord, le coin supérieur droit des signes (comme dans la ligature), et ensuite une large zone en bas à gauche, comme dans notre exemple. Le créateur de signes peut fournir d'autres

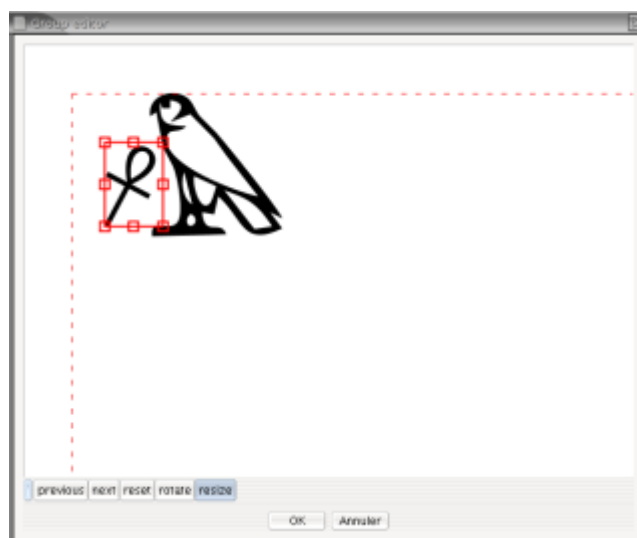
zones, ainsi : .

N.B. : Un signe peut être ligaturé simultanément avec le groupe qui le précède et celui qui le suit.

Éditer des groupes

Lorsqu'il existe un groupe très spécifique, d'un type unique, dans lequel les signes doivent être disposés avec précision et d'une façon bien particulière, il est possible de sélectionner le menu **Éditer le groupe**. Le groupe édité sera soit le groupe sélectionné, s'il y a une sélection (surlignée en bleu clair), soit le dernier groupe avant le curseur, s'il n'y a aucune sélection.

Ce menu ouvre la fenêtre suivante :



Il sera alors possible de déplacer les signes, les mettre à l'échelle et les tourner.

N.B. : Les mises à l'échelle et les rotations sont déclenchées par deux boutons. Pour déplacer un signe, il suffit de cliquer dessus et de le déplacer où on le veut. Pour le tourner ou le mettre à l'échelle, il suffit de cliquer-glisser l'une des petites cases rouges situées autour du signe en cours de sélection.






Édition à l'aide du clavier

Saisie des hiéroglyphes

Pour saisie des signes à l'aide du clavier, cliquer sur la fenêtre principale (celle qui affiche le texte hiéroglyphique en cours, et non celle avec le *Manuel de codage*). Lorsque vous saisissez une lettre ou un nombre dans la fenêtre principale, cette lettre ou ce nombre apparaissent dans le petit panneau en bas à gauche de l'écran de JSesh. Ceci vous permet de saisir des codes au format du Manuel de codage.

De nombreux signes ont un code phonétique qui correspond à leur translittération. Tous les signes sont accessibles par leur code selon la police Gardiner. Pour connaître ce code, vous pouvez vous reporter au menu **Hiéroglyphes**. Une fois le code saisi, vous devez le valider. Pour ce faire, appuyez au choix sur les touches **Espace**, **':'**, **'*'**, **'-'** ou **Entrée**. Le signe sera ajouté au texte hiéroglyphique.

Les codes phonétiques peuvent correspondre à plus d'un signe.

Par exemple, *iw* correspond à , , ,  et .

Le signe "officiel" pour *iw* est  selon le MdC. Toutefois, vous pouvez désirer que ce soit un autre signe. Dans ce cas, c'est très simple : appuyez

sur **Espace** et le système fera défiler toutes les possibilités, une par une. Votre dernier choix sera enregistré pour la prochaine fois que vous saisirez le même code au cours de la session.

Codes phonétiques

Les signes unilitères possèdent les codes suivants, qui sont également utilisés lorsque l'on saisit leur translittération :

Code	Lettre
A	ʾ (aleph)
i	י (yod)
y	י
a	ʿ (ayin)
w	w
b	b
p	p
f	f
m	m
n	n
r	r
l	l
h	h
H	ח
x	ח
X	ח
z	z
s	s
S	s
q	ק
k	k
g	g
t	t
T	ט
d	d
D	ד

W est également utilisé pour Z7, M pour Aa15, et N pour S3.


Signes groupés

Le signe que vous saisissez au moment d'une validation peut être utilisé pour grouper des hiéroglyphes.

- 'Espace' et '-' ont pour effet d'ajouter le signe suivant dans un nouveau cadrat.
- ':' et '*' ajouteront respectivement le signe suivant en dessous et à côté du dernier signe saisi.
- 'Entrée' a deux conséquences : cela valide un signe (s'il y en a un), et cela ajoute une nouvelle ligne.

Si aucun code n'a été entré (c'est-à-dire, si la fenêtre de code est utilisée), la saisie de groupement de signes précédentes groupera les deux derniers cadrats. Cela paraît un peu étrange, mais vous verrez que c'est assez naturel. Vous pouvez vous en servir pour grouper des signes après coup

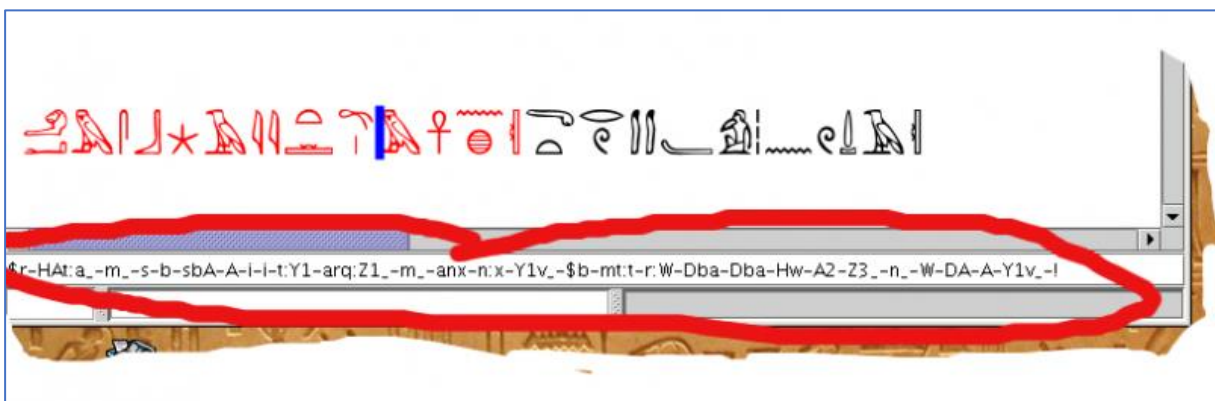
Exemple

Si vous voulez saisir le mot , vous pouvez :

- saisir 'p', puis '*', 't', ':', 'pt'
- saisir 'p', 't', puis '*', Espace (ce qui, en quelque sorte, valide le '*'), pt, ':', et Espace.

Saisie directe de codes du Manuel de codage

Lorsqu'un texte s'affiche, les codes du Manuel de codage pour la ligne en cours sont visibles dans un champ texte :



Ce champ n'est pas là que par amour de la beauté du Manuel de codage.

En fait, vous pouvez éditer son contenu, et le texte dans la fenêtre d'édition

principale se mettra à jour en conséquence lorsque vous validerez (en appuyant sur la touche **Entrée**).

Notez qu'un code incorrect (selon le Manuel de codage) sera rejeté.

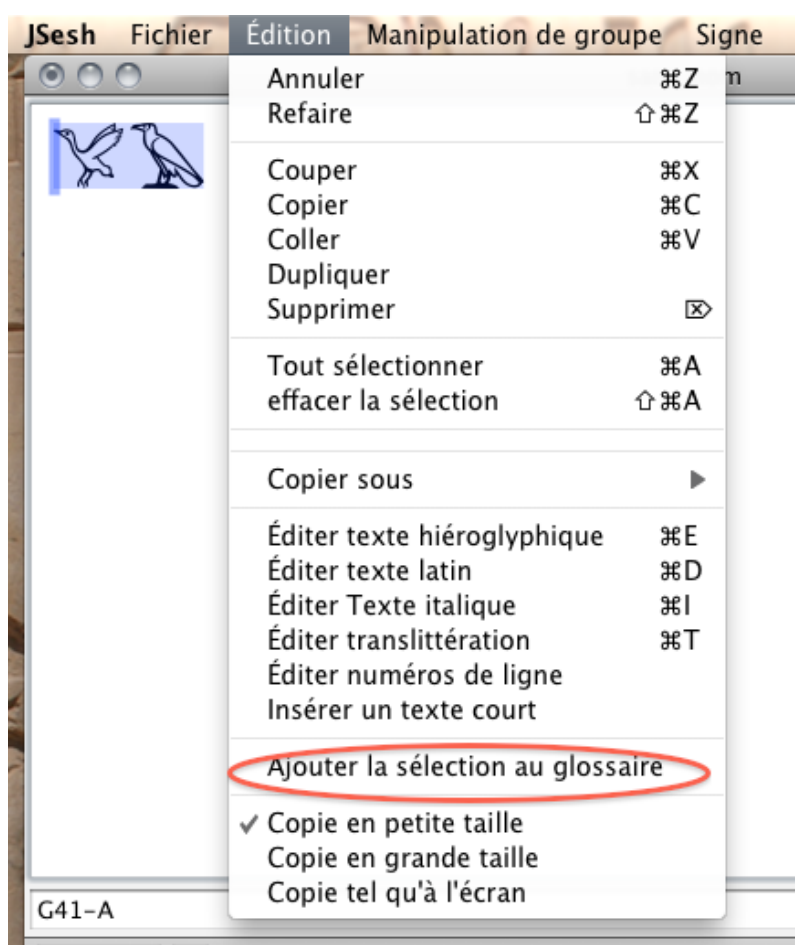
Je n'utilise ce système que dans de rares cas (par exemple, pour corriger des erreurs dans un texte en alphabet 'latin').

Éditeur de glossaire

L'éditeur de glossaire permet de créer des groupes précomposés (y compris des mots ou des phrases) et de les retrouver facilement pour une saisie rapide.

Supposons que vous souhaitez saisir l'élément p^3 dans sa forme hiératique, qui est G41-A1, simplement en saisissant "pA". Voici ce que vous devez faire :

- saisissez le texte pour le groupe que vous voulez ;
- sélectionnez-le, et appelez le menu **Edition > Ajouter la sélection au glossaire**.



- la fenêtre de l'éditeur de glossaire s'affiche, et tout ce que vous avez à faire est de taper le code que vous souhaitez utiliser, puis de cliquer sur le bouton **Ajouter**.



Aussitôt que le groupe a été ajouté, vous pouvez l'insérer en saisissant son code, de la même façon que vous saisissez le code d'un signe.

Dans cet exemple, nous avons utilisé "pA" comme code ; "pA" permettant de saisir G41 et G40, la barre d'espacement vous permettra de faire défiler toutes les solutions.

Compléments

- vous pouvez supprimer des groupes précomposés en cliquant sur le bouton "Supprimer" qui les suit dans l'éditeur de glossaire ;
- vous pouvez ouvrir ou fermer l'éditeur de glossaire en utilisant le menu **Fenêtre > Glossaire**.

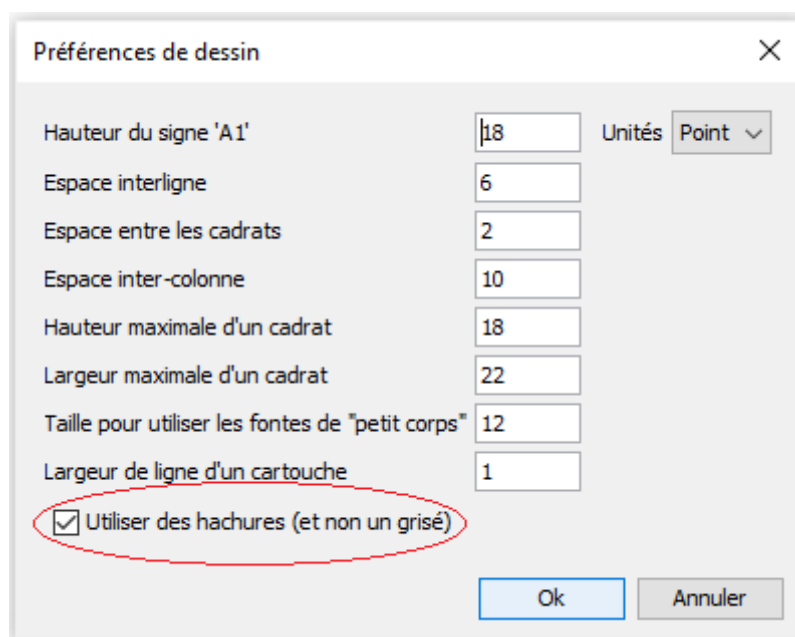
Ombre/Hachurer un texte

Ombre ou hachurer est actuellement un peu complexe. La mauvaise nouvelle, c'est ce que cela va empirer.

En tant qu'option (**Fichier/Propriétés du document**), l'ombrage peut être rendu soit par les classiques lignes entrecroisées que l'on peut voir dans les anciennes publications, soit par un fond gris clair.

Bien que de nombreux utilisateurs préfèrent la première solution, plus traditionnelle, la seconde est meilleure sur un plan typographique.


Les lignes entrecroisées rendent difficile la lecture des signes, ce qui n'est pas le cas d'un fond gris clair.



L'ombrage se présente de nombreuses façons. Cela est dû à l'histoire du Manuel de Codage.

Un bon résumé est disponible dans le document de démonstration⁴ de JSesh (celui qui apparaît lorsque l'on lance JSesh, et qui est désormais inclus dans la bibliothèque de textes de JSesh).

Néanmoins, pour dire toute l'histoire :

- Originellement, il existait deux systèmes d'ombrage. Le premier ombrait une partie entière du texte : c'est celui que vous obtenez quand vous utilisez **Ombrer une zone**. Dans JSesh, il fonctionne sur le texte sélectionné.
- Le second système d'ombrage utilisait des symboles d'ombrage, utilisés de la même façon que des hiéroglyphes. Dans le manuel originel, les cadrats étaient ombrés en les recouvrant avec les symboles d'ombrages correspondants. Vous pouvez les obtenir depuis le menu **Symboles d'ombrage**. Ainsi,  peut être obtenu en saisissant "wn", puis en ajoutant un symbole d'ombrage, et en empilant les deux verticalement. JSesh comprend ce genre d'ombrage depuis le début, mais le menu pour l'utiliser n'est disponible que depuis la version 2.10.
- ensuite, aux alentours de 1994, un nouveau système d'ombrage fut proposé, qui couvrait des cas plus simples. C'est celui que l'on obtient avec le menu **Ombrage**. Il s'applique au groupe en cours (celui devant le curseur), et peut être utilisé pour ombrer n'importe quel quartier d'un groupe.
- MacScribe⁵ était capable d'ombrer les quartiers de signes individuel. Sur MacScribe, "p##13*p:pt" montrerait le signe "p" à moitié

⁴ disponible en PDF (en anglais) ici :

https://www.shpylgoreih.fr/documents/Jsesh_and_the_Manuel_de_codage.pdf

⁵ Le premier éditeur hiéroglyphique sur Macintosh, publié en 2001. N'existe plus.

ombré. Une variante de ce système est comprise par JSesh. Le logiciel externe [MacScribe convertir](#)⁶ comprenait cela.

- Un système d'ombrage *libre* serait agréable également. Dans ce cas, les limites de la zone d'ombrage seraient librement tracées. Je prévois également de l'ajouter (en prenant appui sur l'éditeur de groupe). Mais pour le moment, si vous avez un tel besoin de précision, la seule option que JSesh vous donne est d'exporter votre fichier dans un format pratique (par exemple SVG) et de le modifier sous cette forme.

Ombre une partie de cadrat

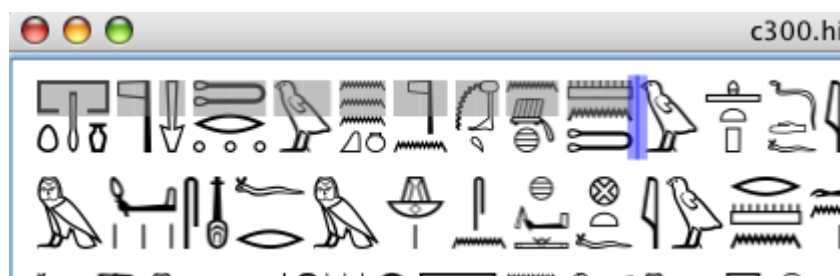
Le menu **Manipulation de groupe > Hachures** peut s'appliquer à une sélection. En fait, vous pouvez ombrer la partie supérieure de tous les groupes d'une sélection (auparavant, il fallait ombrer chaque groupe un par un).

En outre, pour ceux qui saisissent des textes comportant de nombreuses lacunes, j'ai ajouté un menu popup, qui apparaît en appuyant sur la touche “#” du clavier. Vous pouvez y naviguer à l'aide du clavier ou sélectionner des entrées par lettres spécifiques.

Voici un exemple : j'ai sélectionné une partie du texte et tapé “#”. Le menu popup apparaît :



Maintenant je peux sélectionner l'ombrage dont j'ai besoin de différentes façons (cela dépend de votre système d'exploitation). Par exemple, je peux taper “3”, ou me déplacer dans le menu à l'aide des touches fléchées du clavier ou sélectionner l'option de menu avec la souris. Ensuite, le texte ainsi sélectionné est entièrement ombré :

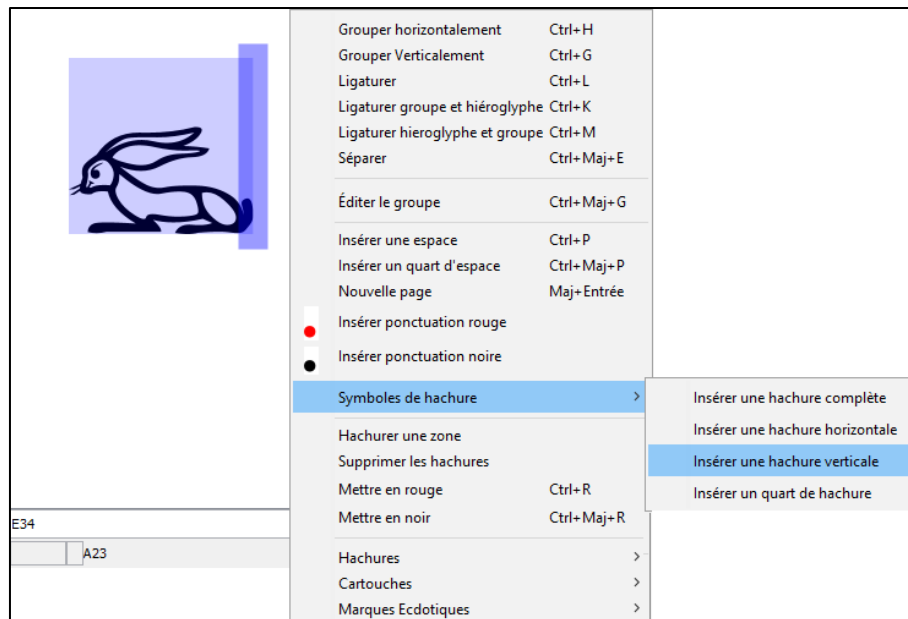


⁶ Semble ne plus exister à la date d'édition du présent PDF.

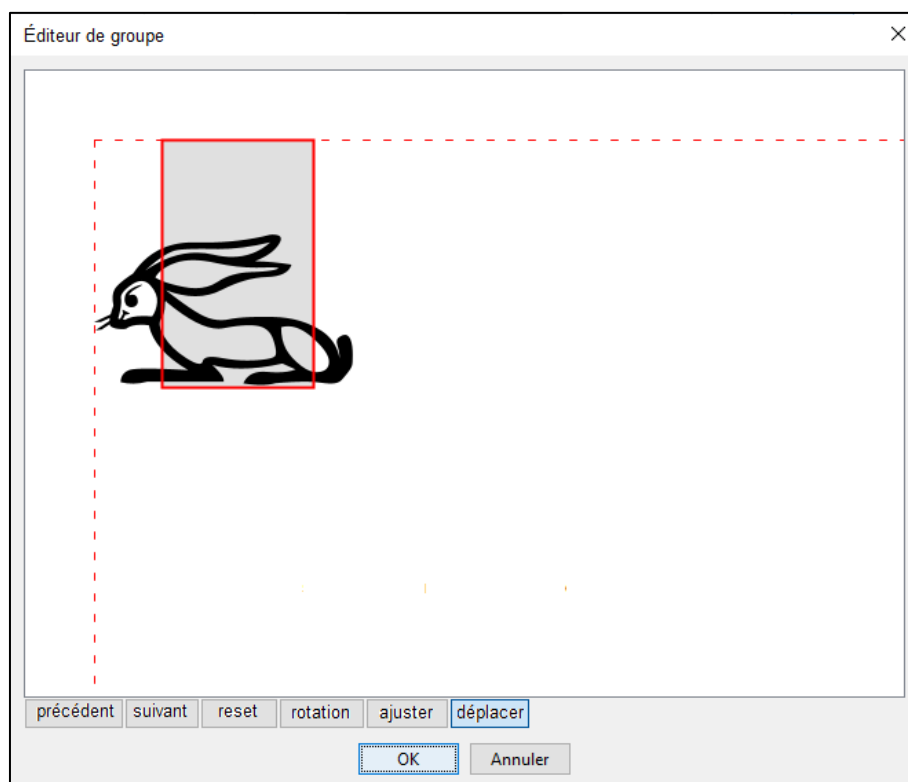
Options courantes pour réaliser des ombrages difficiles

Si vous souhaitez, par exemple, ajouter une zone d'ombrage au-dessus du milieu d'un signe, vous pouvez désormais le faire. En fait, c'est la raison pour laquelle j'ai ajouté un menu pour retrouver les "anciens" symboles d'ombrages.

Admettons que vous vouliez créer . Vous devez d'abord saisir ces deux signes. L'ombrage est ajouté à travers les menus :



Ensuite, vous n'avez plus qu'à utiliser l'éditeur de groupe pour déplacer l'ombrage :



Autant que possible, il faudrait saisir le symbole d'ombrage en premier, et ensuite le hiéroglyphe. Ainsi, si votre format de rendu final ne comprend pas la transparence, vous aurez quand même un rendu correct.

(Au passage, les marques ecdotiques comme [...] sont considérées comme des hiéroglyphes. À ce titre, elles peuvent être déplacées à l'aide de l'éditeur de groupe).

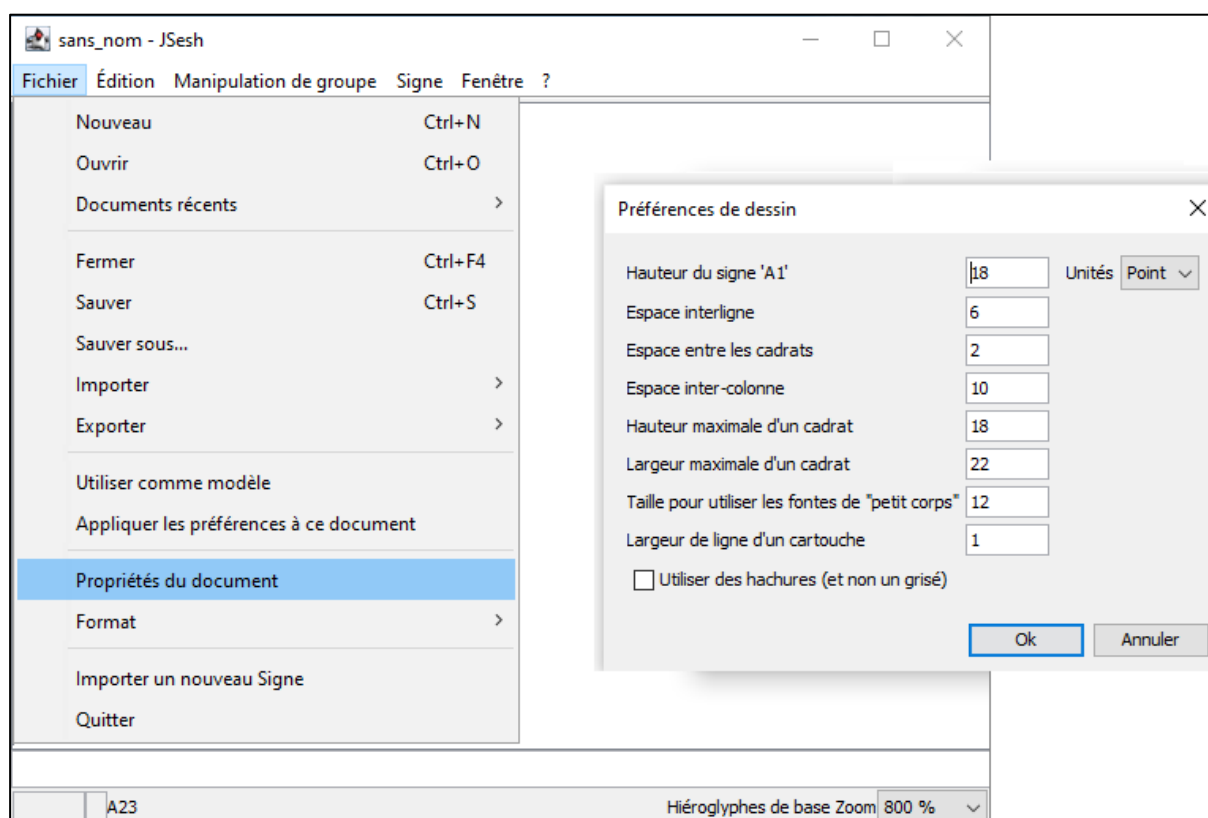
Présentation des documents

JSesh vous permet de définir plusieurs options quant à l'apparence finale des documents. Vous pouvez ainsi modifier l'espacement des lignes, la taille des signes, etc.

Depuis JSesh 5, la plupart des informations concernant le rendu sont sauvegardées avec chaque document. Les changer modifiera seulement le texte en cours d'édition.

Préférences de dessin

Les paramètres de préférences pour les dessins sont accessibles à partir du menu **Fichier > Propriétés du document** :



Préférences d'exportation

Les paramètres de préférences pour les formats d'exportation des graphiques sont accessibles à partir du menu **Édition > Préférences**.

Voir le chapitre suivant qui leur est consacré.

Exportation des graphiques

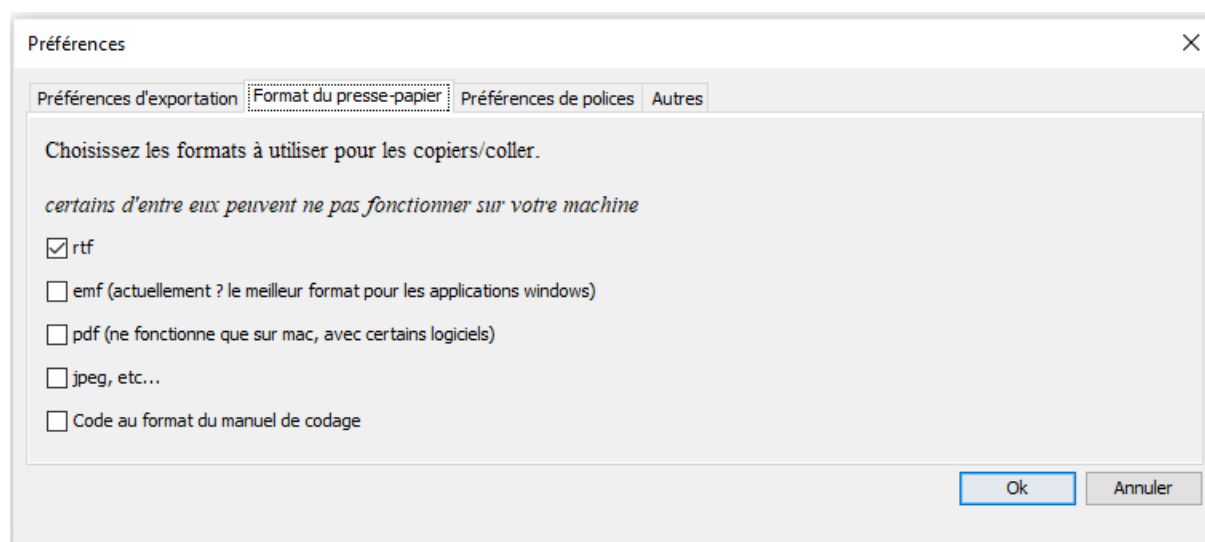
Copier/coller

Souvent, on a besoin de deux tailles de signes dans une application : une pour le texte principal et une plus petite pour les notes de bas de page. JSesh permet de conserver deux configurations pour le copier-coller. La taille actuelle peut être sélectionnée dans le menu **Édition** : choisissez simplement *copie en petite taille*, *copie en grande taille* ou *copie tel qu'à l'écran* (cette dernière doit être utilisée si vous voulez que le glyphe copié garde la même disposition que ceux d'origine).

Choix du format

Un copier-coller fait intervenir deux logiciels : JSesh et votre traitement de texte (Word, OpenOffice, etc.) Désormais, JSesh proposera un certain nombre de formats différents à votre traitement de texte, lui permettant de choisir celui qu'il préfère. Pour contrôler ce qui sera proposé par JSesh, vous pouvez utiliser la boîte de dialogue **Choix du format** (dans les préférences).

Je fais la plupart de mes tests avec *OpenOffice* et *LibreOffice*. Ils ne sont pas parfaits mais ils ont tendance à traiter correctement les images et l'unicode, c'est pourquoi j'encourage fortement leur utilisation. Leurs fichiers ont tendance à être très portables d'un ordinateur à l'autre. Pas de problème de différentes versions du traitement de texte détectés.



- RTF : probablement le choix le plus polyvalent pour l'instant. Fonctionne (presque) partout et donne des résultats raisonnables, en particulier pour les hiéroglyphes mélangés avec du texte alphabétique. Autrefois problématique avec certaines versions de Word sur Mac, mais s'est récemment (en 2016) améliorée.
- PDF : actuellement le choix le plus précis. Pour autant que je sache,

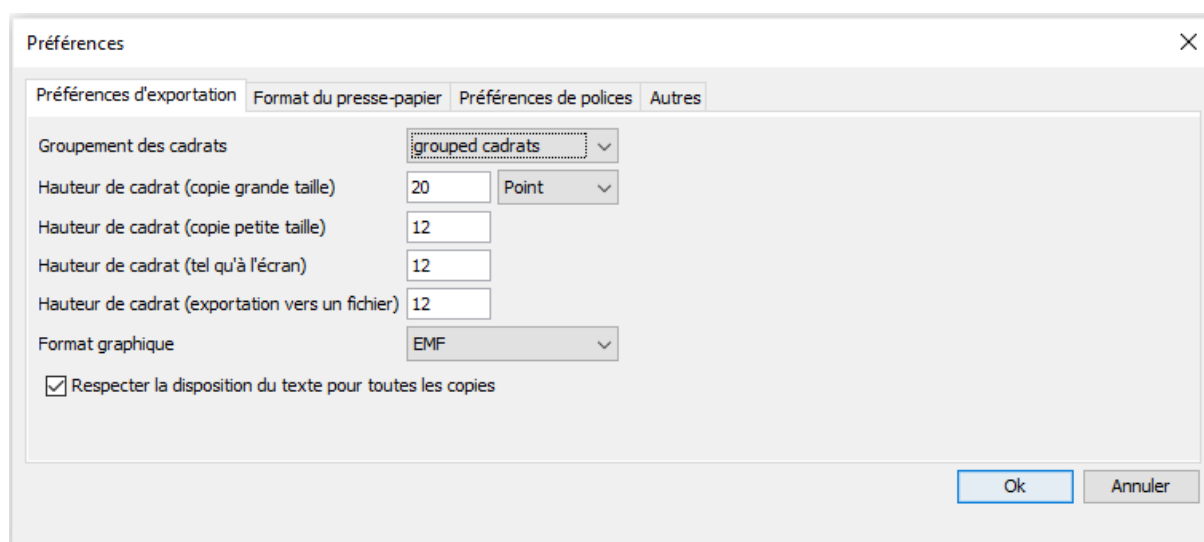
seul Mac OS X prend en charge le copier-coller de PDF, et peu de logiciels le comprennent. Par exemple, Word 2004 ne le fait pas. Word 2008 le fait, cependant. Textedit a la mauvaise idée d'accepter le copier-coller du PDF, mais de transformer le résultat en un bitmap.

- bitmap : donne un bitmap (actuellement) basse résolution, adapté aux pages web par exemple.
- texte brut : copiera l'encodage selon le Manuel de Codage dans le presse-papiers.

Pour des informations spécifiques à votre traitement de texte, consultez le chapitre *Problème des traitement de texte*.

Réglages fins

Les paramètres pour *grande taille/petite taille/tel qu'à l'écran* peuvent être modifiés dans les préférences.



Les *Préférences d'exportation* vous permettent de choisir comment le texte hiéroglyphique sera collé dans les exportations RTF. Il y a trois options :

- *comme une grande image* : L'ensemble du texte sélectionné sera collé dans votre traitement de texte comme une seule image. C'est parfait pour le texte en colonnes, par exemple, ou si vous souhaitez conserver la disposition graphique. Il semble également que la plupart des logiciels de traitement de texte préfèrent gérer de grandes images plutôt que de nombreuses petites.
- *cadrats groupés* : Les cadrats hiéroglyphiques adjacents seront regroupés dans une seule image. Le texte résultant alternera texte normal et grandes images.
- *une image par cadrat* : Le texte collé contiendra du texte normal et des images pour le texte hiéroglyphique. Mais alors, chaque cadrat sera rendu comme une seule image. C'est très bien pour la coupe de ligne et peut être intéressant si vous mélangez du texte et des hiéroglyphes.

Hauteur du cadrat modifie la taille du texte hiéroglyphique collé.

Format graphique permet de sélectionner le type de format pour l'exportation du graphique.

Limitations du copier-coller

En raison de problèmes techniques, nous avons limité le copier-coller. Il n'est pas possible de coller des textes très longs dans un traitement de texte (vous pouvez plutôt les *exporter au format RTF*). Nous allons essayer de l'améliorer, mais nous pensons que c'est une nuisance mineure, car on colle généralement de petites parties de textes hiéroglyphiques plutôt que l'ensemble d'un grand document (la limite est actuellement de 1000 cadrats).

N'hésitez pas à écrire à l'auteur (serge.rosmorduc@qenherkhopeshef.org) à ce sujet (ou sur d'autres problèmes avec JSesh).

Menu "Copier" avancé

Normalement, dans JSesh, vous choisissez le format que vous souhaitez copier/coller dans le menu des préférences. Mais dans certains cas, vous souhaitez peut-être copier votre texte dans un format spécifique qui n'est pas celui que vous avez choisi. Au lieu de revenir aux préférences, vous pouvez utiliser le menu **Copier sous**. Il vous permet de copier le texte sélectionné au format PDF, RTF ou Bitmap.

Possibilité de coller le texte hiéroglyphique dans JSesh

Principes et problèmes

(Je suis désolé, cette partie est un peu technique. Si quelqu'un est capable d'écrire les mêmes explications de manière plus claire, ce serait une contribution bienvenue pour le site de JSesh).

Sur Windows (et sur Mac OS antérieur à Mac OS X), il existe une fonctionnalité appelée *Object Linking and Embedding* (OLE), qui est plus ou moins un copier-coller intelligent. Vous copiez/collez un document de votre éditeur hiéroglyphique dans Word, puis si vous double-cliquez sur l'image collée, cela ouvrira l'éditeur hiéroglyphique, afin que vous puissiez modifier le texte.

Ce n'est pas possible avec JSesh, car ce type d'action est très spécifique au système (si vous en avez vraiment besoin, je crois savoir qu'il est assez bien pris en charge par *Inscribe*). Cependant, à partir de la version 2.11, JSesh fournit une version simplifiée.

Fondamentalement, l'astuce (également utilisée dans d'autres éditeurs de

hiéroglyphes, comme MacScribe) est d'utiliser le champ de commentaire disponible dans certains formats d'image (comme PDF ou EMF), et de mettre le texte du manuel de codage dans ce champ de commentaire. Lorsque l'image est collée dans JSesh, on peut simplement extraire le code. Bien sûr, c'est plus ou moins automatique. Mais j'avais besoin de l'expliquer, car cela dépend d'un certain nombre de facteurs :

- le format de l'image doit prendre en charge les commentaires, et je dois écrire le code correspondant. Ceci est actuellement fait uniquement pour EMF et PDF.
- le traitement de texte doit accepter l'image collée *telle quelle*, et ne pas la modifier. Les modifications suppriment généralement le commentaire. Par exemple, sur un Mac, vous pouvez coller une image PDF dans MS Word. Mais Word transformera (je suppose) l'image dans son propre format et perdra le commentaire au cours du processus.

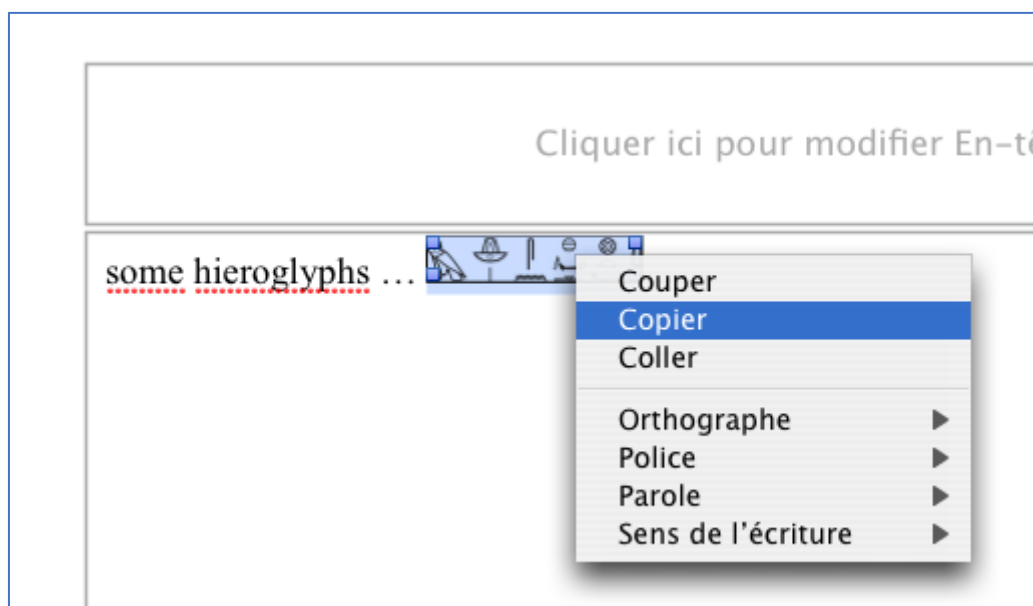
Toutes ces explications pour vous dire que ce système ne fonctionnera pas avec tous les traitements de texte... et surtout pas avec MS Word (en tout cas cela ne fonctionnait pas avec Word 2008 sur Mac, désolé).

Maintenant, voyons les solutions possibles.

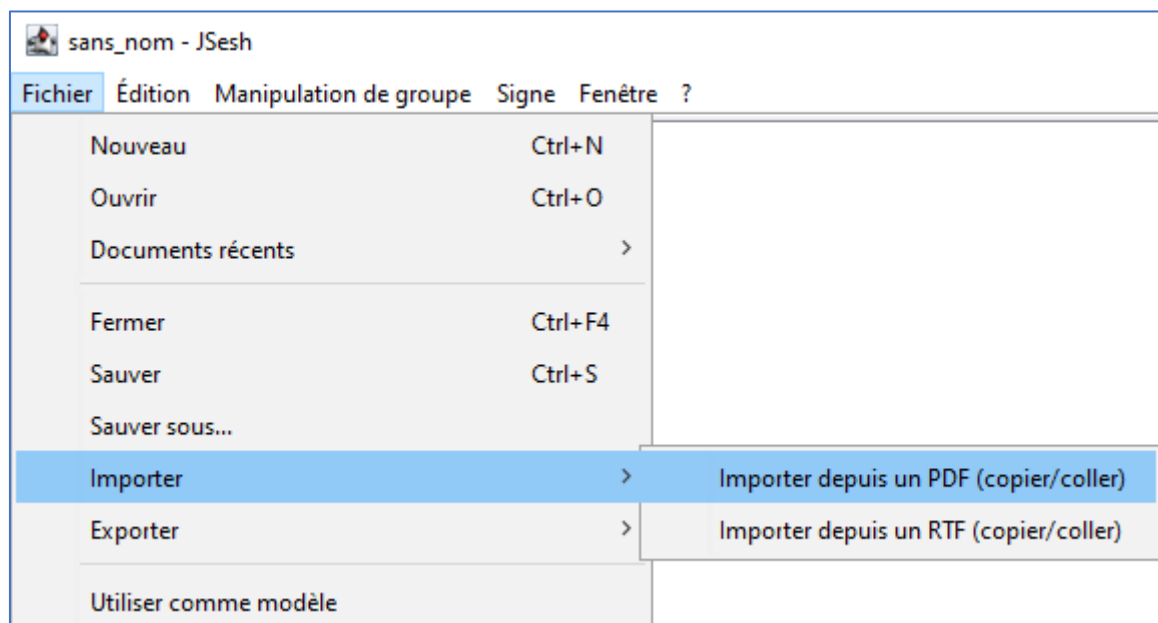
Les éditeurs Mellel, Nissus et le PDF

Si vous utilisez Mellel ou Nissus Writer (et si vous configurez votre option *Copier-coller* pour utiliser le format PDF), vous pourrez coller vos hiéroglyphes dans JSesh.

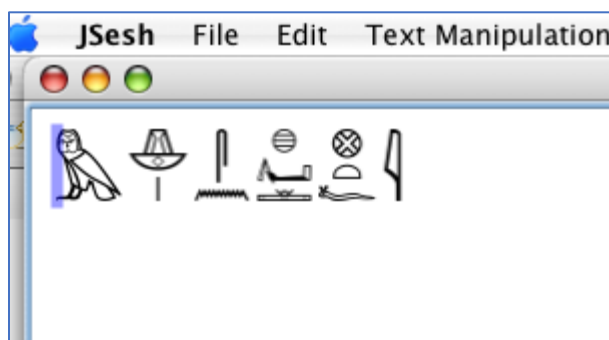
Sélectionnez simplement les hiéroglyphes dans votre traitement de texte (par exemple Mellel) et copiez-les :



Ensuite, allez dans JSesh et sélectionnez **Fichier/Importer/Importer depuis un PDF (copier/coller)** :



Le résultat remplacera votre document JSesh actuel :



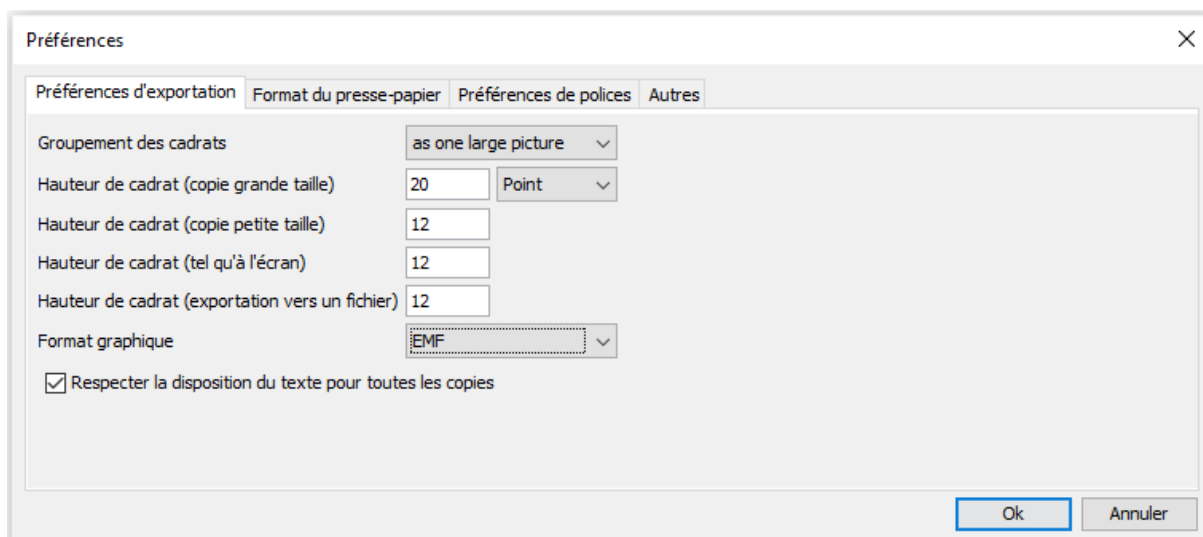
NeoOffice (OpenOffice) et texte RTF

Ce chapitre décrit les manipulations effectuées sur un Macintosh en utilisant NeoOffice comme traitement de texte.

Il fonctionne également avec OpenOffice sur Windows et Linux (mais il semble échouer avec OpenOffice 3.1.1 sur Mac).

Si vous configurez votre option couper-coller pour utiliser le format RTF et si vous avez choisi EMF comme format d'image dans JSesh (voir ci-dessus), vous pourrez recoller vos hiéroglyphes dans JSesh.

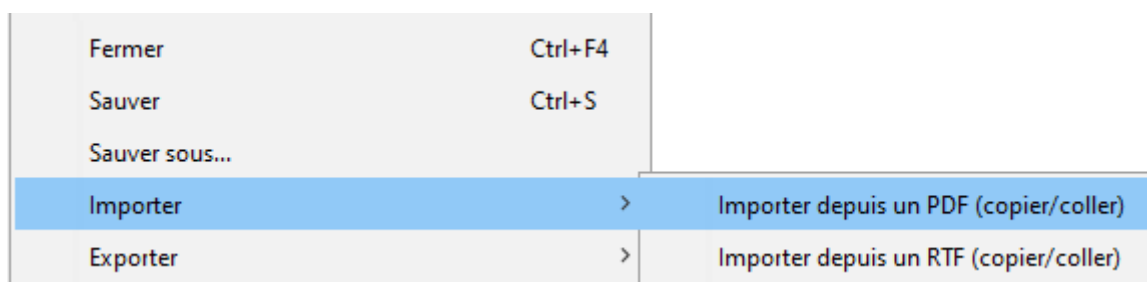
Tout d'abord, vous devez sélectionner "EMF" comme format graphique à utiliser avec JSesh.



Le texte collé en EMF dans NeoOffice :



peut ensuite être recollé dans JSesh :



Voilà :



Maintenant, il y a un petit problème. Nous voulons que NeoOffice génère du texte RTF (avec l'image intégrée). Cela ne se produira pas si vous sélectionnez directement l'image :



Notez que dans le cas d'une *bonne* sélection, l'image est encadrée en noir.

La sélection directe d'une image entraînera un message d'erreur (inoffensif) :



La bonne façon de sélectionner l'image est de cliquer devant elle dans le texte et de la faire glisser avec la souris. Alternativement (et avec moins de dextérité), vous pouvez sélectionner l'image en déplaçant le curseur de texte devant elle, puis en appuyant sur "shift" et l'une des flèches du clavier (c'est beaucoup plus simple de le faire que de le décrire).

Notez que vous pouvez sélectionner plusieurs images :



Et vous obtiendrez une ligne par image dans JSesh :



Veuillez noter que cela ne fonctionne que si vos images ont été collées à partir de la version 2.11 ou plus de JSesh, et dans le bon format.

Autres

Je suis très intéressé de savoir si d'autres configurations fonctionnent.

Impression

Actuellement, l'impression ne fonctionne pas et n'est pas une priorité. Utilisez plutôt l'exportation PDF ou SVG et imprimez à partir d'Acrobat Reader ou d'Inkscape. Merci.

Export en PDF

Pour exporter votre texte au format PDF, choisissez **Exporter comme PDF**. Explorez les différents rendus pour de meilleurs résultats. Il peut y avoir actuellement des problèmes avec les polices latines et de translittération.

Export en JPEG ou PNG

Les images sur les ordinateurs se présentent sous deux formes : *images bitmap* et *images vectorielles*. Pour avoir une idée rapide des différences, disons que les fichiers bitmap sont généralement édités avec des logiciels comme *Photoshop* (ou *Gimp* si vous utilisez un logiciel gratuit), et les images vectorielles sont éditées à l'aide de logiciels comme *Adobe Illustrator* (ou *Inkscape*). La plupart des logiciels peuvent afficher et lire des fichiers bitmap, ces fichiers ont donc tendance à être plus faciles à partager.

Leur principal problème est que vous ne pouvez pas bien zoomer dessus. Un fichier bitmap adapté à l'affichage à l'écran s'imprimera probablement mal. Cependant, le partage d'images vectorielles est assez difficile (sans raison valable, mais c'est une triste constatation), vous aurez donc intérêt à utiliser plutôt des bitmaps haute résolution pour l'impression.

Pour exporter un fichier bitmap (JPEG ou PNG), choisissez l'entrée de menu **Fichier/Exporter/Exporter comme JPEG ou PNG**.

Vous pouvez donc exporter une version JPEG ou PNG⁷ de votre texte. Si aucun texte n'est sélectionné, le texte entier sera rendu. S'il y a une sélection, seul le texte sélectionné sera dessiné.

Export en RTF

RTF (Rich Text Format) est un format de document portable qui peut être lu par la plupart des traitements de texte (par exemple, MS Word et OpenOffice).

Pour exporter un texte en RTF, choisissez l'entrée de menu **Fichier/Exporter/Exporter comme RTF**.

Problèmes spécifiques aux traitements de texte

Hélas, hélas, hélas, le copier/coller et l'inclusion graphique sont bien plus complexes qu'ils ne devraient l'être, du fait d'un manque généralisé de coopération entre les principaux acteurs du domaine.

Ainsi, nous discutons ici des problèmes liés aux traitements de texte spécifiques (et similaires).

Choix de format pour les applications sur Macintosh

À ce jour (21 avril 2016)

⁷ Sachez que les fichiers PNG ont une meilleure résolution que les fichiers JPEG car leur taux de compression est inférieur, ce qui augmente leur qualité, notamment dans le rendu de la couleur rouge. Un fichier PNG gère aussi la transparence, ce qu'un JPEG ne sait pas faire (Note du traducteur).

- Word 2011 pour Mac a commencé (enfin) à accepter le copier/coller de textes RTF contenant des images (la manière "standard" pour JSesh d'effectuer un copier/coller), et il accepte maintenant le format EMF. Le PDF n'est plus la "meilleure" solution car les images PDF ont tendance à être mal recadrées.
- Sur Word 2008, les formats utilisables sont RTF et PDF, PDF donnant les meilleurs résultats. Il semble que vous puissiez utiliser des images "EMF" dans l'export RTF. EMF donne de bien meilleurs résultats que MacPict.
- Sur Word 2004, seul le RTF est raisonnable. Il doit être configuré pour utiliser des images MacPict
- Sur Nissus et Mellel, on peut utiliser soit le PDF soit le RTF. Comme le PDF conserve ses commentaires, il est possible de copier-coller vers et depuis JSesh. Pour RTF, vous devez utiliser MacPict comme format de sortie.
- Sur OpenOffice/NeoOffice, le PDF ne fonctionne pas encore, il faut donc utiliser RTF. Le format d'image suggéré est EMF (avec de meilleurs résultats graphiques).
- pour les logiciels non orientés texte (comme PowerPoint ou Keynotes) : utilisez PDF.

Choix de format pour les applications Windows

- pour Word ou OpenOffice Writer : utilisez RTF, avec EMF comme format graphique. Vous pouvez également utiliser EMF directement.
- pour les logiciels non orientés texte (comme PowerPoint) : utilisez EMF.

Choix de format pour les applications Linux

Pour OpenOffice et al. : utiliser RTF, avec EMF comme format graphique.


Trucs et astuces sur l'encodage de textes hiéroglyphiques

Avant de décrire le manuel de codage lui-même, j'aimerais faire une ou deux remarques sur la façon dont on devrait encoder un texte hiéroglyphique.

Typiquement, vous avez une source, qui peut être a) une source imprimée, avec des hiéroglyphes composés, b) une photographie ou un fac-similé fidèle du texte original, ou c) un hiéroglyphe manuscrit d'une publication égyptologique (par exemple, l'Urkunden⁸).


Vous devez décider à quel point vous devez être fidèle à l'original, et ce n'est pas une question simple. Le premier point est que vous pouvez être sûr que votre texte de Manuel de Codage ne peut pas être un fac-similé, donc, d'une manière ou d'une autre, il trahira l'original. Si le texte est finalement un texte hiératique, vous créez déjà quelque chose de complètement différent de toute façon.

Maintenant, lorsque vous avez un signe, devez-vous rechercher la variante la plus précise ou en utiliser une en quelque sorte standardisée ? Pour y répondre, il faut se demander si cette variante est pertinente, dans le texte, et dans l'usage que l'on entend en faire. Par exemple, dans les textes Kanaïs de Sethi Ier, le pronom à la première personne utilise de nombreuses

variantes de A40 () . Si vous souhaitez que le texte soit destiné à une étude grammaticale, ce n'est pas très pertinent. Si vous vous demandez si c'est un jeu libre d'ajouter de la diversité au texte, alors vous pouvez prendre le temps d'encoder les variantes. En règle générale, *si vous êtes débutant en égyptien*, je vous conseille d'encoder le signe *standard*, car il vous obligera à *lire le texte, pas à copier les signes* . Le problème que vous rencontrez quand vous voulez être très précis, c'est que vous trouverez de nombreux cas de variantes qui n'ont de toute façon aucun encodage, et vous prenez le risque de tromper vos lecteurs en donnant une fausse impression de précision.

Lorsque votre source est une version manuscrite d'un texte, comme dans les Urkunden, méfiez-vous d'un point : à moins qu'un signe ne soit très particulier, un égyptologue dessinera généralement ses glyphes de la



manière la plus simple. Cela signifie par exemple qu'il utilisera V23A ()

au lieu de  (V22) qui est le signe hiéroglyphique normal. Vous devez comprendre cela, et lorsqu'une sélection de signe est simplement causée par la main individuelle de l'auteur, rendez-la avec le signe *standard*. En revanche, lorsque l'auteur a dessiné une forme de signe très spécifique, il





⁸ Voir par exemple <http://www.egyptologyforum.org/EEFUrk.html>

peut être intéressant de lui accorder plus d'attention.

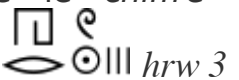
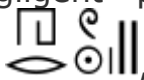
Encodage de textes hiératiques

La règle ci-dessus est encore plus forte pour les textes hiératiques. Il ne sert à rien de faire une distinction entre  et , car les deux sont rendus de la même manière en hiératique !

Or, il y a un certain nombre de particularités du hiératique qui méritent plus d'attention. Ceux-ci ont été identifiés par sir Alan Gardiner dans son article *The Transcription of New Kingdom Hieratic*, JEA 15 (1929), p. 48-55. Le rendu des textes hiératiques doit faire allusion à la façon dont les signes originaux sont placés, afin de permettre au lecteur de comprendre la raison du rendu et de revenir à l'original si nécessaire. JSesh inclut un certain nombre de signes utiles pour le rendu des textes hiératiques :

Signe	Code	Utilisation
	Ff1	un signe particulier aux textes ramessides, qui sert en quelque sorte de « joker ». Il est différent du signe Z5 (). Vous pouvez l'obtenir dans JSesh en tapant “,” (virgule) puis la touche espace.
	Ff100	un signe non standard, “dot space fill”. Utilisez-le pour rendre les différents points qu'un scribe peut utiliser, lorsqu'ils ne sont pas une ponctuation.
	Ff101	un panneau non standard, remplisseur d'espace horizontal. Utilisez-le pour rendre divers signes horizontaux dénués de sens que le scribe peut utiliser pour combler les vides.

JSesh inclut également des signes spécifiques pour les nombres dans les textes hiératiques. Dans ces textes, le *déterminatif* Z1 est souvent plus petit que le *chiffre* Z1. Un rendu négligent peut entraîner la lecture



de  *hrw 3* alors que l'original a , qui est clairement *hrw 2*.

Disposition des signes

Utilisez les différentes capacités de *ligature* de JSesh, et si tout le reste échoue, optez pour l'*éditeur de groupe*.

Quelques points intéressants :

Dans le groupe “M17-M17”, les signes doivent souvent être plus proches que ne le permet l'espacement standard de JSesh (en fait, cela est déjà

indiqué par sir A. Gardiner dans le catalogue de ses polices). Si l'on écrit i*i:k (c'est-à-dire deux yod groupés horizontalement et un "k" en dessous), le groupe créé par JSesh est . Pour obtenir la meilleure mise en page :  est simple : il suffit de ligaturer les deux yods (tapez "i" "&" "i", par exemple).

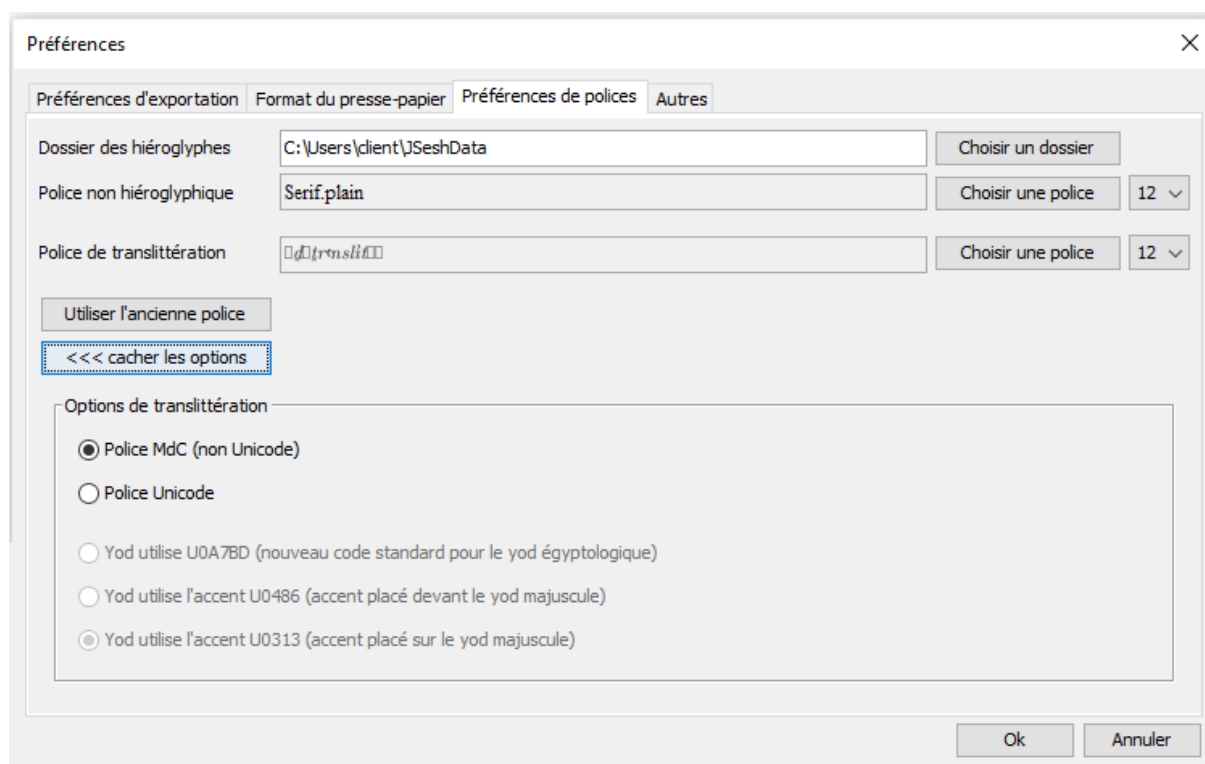
Translittération

L'édition de la translittération devrait être une affaire simple. Après tout, elle n'utilise que quelques signes alphabétiques. Hélas, ce n'est pas le cas. Dans le MdC, une norme simple basée sur l'ASCII a été utilisée. Mais cette norme nécessite des polices spéciales, où "a" n'est pas "a" mais "ayin". Cela a conduit à de nombreux problèmes lors de l'échange de textes. Passer du Mac au PC était un vrai problème, et les éditeurs étaient très mécontents. Des centaines d'heures de travail inutile ont été consacrées à la conversion des fichiers - ou à leur ressaisie. Normalement, Unicode devrait résoudre tous ces problèmes. Ainsi, actuellement, JSesh propose des choix à l'utilisateur. Votre choix personnel dépendra de ce que vous voulez faire avec vos textes. Il est fort probable que l'unicode soit nécessaire pour la plupart des publications, par exemple. Vous pouvez choisir la manière dont la translittération sera traitée dans le menu **Édition > Préférences**.

Utiliser une police unicode

Pour utiliser une police unicode, vous devez disposer d'une police avec les bons signes. Actuellement (2016⁹), les polices gratuites possibles sont :

- [EgyptoSerif](#), ma propre police de translittération. Les signes ne sont pas très beaux, mais ils sont au bon endroit.
- Gentium Plus
- Nouvel Athena Unicode (voir les détails exacts)



Menu de gestion des polices

⁹ Voir ici https://fr.wikipedia.org/wiki/Liste_de_polices_d'écriture#Polices_Unicode pour une liste plus ou moins d'actualité.

1. Vous devez ouvrir le menu "Préférences" (sur Mac, c'est dans le menu "JSesh" ; ailleurs, c'est sous le "Menu Edition").
2. Allez dans l'onglet « Préférences de polices ».
3. Sélectionnez une police pour la translittération
4. Ouvrez le panneau "Afficher les options" et cochez "Police de translittération Unicode".

Problèmes actuels avec les polices Unicode

Le support actuel dans le système d'exploitation et les logiciels pour les polices de translittération n'est pas bon. Même Java sur Mac - et donc JSesh - a des problèmes.

Les vrais problèmes sont H_̇ et yod

Majuscule H_̇

Sur certains systèmes, H_̇ (comme dans le dieu-bélier *Hnmw*) s'affiche mal. Cela dépend de la police et du logiciel et du système. Dans la plupart des cas, ça va. Une exception ennuyeuse est qu'il ne donne pas un bon résultat avec JSesh sur Mac OS X. Apparemment, le système de rendu donne de meilleurs résultats sous Windows. Notez que si vous collez votre texte dans un traitement de texte, vous obtiendrez probablement un bon résultat de toute façon.

Yod

Unicode vous offre un certain nombre de choix. J'ai sélectionné les deux meilleures solutions (la troisième, utilisant une sorte de demi-cercle de l'accent yod, reproduit simplement une astuce utilisée lorsque les gens avaient des machines à écrire et fabriquaient à la main leurs yods avec des lettres "c"). Le problème est que, techniquement, le support pour ces solutions fait défaut.

Le yod est censé être codé avec un "i" (un *i pointillé*, pas un sans point) et un accent. La police doit contenir suffisamment d'informations pour afficher correctement ces signes. Le problème est que beaucoup de logiciels ne regardent pas le contenu de la police et fonctionnent tout seuls. Le résultat est que le point n'est pas supprimé sur de nombreux systèmes (y compris JSesh sur Mac). Le positionnement de l'accent devant le « I » majuscule est un autre problème.

Avec une configuration correcte, il est possible d'obtenir de bons résultats avec OpenOffice et Word.

Les deux possibilités pour yod sont :

1. U+0313 : Comme i + "virgule au dessus" est déjà utilisé dans d'autres langues, ses chances d'être correctement supporté sont meilleures. Le seul problème est que dans l'affichage normal de cette combinaison pour le "I" majuscule, l'accent est censé être au-

dessus du I. Comme vous le savez, l'affichage correct du yod majuscule a l'accent devant le "I". Si vous pouvez vivre avec cela, cette solution est la plus sûre. Il donne de bons résultats dans la plupart des cas (mais pas avec Java et Mac OS X, donc l'affichage JSesh sur Mac n'est pas correct).

2. U+0486 : Cet accent a la particularité intéressante de n'être utilisé pour rien d'autre sur les lettres latines. Ainsi, une police est libre de la placer devant le "I" majuscule. En théorie, c'est la bonne solution. En pratique, cela fonctionne avec certains logiciels, mais pas tous. OpenOffice donne de bons résultats, ainsi que le logiciel tout simple TextEdit sur Mac. J'ai pu réussir avec Word sur Mac en activant toutes les ligatures.

JSesh ne prend pas en charge le système IFAO/Unicode actuel. Cette norme était un patch temporaire utilisant des signes Unicode disponibles, effectué lorsqu'aucun encodage n'était disponible pour la translittération égyptienne. Maintenant que "Aleph" et "Ayin" ont du code, j'encourage (enfin, j'impose) 😊 leur utilisation.

Références

* [Article Wikipédia sur la translittération et l'unicode](#)

Ne pas utiliser unicode

Vous pouvez sélectionner une police avec un encodage compatible MdC.

Dans ce cas, JSesh ne traite pas les majuscules.

Vous pouvez choisir votre propre police, si vous en avez une, ou utiliser la police de translittération JSesh par défaut.

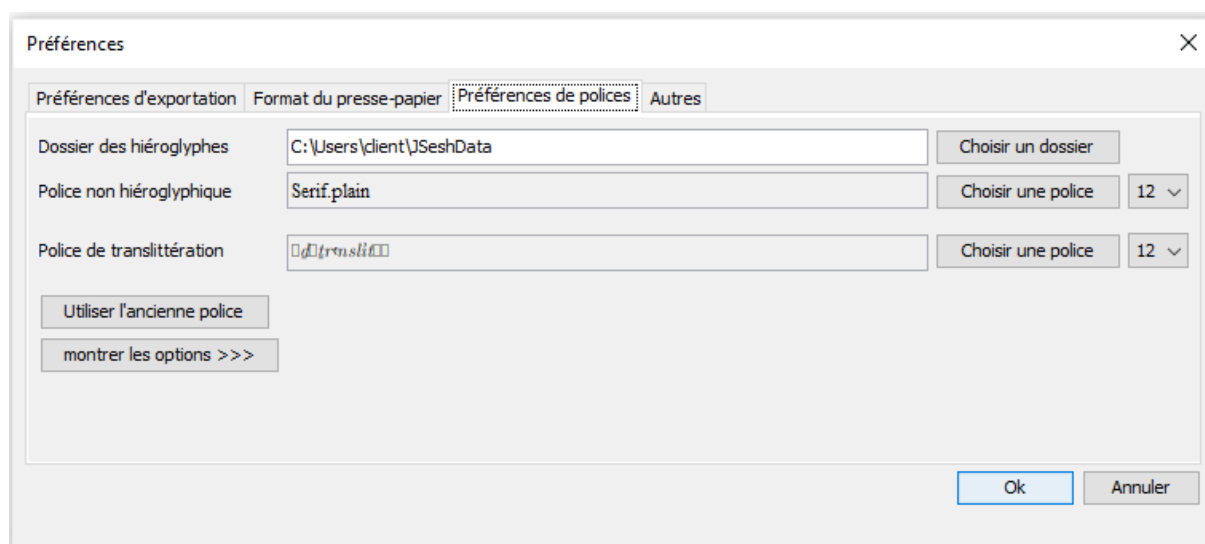
Une version de cette police est embarquée dans le logiciel, mais vous pouvez l'installer sur votre système (à utiliser avec un traitement de texte par exemple).

Une copie de cette police est disponible dans le dossier *Polices* de l'installation de JSesh (sur Mac).

Méthode simple avec l'ancienne police

Vous devez ouvrir le menu **Préférences** (sur Mac, c'est dans le menu **JSesh**, ailleurs, c'est sous le menu **Édition**).

Allez dans l'onglet **Préférences de polices** et cliquez sur le bouton *Utiliser l'ancienne police*.



Et c'est tout. Pour vos propres documents Word (ou OpenOffice...), vous pouvez trouver la police dans le dossier de l'application JSesh (dans le dossier *Polices* sur Mac).

Utilisation de vos propres polices compatibles MdC

Si vous avez une police compatible avec le MdC (c'est-à-dire que le "a" représente un ayn, le "A" représente un aleph, etc...), vous pouvez également l'utiliser.

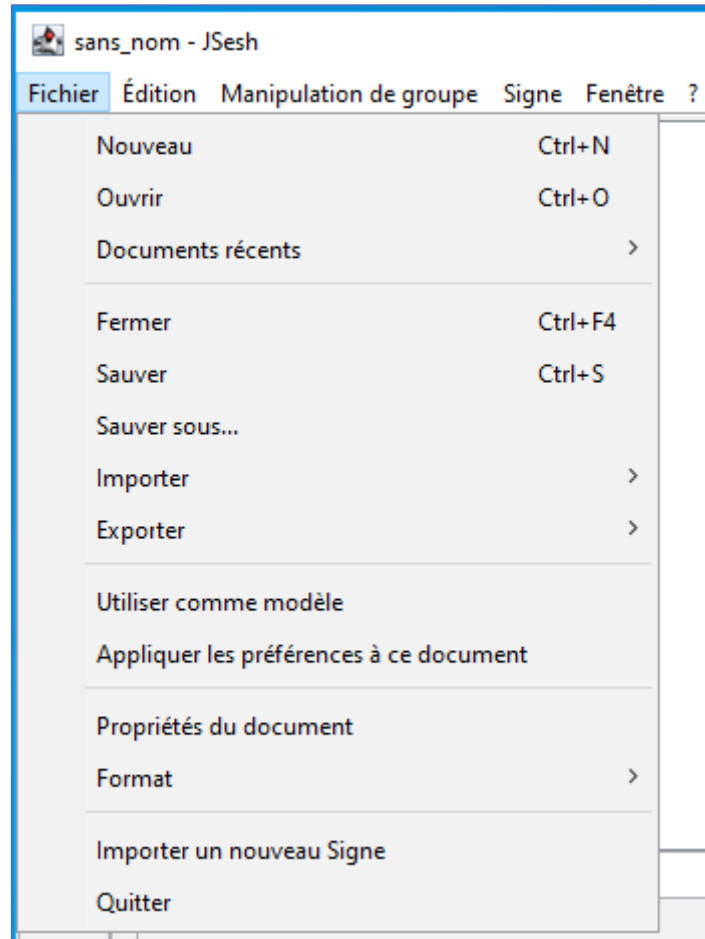
Sélectionnez simplement votre police (en utilisant *Choisir la police* devant la police de translittération), puis, dans les options avancées, sélectionnez *Police de translittération MdC (non Unicode)*.

Explication des menus de JSesh

Version Windows 10 : 7.5.5

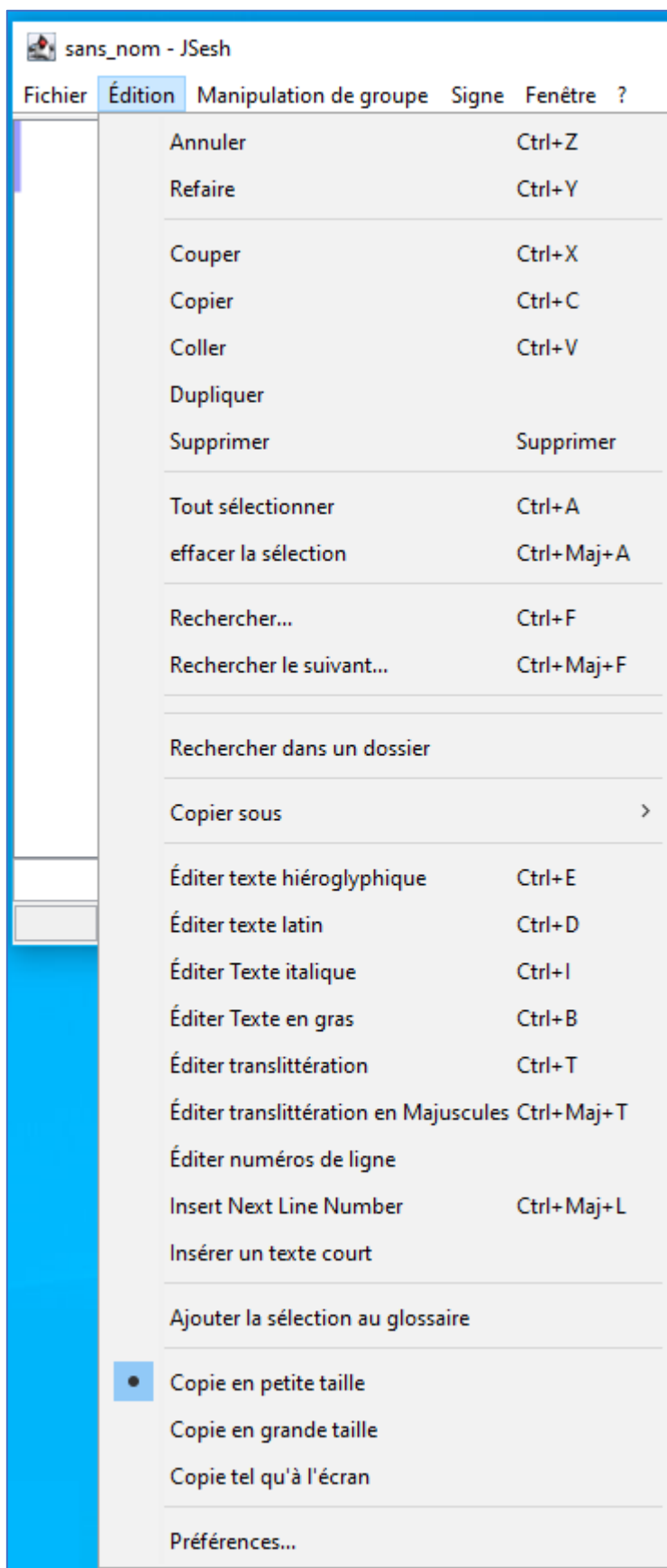
Version Macintosh macOS 12.1 Monterey : 7.5.5

Fichier



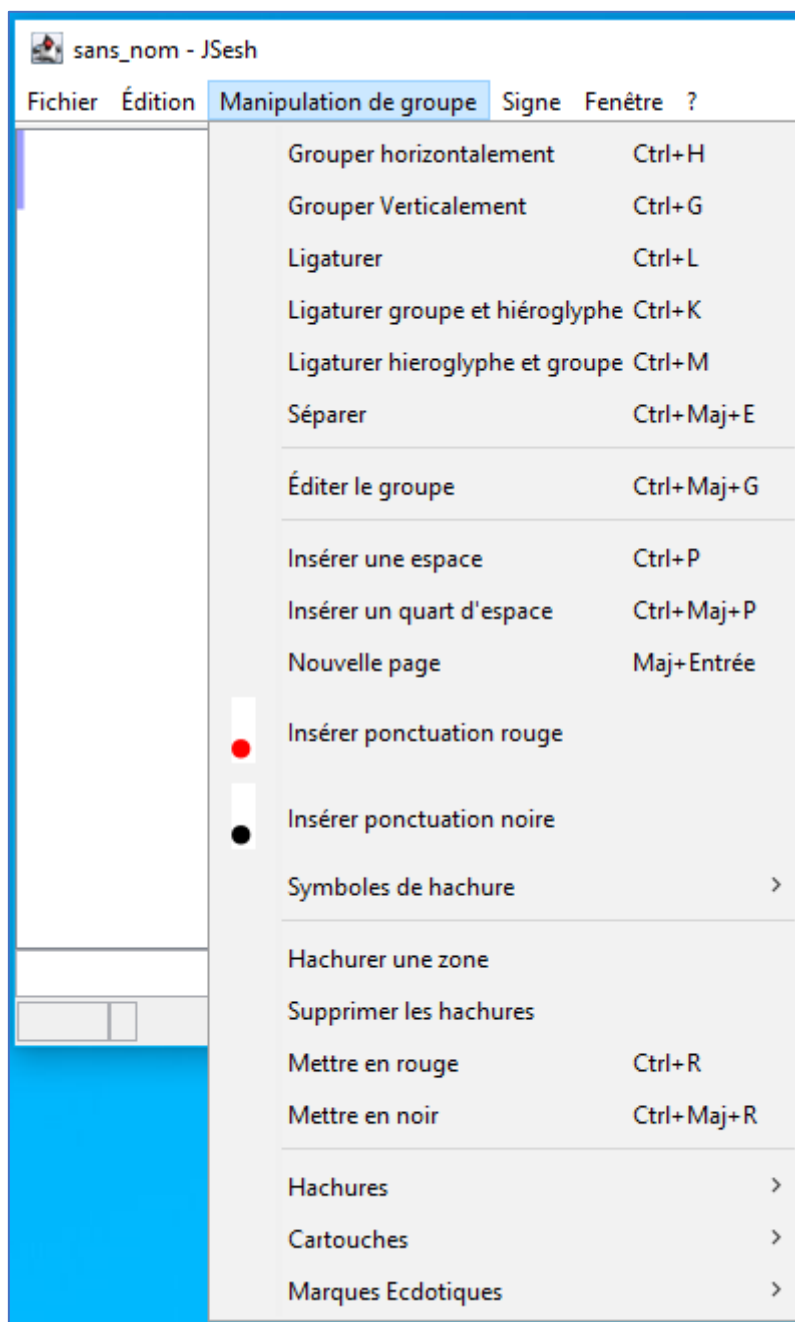
Nouveau	création d'un document (extension .gly)
Ouvrir	ouvre un document existant
Documents récents	liste les derniers documents utilisés (ouverts ou créés)
Fermer	ferme le document en cours d'édition
Sauver	enregistre les modifications apportées à un document
Sauver sous	enregistre les modifications apportées à un document sous un autre nom
Importer	importe un document au format PDF ou RTF par copier/coller
Exporter	exporte le document en cours d'édition selon différents formats : JPEG, PNG, WMF, EMF, MacPict, EPS, SVG, PDF, RTF ou HTML.
Utiliser comme modèle	
Appliquer les préférences à ce document	lorsque JSesh ouvre un document .gly, les premières lignes du fichier contiennent les préférences qui ont été sélectionnées pendant son édition. Si l'on change ces préférences avec JSesh, celles-ci sont mises à jour dans le fichier .gly au moment de son enregistrement, mais l'on peut aussi ouvrir un nouveau document dans une deuxième session d'édition et choisir d'appliquer les préférences de la première session en utilisant l'option « Appliquer les préférences à ce document ».
Propriétés du document	permet de modifier les préférences de dessin
Format	permet de changer le format d'affichage des textes, en lignes, en colonnes, de gauche à droite, de droite à gauche, etc.
Importer un nouveau signe	permet d'importer dans la session d'édition un nouveau signe, au format TTF, bzs ou SVG.
Quitter	termine la session d'édition.

Édition



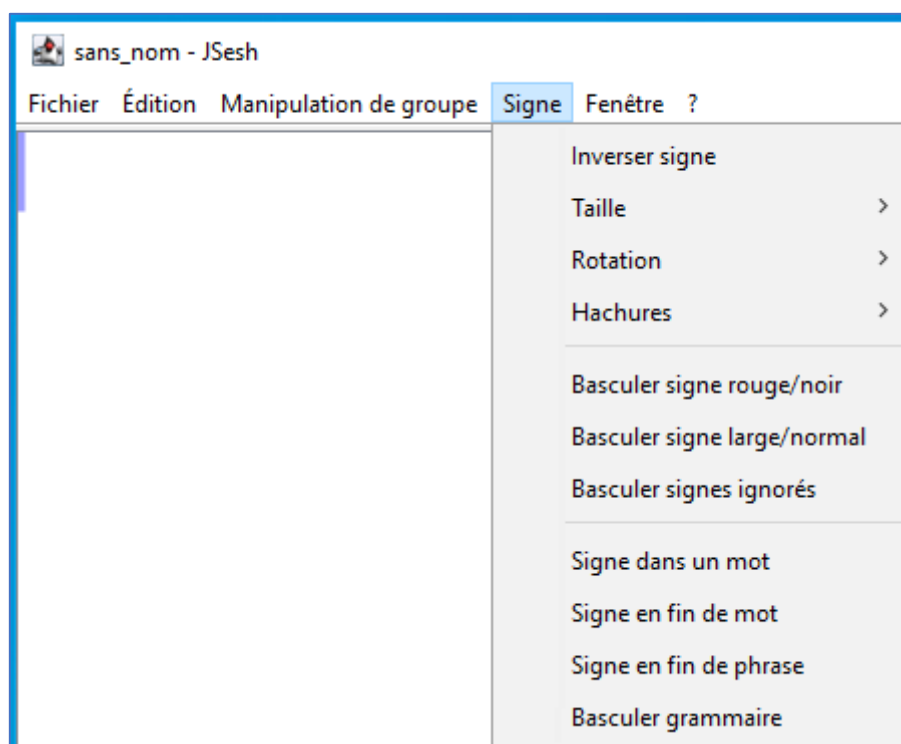
Annuler	annule la dernière action
Refaire	recommence une action annulée
Couper	coupe l'objet sélectionné
Copier	copie l'objet sélectionné
Coller	colle l'objet sélectionné
Dupliquer	duplique l'objet sélectionné
Supprimer	efface l'objet sélectionné
Tout sélectionner	sélectionne tout le document (pas seulement ce qui est affiché dans la fenêtre d'édition)
Effacer la sélection	annule la sélection (n'efface pas le texte sélectionné)
Rechercher	recherche la sélection en cours dans le texte
Rechercher le suivant	recherche l'occurrence suivante
Rechercher dans un dossier	recherche la sélection dans tous les fichiers .gly d'un dossier spécifié
Copier sous	copie la sélection selon plusieurs formats : PDF, RTF, JPEG, PNG, MdC, Unicode, Unicode 12
Éditer texte hiéroglyphique	
Éditer texte latin	
Éditer texte italique	
Éditer texte en gras	
Éditer translittération	
Éditer translittération en majuscules	
Éditer numéros de lignes	
Insérer nouveau numéro de ligne	permet d'insérer le numéro de la ligne suivante
Insérer un texte court	permet d'insérer un texte à la position du curseur
Ajouter la sélection au glossaire	ajoute la sélection au glossaire (📖)
Copie en petite taille	
Copie en grande taille	
Copie tel qu'à l'écran	
Préférences	affiche le menu des préférences d'édition

Manipulation de groupes



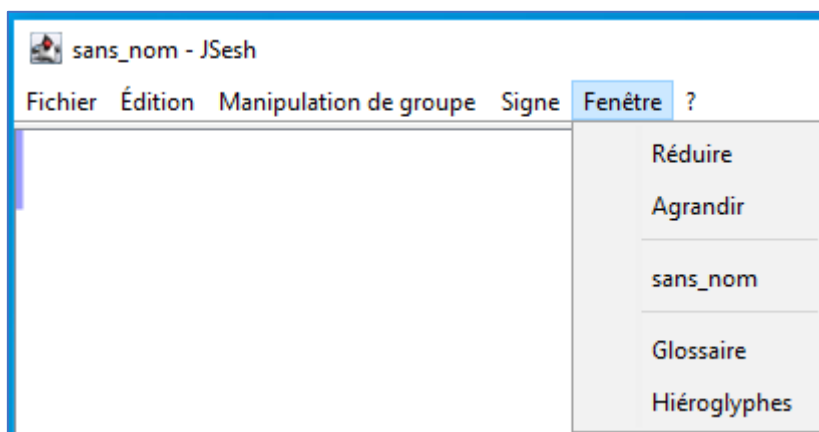
Grouper horizontalement	créé un groupe horizontal de deux signes (les deux signes ne faisant plus qu'un élément sélectionnable)
Grouper verticalement	créé un groupe vertical de deux signes (les deux signes ne faisant plus qu'un élément sélectionnable)
Ligaturer	superpose tous les signes sélectionnés en un seul
Ligaturer groupe et hiéroglyphe	superpose un groupe par-dessus un signe
Ligaturer hiéroglyphe et groupe	superpose un signe par-dessus un groupe
Séparer	supprime un groupe ou une ligature
Éditer le groupe	appelle l'éditeur de groupes
Insérer un espace	insère un espace à la position du curseur
Insérer un quart d'espace	insère un quart d'espace à la position du curseur
Nouvelle page	ajoute un séparateur de page (trait horizontal)
Insérer ponctuation rouge	ajoute un point rouge à la position du curseur
Insérer ponctuation noire	ajoute un point noir à la position du curseur
Symboles de hachure	permet d'ajouter une hachure complète, horizontale, verticale ou un quart de hachure
Hachurer une zone	hachure un ensemble contigu de signes
Supprimer les hachures	supprime les hachures des signes sélectionnés
Mettre en rouge	change en rouge la couleur des signes sélectionnés
Mettre en noir	change en noir la couleur des signes sélectionnés
Hachures	permet de positionner une ou des hachures dans un cadrat (16 positions)
Cartouches	ajoute un cartouche ou serekh autour des signes sélectionnés (23 possibilités)
Marques ecdotiques	ajoute une marque ecdotique à la position du curseur

Signes



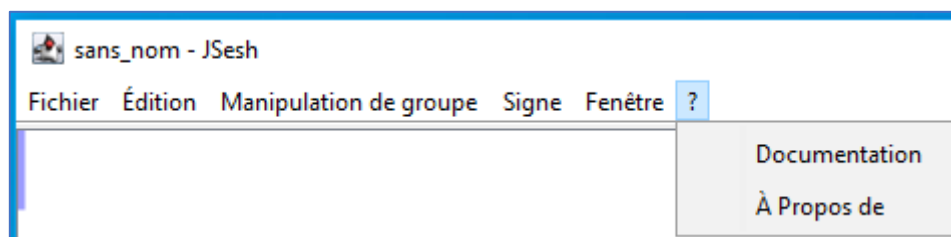
Inverser signe	change le sens de lecture d'un signe
Taille	change la taille d'un signe (en pourcentage)
Rotation	effectue une rotation sur un signe (par incréments)
Hachures	permet de positionner une ou des hachures dans un cadrat (16 positions)
Basculer signe rouge/noir	convertit un signe rouge en noir et vice-versa
Basculer signe large/normal	
Basculer signes ignorés	permet de griser un signe à ignorer
Signe dans un mot	
Signe en fin de mot	
Signe en fin de phrase	
Basculer grammaire	

Fenêtre



Réduire	enlève la fenêtre de JSesh du bureau
Agrandir	ajuste la fenêtre de JSesh à la taille de l'écran
(sans_nom)	affiche la liste des fenêtres d'édition ouvertes (« sans_nom » est le nom par défaut donné lorsqu'on crée une nouvelle session d'édition)
Glossaire	affiche le glossaire
Hiéroglyphes	affiche la palette des hiéroglyphes

Aide



Documentation : permet d'accéder à la présente documentation sur Internet

A propos de : affiche le numéro de version de JSesh et quelques informations complémentaires.

Extension de la liste des signes

Présentation

À partir de la version 2.0beta, les utilisateurs de JSesh peuvent créer leurs propres signes.

Un éditeur de signes et un système de base de données élaboré sont prévus à l'avenir, mais, de façon plus pratique, il a été décidé de permettre l'importation de signes créés avec différents logiciels.

Donc :

a) JSesh n'a actuellement pas d'éditeur de signes mais

b) JSesh peut importer des signes depuis :

- des polices TrueType (voir l'éditeur gratuit FontForge) ;
- des fichiers SVG : SVG est un format relativement récent pour les graphiques vectoriels. Il est très puissant et complet.

Actuellement, JSesh comprend les fichiers SVG si le signe est dessiné en noir sur blanc. Vous pouvez éditer des fichiers SVG avec un certain nombre de programmes ; l'un des meilleurs gratuits est Inkscape, qui a l'avantage d'être multi-plateformes.

Pour la compatibilité avec mon ancien logiciel (tkssh), il peut aussi lire :

- les fichiers de polices exportés depuis tkssh (fichiers .tml)
- les fichiers de polices des utilitaires de polices GNU (fichiers .bzip).

Les deux types de fichiers peuvent être édités avec le logiciel FontEdit, inclus dans tkssh. Mais cela n'a que peu d'intérêt pour l'utilisateur générique.

Importation de nouveaux glyphes

Choisir un dossier pour vos signes

Afin de pouvoir ajouter de nouveaux signes, vous devez d'abord choisir où ils seront stockés sur votre ordinateur. Pour ce faire, sélectionnez simplement **Édition > Préférences**, puis, dans l'onglet **Préférences de polices**, saisissez le dossier que vous souhaitez utiliser dans le champ *Dossier des hiéroglyphes*. Vous devez créer un dossier vide à cet effet spécifique.

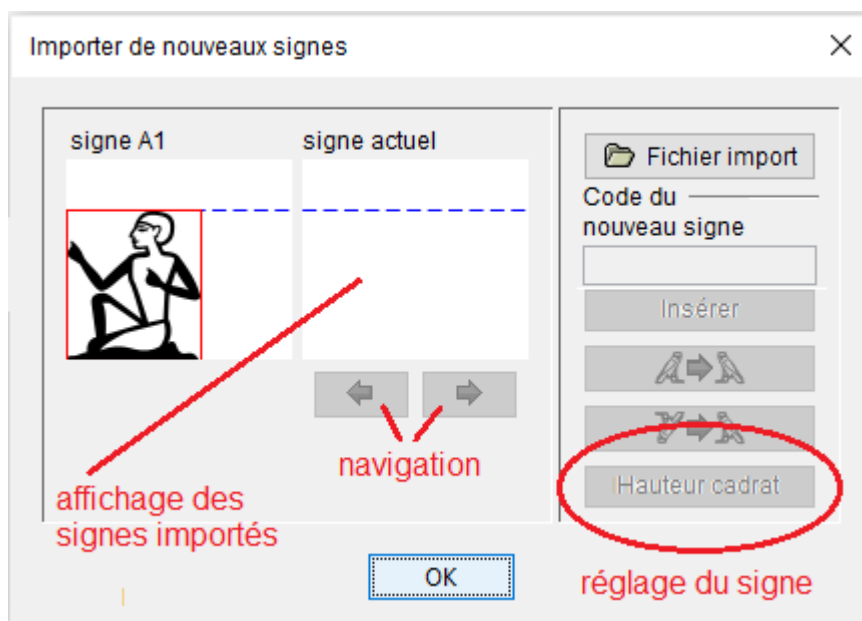


Vous n'avez à le faire qu'une seule fois. Les prochains appels de JSesh utiliseront ce dossier.

Insertion d'un nouveau signe

Pour ajouter de nouveaux signes, vous devez les importer à partir d'un fichier créé avec un autre logiciel, puis attribuer de nouveaux codes aux signes créés. L'interface d'import de signes peut être lancée à partir du menu **Fichier > Importer un nouveau signe**. L'import des signes se fait en deux phases :

- vous importez une image ou un ensemble d'images à partir d'un fichier (dessins SVG, polices TrueType, etc.)
- vous attribuez un code pour chaque signe que vous avez importé, puis vous insérez le nouveau signe dans la liste des signes JSesh.



Importation de dessins

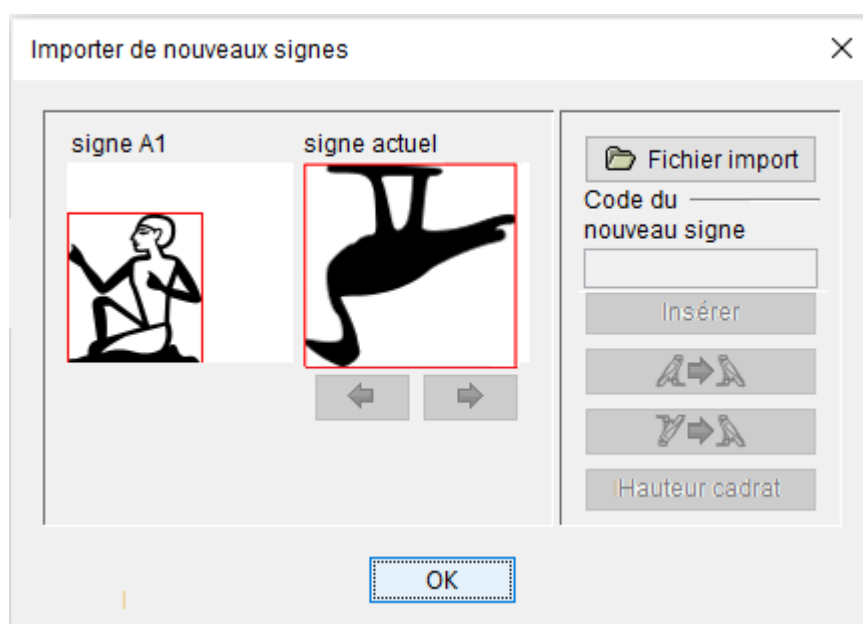
L'importation de dessins est simple. JSesh peut lire :

- des polices TrueType
- des fichiers .tml et bzt, qui peuvent être créés avec le précurseur de JSesh, tksesh.
- des fichiers SVG



Cliquez simplement sur le bouton **Importer le fichier** et sélectionnez le fichier qui contient votre (vos) image(s). Vous pouvez utiliser les boutons de navigation (ceux avec des flèches) pour parcourir les signes disponibles.

Affiner vos signes

Dans certains cas, la taille ou l'orientation de vos signes ne sera pas correcte. Par exemple, dans l'image ci-dessous, le canard est trop gros et son orientation est erronée.



Vous pouvez corriger ceci :

- les boutons  et  peuvent être utilisés pour changer l'orientation du signe ;
- le bouton **Hauteur cadrat** donne au signe la même taille que le signe A1 ;
- si vous cliquez sur la fenêtre du signe et faites glisser la souris, la position de la souris définira la hauteur du signe.

Attacher des codes aux dessins et les insérer dans JSesh

Une fois que votre signe est correct, il est temps de lui donner un nom. Dans un certain nombre de cas, JSesh en aura deviné un, mais cela ne signifie pas que le signe a été enregistré dans sa liste. Rien n'est fait tant

que vous n'avez pas appuyé sur le bouton **Insertion**.

Sauf lorsque vous créez des signes qui sont déjà documentés dans le Manuel de Codage (par exemple, parce qu'ils ne sont pas disponibles dans JSesh), vous devez d'abord obtenir un identifiant d'utilisateur (UID), qui différenciera vos signes des signes créés par d'autres utilisateurs.


Pour obtenir un UID, envoyez un mail à l'auteur serge.rosmorduc AT qenherkhopeshef.org.

Remarque pour les utilisateurs de tksesh : l'UID est simplement votre identifiant tksesh.

Soyez prudent lorsque vous donnez un nom aux signes. Si vous souhaitez que vos fichiers soient lisibles par n'importe qui, soyez donc fidèles au "Manuel de codage". Nos suggestions actuelles sont les suivantes :

- Si vous créez votre propre version d'un signe dans le Manuel (soit parce qu'il n'est pas encore disponible dans JSesh, soit parce que vous n'aimez pas le signe par défaut fourni, utilisez soit le Manuel de codage standard pour le signe, ou utilisez la notation suivante : **USuid + code Gardiner**, où uid est l'identifiant de l'utilisateur.
- Lorsque vous créez un signe que vous considérez comme une variante d'un signe standard, vous devez construire son nom comme ceci : **USuid + Code Gardiner + "VAR" + CODE VARIANT** où : Code Gardiner est le code standard du Manuel de codage pour le signe de base et VARIANT CODE est généralement une lettre majuscule. La forme ramesside habituelle du lièvre a une queue d'animal Seth. Si je veux le distinguer du signe "normal", je peux l'appeler US1E34VARA.
- Lorsque le signe que vous créez correspond à un signe entièrement nouveau, donnez-lui un code de la forme : **USuid + Catégorie Gardiner + NUMERO + "XT" + CODE VARIANT OPTIONNEL**.

Vous êtes libre de choisir le numéro qui vous convient, mais je vous suggère d'essayer de trouver une disposition logique. Il serait également préférable que ce numéro ne corresponde pas à un signe "standard" (pour éviter d'induire en erreur les innocents utilisateurs d'autres logiciels), Vous pouvez donc par exemple commencer votre numérotation à 1000. Notez également que le numéro ne doit pas contenir de zéros non significatifs. La catégorie doit être correcte, et si vous n'en avez vraiment aucune idée, vous devez utiliser la catégorie Aa. "Ff" est réservé aux signes utilisés spécifiquement pour la transcription hiératique.

Exemple : la Harpe Louvre E 116 A vient d'être publiée par C. Barbotin (La voix des hiéroglyphes, p. 66-67), et dans son texte figure le signe inconnu . Je peux créer ce signe et lui donner, par exemple, le code US1Aa1000XT. D'autre part, la stèle d'Israël contient un signe qui est

très probablement un griffon, mais ne correspond à aucun signe de griffon enregistré dans le manuel. Donc je lui ai donné le code US1E162VARA, car E162 est un signe de griffon.

- Utilisation de ce système lors de l'import de textes depuis d'autres logiciels : il est possible que d'autres logiciels fournissent leurs propres codes « non standard ». Si vous souhaitez importer des textes de ces logiciels dans JSesh, vous devrez peut-être donner un nom aux nouveaux signes qu'ils contiennent (notez que les polices sont généralement protégées par les lois sur le droit d'auteur, vous devrez donc redessiner les signes vous-même, soit en adaptant les signes JSesh ou en trouvant une image des signes dans les sources hiéroglyphiques originales). Pour des raisons de compatibilité, les codes utilisateurs suivants peuvent être utilisés pour d'autres logiciels : winglyph 1000, macscribe 1001, inscribe 1002, got 1003, visualglyph 1004. Si vous pensez à d'autres logiciels que je devrais ajouter à cette liste, dites-le moi.
- code standard du manuel de codage. Vous trouverez ces codes dans le dictionnaire de Hannig (1995), par exemple. Un certain nombre de listes sont également disponibles sur le Web. Veuillez noter que l'utilisation des codes est une chose, mais que les signes réels dessinés dans les polices d'autres logiciels sont protégés par la loi sur le copyright. Donc, vous devez venir avec vos propres versions des signes, soit à partir de sources réelles, soit en les recréant. Vous trouverez plus d'informations sur le dessin des signes dans la section suivante. Dans notre exemple, votre canard remplacerait le canard normal, il aurait donc le code "G39".

Nous suggérons que, si vous créez un signe avec un code Gardiner "normal", vous lui donnez également un code de signe utilisateur. De cette façon, vous serez sûr de le conserver même si un signe avec ce code Gardiner est ajouté ultérieurement au logiciel.

Notez que pour la compatibilité avec tksesh, nous prenons également en charge les "codes de glyphes utilisateur" arbitraires. Ces codes correspondent aux codes que tksesh a donnés aux nouveaux signes. Les codes de glyphes utilisateur ont la forme UG id M mid N sid , où id, mid et sid sont des nombres. Évitez d'utiliser ces codes pour le moment.

Lorsque votre signe est prêt, cliquez sur le bouton **Insertion**.

Système alternatif pour l'insertion de signe

Si votre fichier SVG de signes a un nom qui correspond à un code (disons, US1A1VARA.svg) et que le module quadrant est soit 18px ou 1800px, alors vous pouvez simplement mettre le signe dans votre dossier de hiéroglyphes.

Création d'un signe avec Inkscape

Présentation

JSesh n'inclut pas encore d'éditeur de hiéroglyphes. Cependant, il peut inclure des signes dessinés avec un certain nombre d'autres logiciels, comme FontForge et Inkscape. Même si un éditeur de signes est ajouté, cette fonctionnalité sera conservée.

Dans ce tutoriel, je montre comment utiliser le logiciel Inkscape pour créer un nouveau hiéroglyphe.


Dessiner un nouveau signe est une tâche assez longue et fastidieuse. Cela nécessite une compréhension de base des éditeurs graphiques vectoriels (Inkscape, Adobe Illustrator...), ce qui n'est pas anodin, et cela demande aussi beaucoup de travail de toute façon. Non seulement vous devez dessiner de jolis signes, mais ces signes doivent être en harmonie avec les autres polices.

Quelques notions sur le dessin vectoriel

Dessiner des images vectorielles nécessite une certaine compréhension du sujet. Nous écrirons quelque chose à ce sujet plus tard (sauf si une âme bienveillante veut bien rédiger un joli tutoriel libre de droit¹⁰). Exigence pour les signes utilisables par JSesh : vous devez dessiner vos signes sous forme de contours remplis en noir sur fond blanc. Le module d'import JSesh peut lire des contours non remplis, mais l'import donnera de meilleurs résultats avec des contours remplis de largeur nulle. Inkscape est capable de convertir des courbes en contours, ce n'est donc pas une exigence très lourde.

Création d'une image de fond

Vous pouvez commencer par obtenir une image du glyphe que vous souhaitez créer. Cette image peut être un dessin au trait ou une photographie d'un glyphe réel. Vous voulez que les lignes noires de votre panneau soient lisibles, changez donc les couleurs si nécessaire (par exemple, si vous avez un dessin au trait du signe, colorez-le en gris clair). Notre exemple sera un rendu basse résolution du signe C102 (Ptah assis

avec w's) de la stèle d'Israël. Le bitmap d'origine est  . Avec un

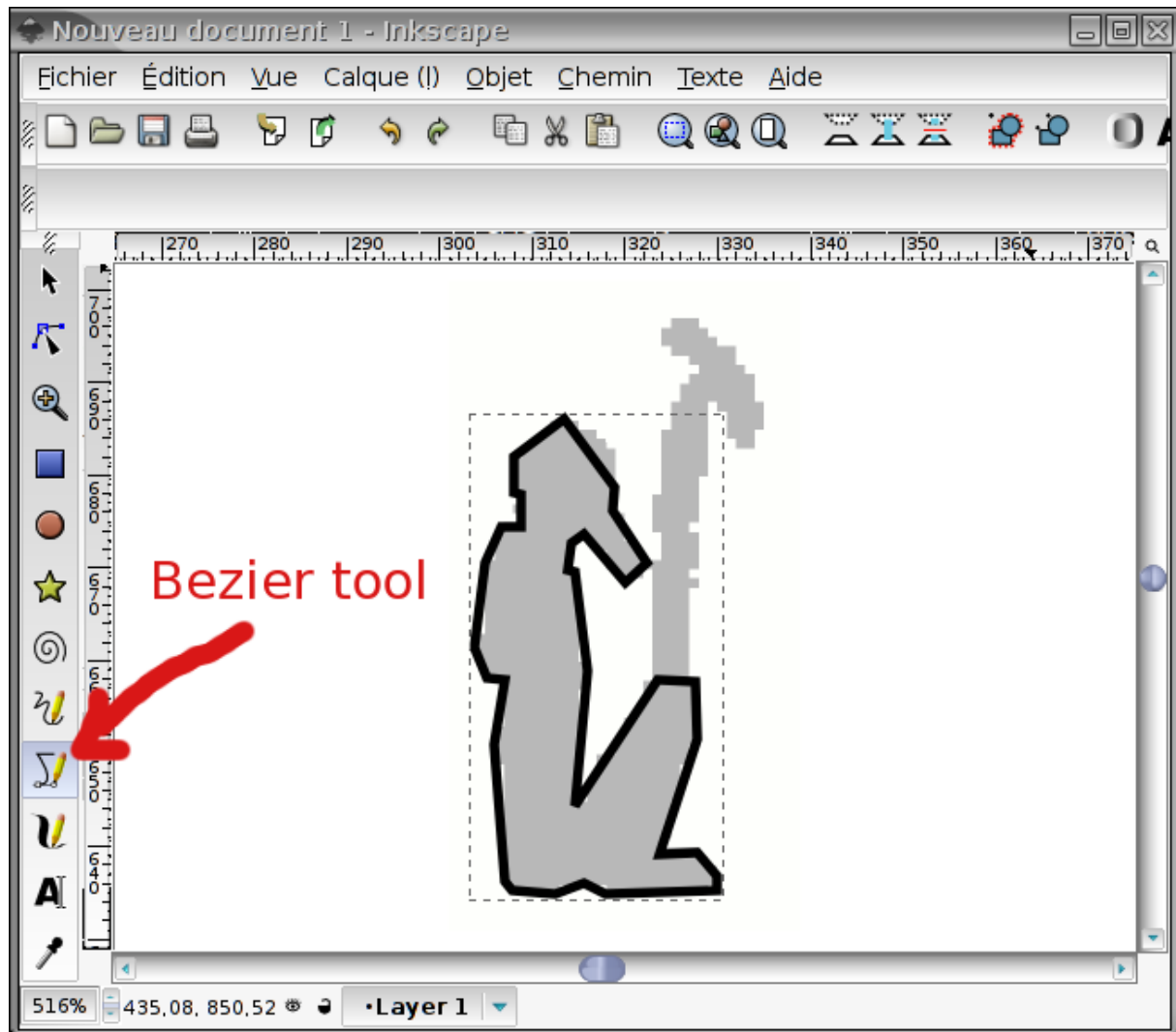
programme de dessin, nous avons estompé l'image pour obtenir 

¹⁰ tel que celui-ci :

https://www.shpylgoreih.fr/documents/Tutoriel_Inkscape_libre_de_droits.pdf

Dessiner le contour

Nous démarrons maintenant Inkscape et importons le dessin bitmap dedans. Créez un nouveau calque, appelez-le "travail". De cette façon, nous ne perturberons pas le dessin bitmap. Nous allons ensuite tracer un contour approximatif de la forme du signe. Comme nous ne sommes pas de grands dessinateurs, nous utilisons l'outil "courbe de Bézier" pour le faire. On obtient le résultat suivant :



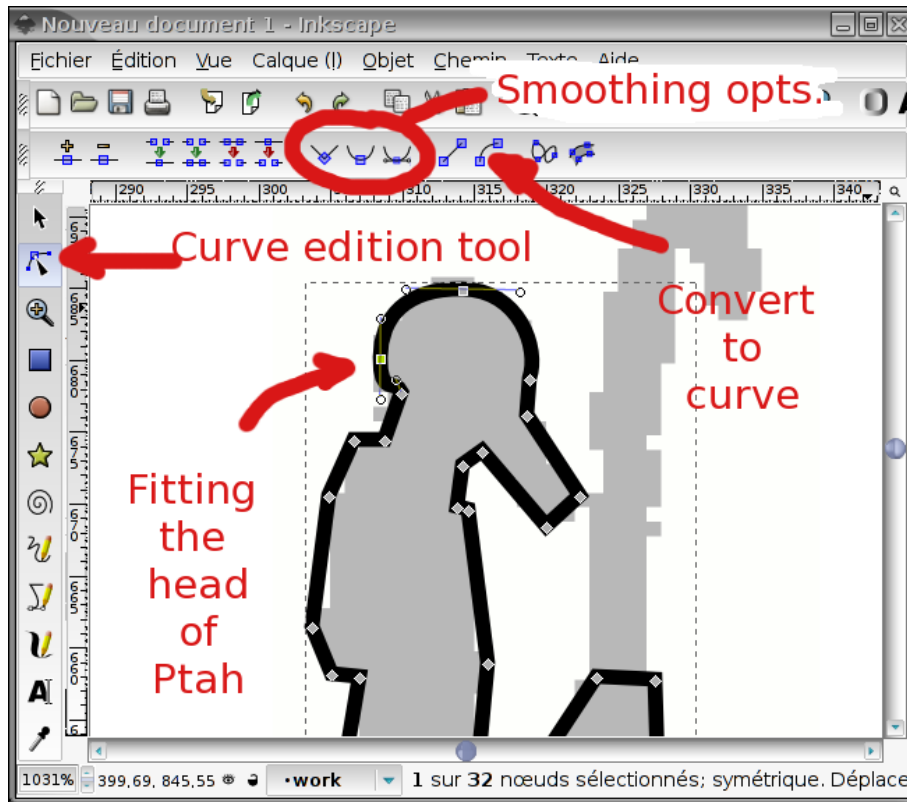
Notez que nous avons sélectionné "pas de remplissage" et une petite largeur de contour.

Nous avons également laissé le signe « W^s » de côté.

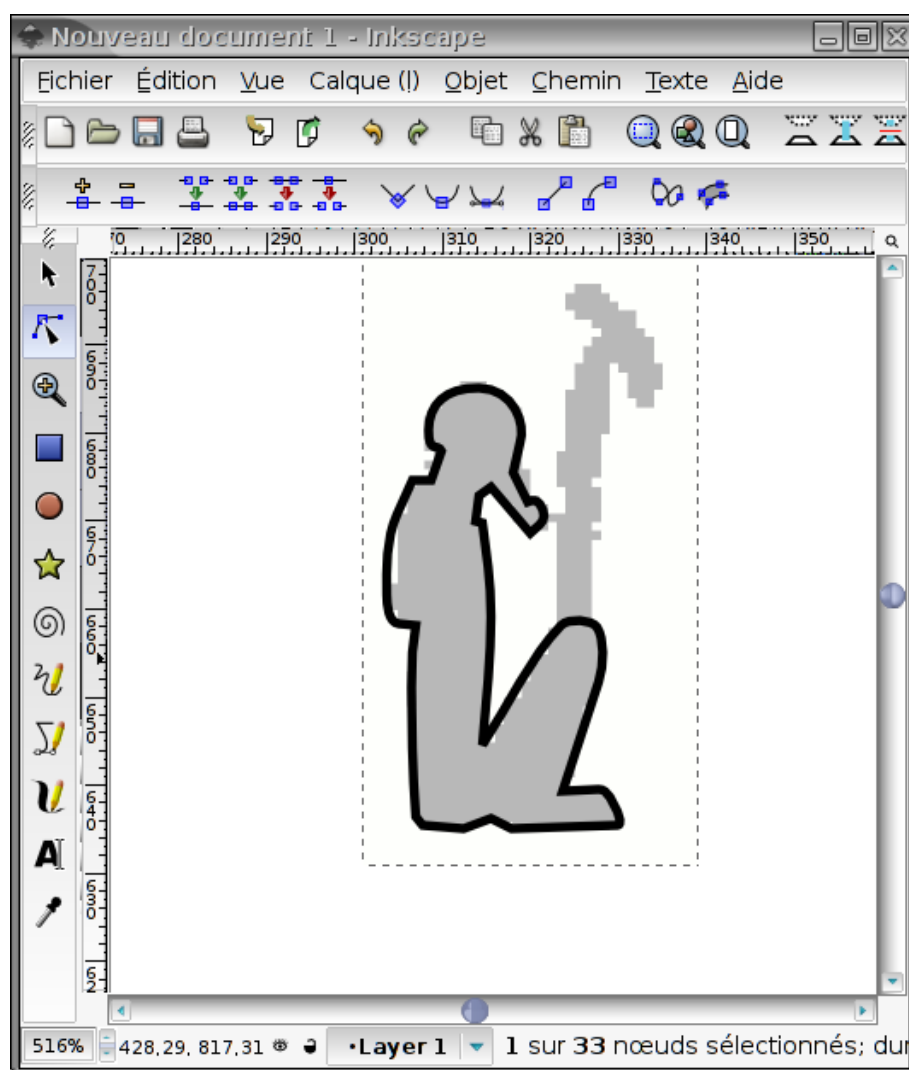
Ajuster le contour

Les segments de ligne droite ne feront pas un bon signe. Donc on convertit certains d'entre eux en courbe, on ajoute de nouveaux points si besoin, etc...

On commence par la tête :



Et voici le résultat :



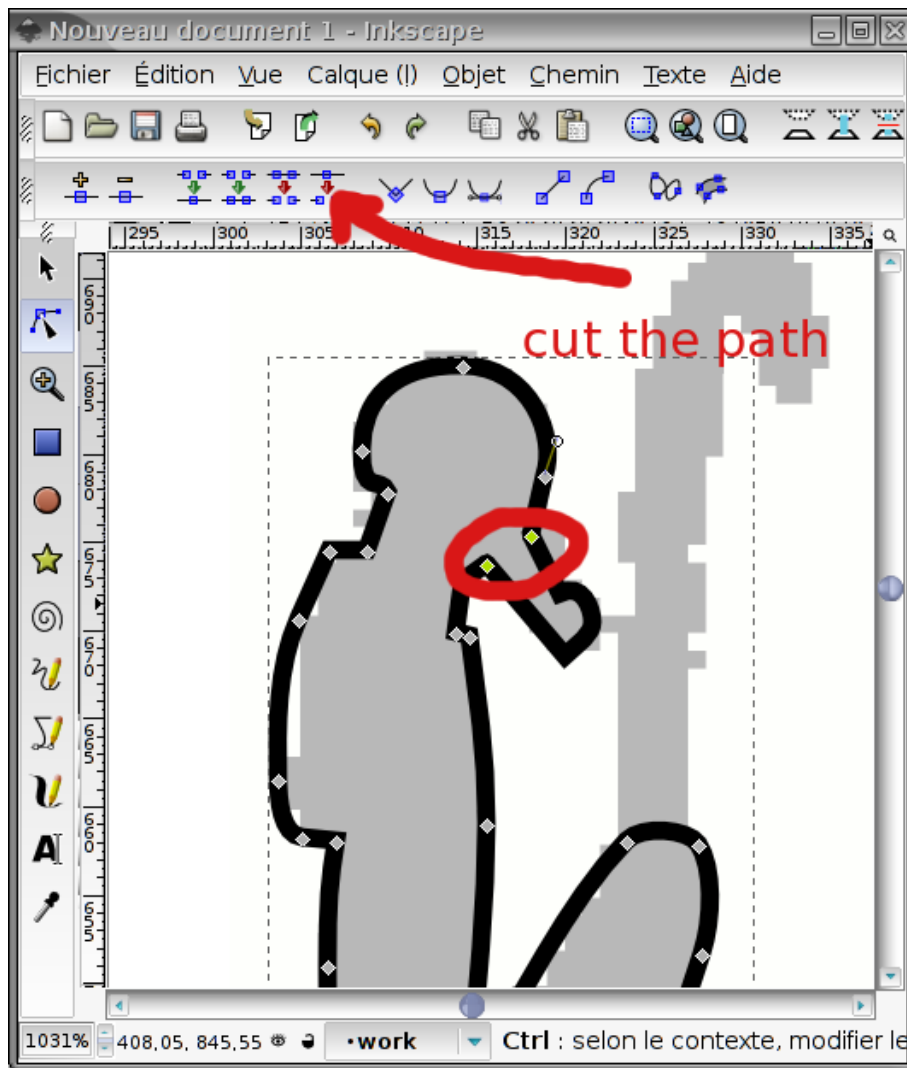
Travail sur les détails

Nous voulons que nos signes soient lisibles en petite taille, nous ne fournirons donc pas trop de détails.

La barbe de Ptah

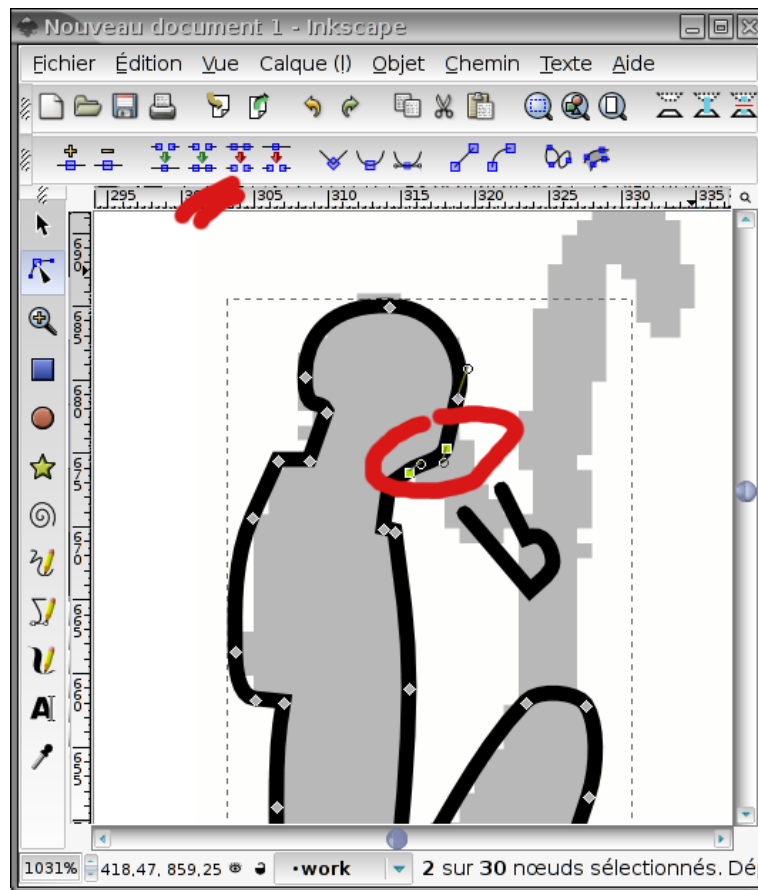
Le contour du corps de Ptah sera agrandi pour faire le signe final. Mais nous aimerions que la barbe soit remplie¹¹. Par conséquent, nous devons la séparer du reste du dessin. Cela se fait en sélectionnant les points de contrôle qui séparent la barbe du corps :

¹¹ Ici, nous avons donné à Ptah une barbe osirienne, ce qui est une erreur.



Et puis, en sélectionnant l'entrée "Séparer" dans le menu "Chemin", on obtient deux objets : la barbe et le reste du corps.

On peut alors combler le vide que l'on vient de créer (et donner un menton à Ptah). 😊



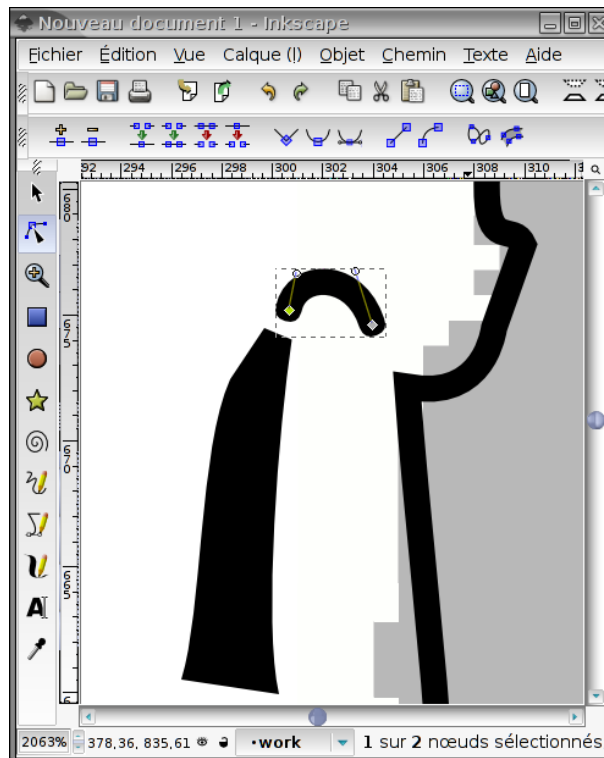
Maintenant, la barbe doit également être fermée. Nous modifions les caractéristiques de remplissage et de trait de la barbe et nous lui donnons un remplissage noir uni, sans trait. Si la barbe est trop fine, on pourra la dilater avec l'action "départ" du menu chemin.

Contrepoids du collier de Ptah

Pour celui-ci, le scan d'origine est un peu approximatif. On creuse un peu l'iconographie, et on arrive à distinguer un collier avec contrepoids en deux parties :

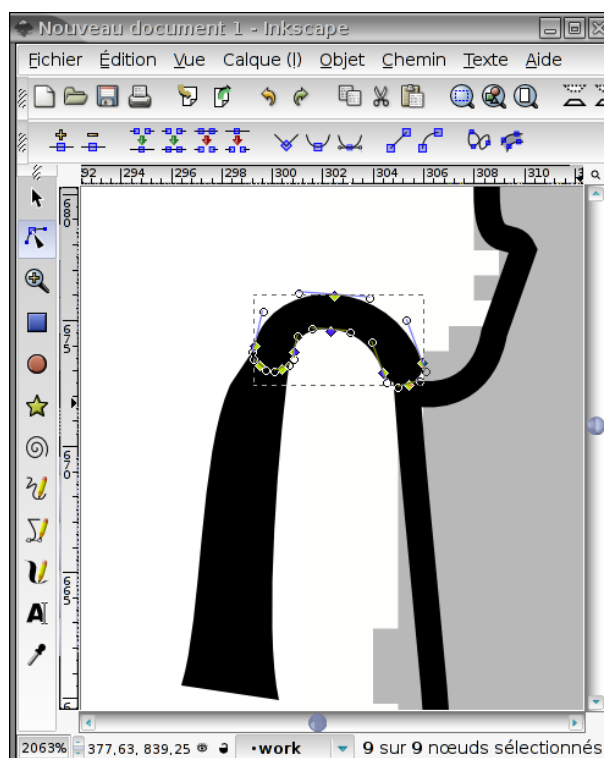


Notre collier sera composé de deux parties : une corde et le contrepoids lui-même. La corde est un simple trait incurvé et le contrepoids est une zone pleine.

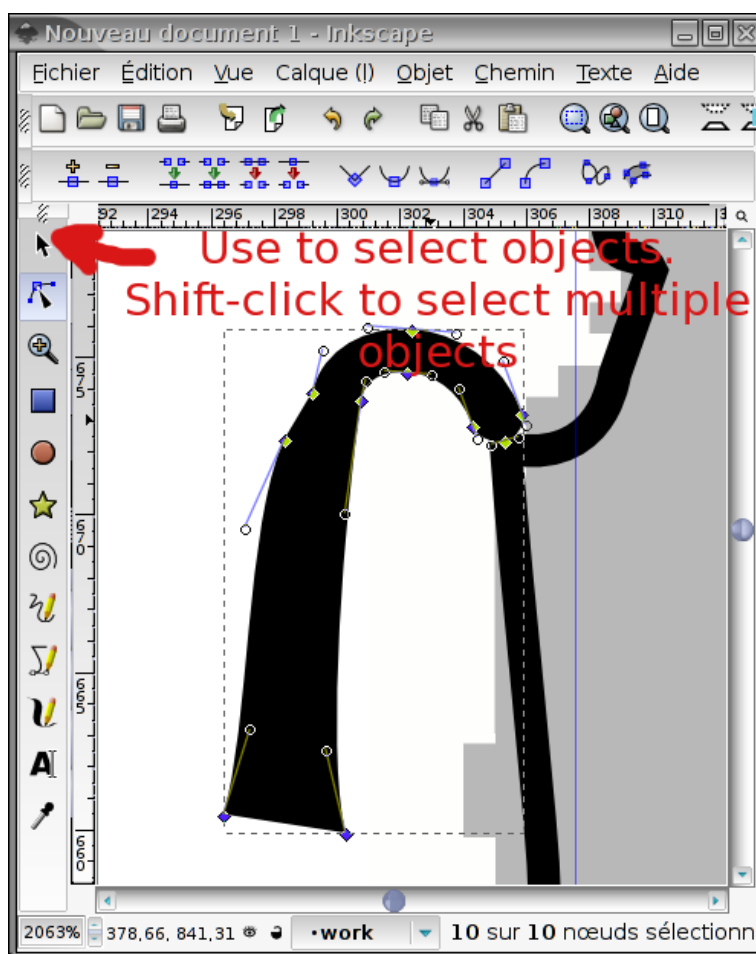


Nous voulons les joindre en un seul objet, qui sera une zone pleine.

Pour ce faire, nous convertissons d'abord la corde en un trait.



On sélectionne ensuite les deux parties du collier, et on utilise l'opérateur "union" dans le menu "chemin".



Choisir la bonne largeur de ligne

À ce stade, vous souhaitez peut-être enregistrer votre travail deux fois. Une fois en tant que sauvegarde, et une, bien sûr, en tant que fichier de travail.

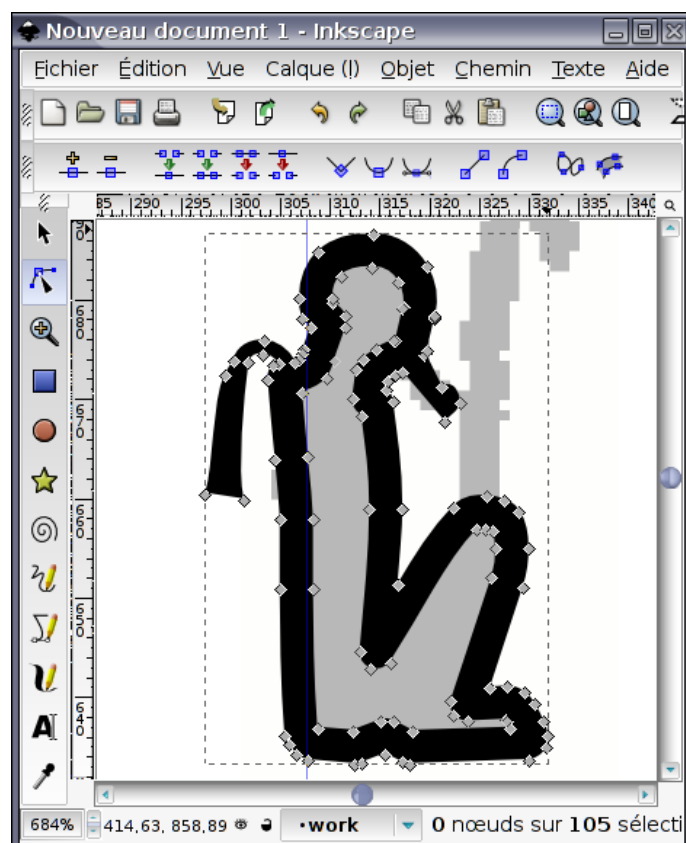
Maintenant, sélectionnez le dessin principal et choisissez une largeur de trait raisonnable pour lui. Cela doit être harmonieux avec le reste des polices.

Il peut être intéressant de faire un zoom arrière afin de voir le signe tel qu'il sera rendu.

Unir tout

Maintenant, assurez-vous que tous les traits sont transformés en chemin et utilisez l'opérateur Union sur eux.

Vous devez obtenir ceci :



Je vous suggère fortement de conserver une version de sauvegarde de votre signe dans laquelle les traits n'ont pas encore été transformés en chemin, cela vous permettra de retravailler votre signe plus facilement ou de le réutiliser dans le cadre d'autres signes. C'est ce que je fais pour mes fontes Ramesside.

Ajout d'autres détails

En utilisant les mêmes techniques, nous arrivons au résultat final :



que nous sauvegardons sous le nom C102.svg.

Commentaires finaux

À partir d'une image de résolution plutôt faible, nous pouvons obtenir un glyphe d'apparence décente. N'oubliez pas que le signe sera plutôt petit et

n'ajoutez pas trop de détails (donner la possibilité d'augmenter les détails du signe avec sa taille est une option intéressante, qui est utilisée dans les polices de Gardiner). Essayez de mélanger avec les signes existants. En fait, si vous avez des sources SVG pour des signes similaires, essayez de les réutiliser.

Le signe donné dans cet exemple n'est pas un très bon exemple. Le "W³s", par exemple, est trop élevé et le signe ne s'harmonisera pas très bien avec les polices existantes. Nous devrions donc raccourcir le sceptre. En général, notre Ptah est trop maigre par rapport aux autres signes. Garder des étapes intermédiaires de votre travail vous permettra de corriger les choses plus facilement.

Par exemple, si les traits sont trop gras et si vous avez conservé une version du signe où les traits sont toujours des traits et n'ont pas encore été remplacés par des chemins, la correction est très facile.

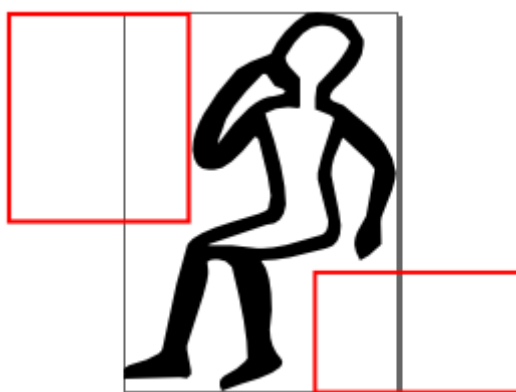
Informations avancées dans Inkscape

Présentation

JSesh a maintenant un mécanisme de ligature relativement avancé, qui n'est pas aussi bon que celui de RES, mais donne des résultats raisonnables dans la plupart des cas. Dans de nombreux cas, JSesh est capable de « deviner » où les groupes ligaturés doivent s'insérer, mais il peut également le faire avec une aide supplémentaire. Cette aide peut être fournie dans le fichier SVG du signe lui-même, et nous avons décidé d'utiliser les mécanismes d'inkscape pour cela.

Zones de ligature

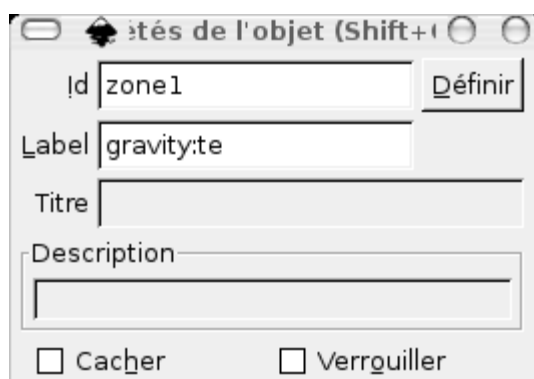
Pour chaque signe, JSesh tentera de calculer deux zones de ligatures, une pour les ligatures de groupes avant le signe, l'autre pour les ligatures de groupes après le signe. Ces zones peuvent être calculées automatiquement, mais l'auteur du signe peut également les spécifier lui-même.



Pour créer l'une des deux zones de ligature dans Inkscape, procédez comme suit :


1. créer un rectangle où le groupe doit s'adapter. Notez que l'ensemble du groupe sera mis à l'échelle pour y tenir, alors dessinez-le suffisamment grand. Je dessine habituellement ces rectangles en rouge, mais ce n'est qu'une convention.
2. ouvrez le menu contextuel du rectangle (clic droit sur le rectangle), et sélectionnez les propriétés de l'objet. Vous obtiendrez la fenêtre des propriétés de l'objet (Figure « Propriétés de la zone » ci-dessous.) Définissez l'ID comme « zone1 » ou « zone2 ». N'oubliez pas de cliquer sur le bouton **Définir** pour valider votre modification. Puis effacez la valeur du champ **Étiquette** (plus d'informations à ce sujet dans la section intitulée « Gravité des zones de ligature »).


Convenons-en, nous utilisons les informations d'identification et d'étiquette d'une manière inaccoutumée. La raison en est purement pragmatique. C'est beaucoup plus facile de faire les choses de cette façon, car le créateur du signe n'a pas besoin de connaître l'organisation interne du format XML.



Gravité des zones de ligature

Le groupe ligaturé ira quelque part dans la zone de ligature. Mais où exactement ? Il peut se tenir au milieu de la zone ou s'accrocher à l'un de ses côtés. En fait, le comportement de l'algorithme de mise en page n'est

pas toujours le même. Dans , le "r" a tendance à s'adapter en bas à

gauche de la zone rectangulaire. Dans , le signe U36 est plus ou moins centré, à la fois horizontalement et verticalement. JSesh permet aux auteurs de signes de concevoir le comportement des "zones de ligature", de la manière exposée ci-dessous. Rappelez-vous de la valeur de l'étiquette supprimée dans le paragraphe précédent. Vous pouvez la renseigner selon la syntaxe suivante :

gravité : spécification de gravité.


où la spécification de gravité peut contenir les valeurs suivantes :

- "s" ou "e" pour demander au groupe de se tenir du côté départ ou du côté fin de la zone¹². Si ni "s" ni "e" ne sont spécifiés, le groupe

¹² Le début et la fin sont tirés du RES de M.-J. Nederhof, et évitez l'utilisation de "gauche" et "droite", qui ne sont pas vraiment utilisables pour les hiéroglyphes !

sera centré horizontalement.

- "t" ou "b" pour demander au groupe de se tenir en haut ou en bas de la zone. Si ni "t" ni "b" n'est spécifié, le groupe sera centré verticalement.

"Propriétés de la zone", *gravity:te* signifie que le groupe qui serait ligaturé dans la zone1 (devant le signe "enfant") collerait au haut du rectangle rouge, et resterait près du signe, par exemple .

Parties de SVG comprises par JSesh

JSesh ne comprend pas complètement SVG. Fondamentalement, il s'attend à ce que l'image soit constituée d'un chemin noir rempli. Toutes les constructions d'un chemin sont comprises, mais je suggère d'utiliser principalement des lignes et des splines.

La plupart du langage est couverte, mais il y a des choses qu'il ne comprend pas encore, en particulier les transformations.

Annexe A. Le système actuel de description des signes

Il est possible de documenter les hiéroglyphes (et en particulier les nouveaux signes) afin que la palette puisse mieux les gérer. Ceci est bien sûr utile pour vos propres nouveaux signes, mais aussi pour les signes JSesh "standard", car les informations sur les signes fournies par JSesh sont actuellement très partielles. Les commentaires des utilisateurs seraient ici les bienvenus, et surtout ceux des professionnels. La principale caractéristique, à partir de la version 2.4.15 de JSesh, est la disponibilité d'un éditeur convivial pour ajouter des informations sur les signes.

Démarrage de l'éditeur de description de signes

L'éditeur de description de signes est un programme distinct. Pour le démarrer :

- sur Windows, il existe un raccourci pour cela¹³
- sur Linux, il devrait y en avoir un aussi
- sur Mac, allez dans votre dossier d'installation JSesh. Vous trouverez l'éditeur dans le dossier "bin". Il s'appelle `signInfoEditor.command`.

Veillez noter que vous pouvez utiliser l'éditeur à volonté, tant que vous n'enregistrez rien. Lorsque vous enregistrez, le résultat sera utilisé la prochaine fois que vous lancerez JSesh.

Modification des descriptions des signes

Les signes sont définis par les informations suivantes :

- Les translittérations associées au signe
- Les signes qui sont des composants de ce signe. Par exemple, A6 contient un signe W54
- Le(s) signe(s) dont ils sont des variantes
- Des descriptions en texte libre du signe, pouvant inclure par exemple des remarques bibliographiques
- Les tags sont de courts éléments de description attachés aux signes et utilisés pour les sélectionner. Par exemple, "ennemi" est attaché aux signes qui représentent un ennemi.

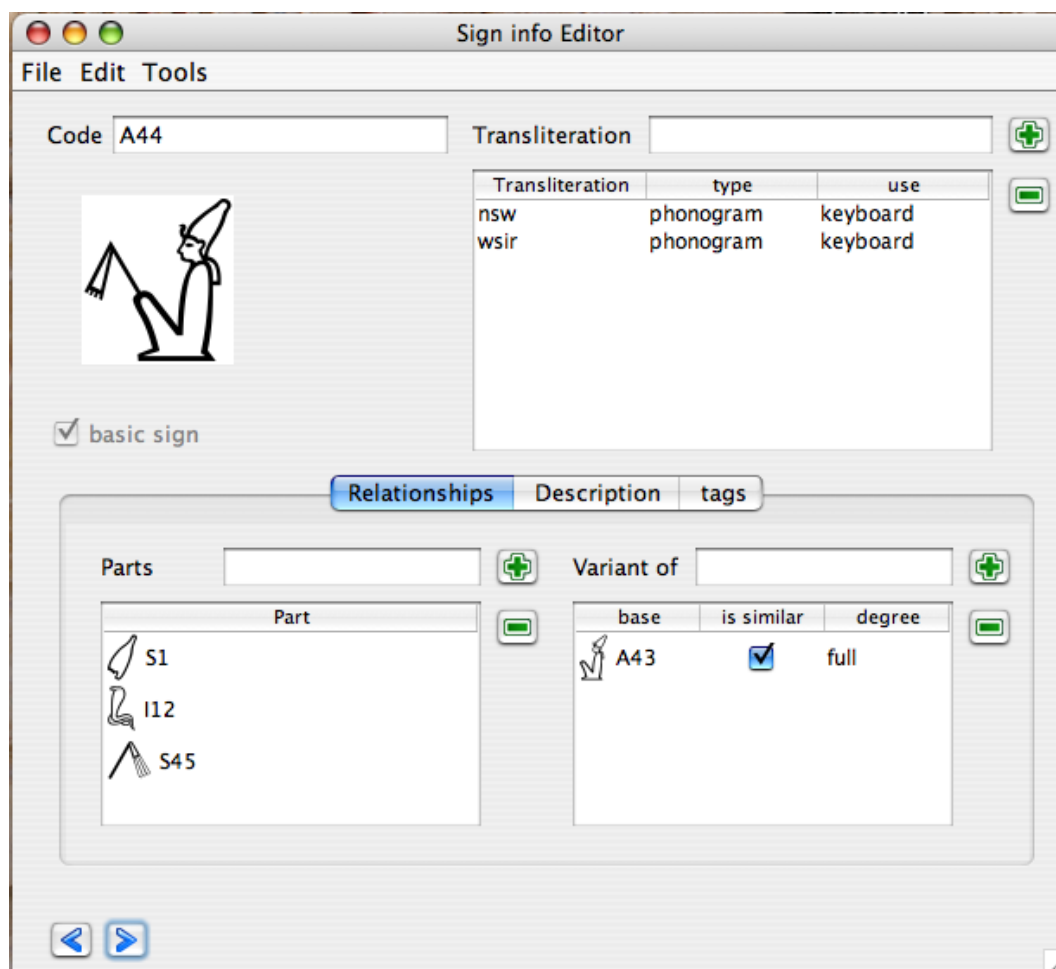
Lorsque vous démarrez l'éditeur de description de signes, il lit automatiquement les descriptions de signes standard ainsi que vos propres descriptions. Vous pouvez ensuite ajouter ou modifier des données, et

¹³ Si le raccourci ne s'est pas créé au moment de l'installation ou si vous l'avez supprimé par accident, allez dans le dossier C:\Programs (x86)\Jsesh-7.5.5 (par exemple), puis faites un clic droit sur le fichier `SignInfo.exe` et sélectionnez **Envoyer vers le bureau – Créer un raccourci** (Note du traducteur).

enregistrer le résultat (en choisissant simplement « enregistrer » dans le menu). Notez que l'éditeur vous empêchera de modifier les informations stockées dans les descriptions des signes système.

Une caractéristique importante de l'éditeur est que vous pouvez utiliser la palette de signes (à partir du menu d'outils) pour sélectionner des signes.

La fenêtre principale de l'éditeur d'informations sur les signes



La fenêtre principale de l'éditeur d'informations sur les signes permet de modifier les informations sur un signe donné. La sélection du signe sur lequel vous souhaitez travailler peut se faire de plusieurs manières. Vous pouvez naviguer avec les flèches en bas de la fenêtre, taper le code du signe dans le champ **Code**, ou simplement faire glisser et déposer le signe depuis la palette. Une fois qu'un signe apparaît, vous pouvez ajouter des informations à son sujet.

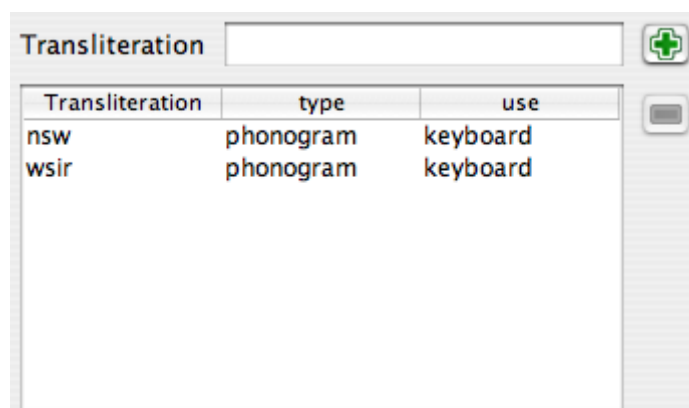
La case à cocher *signe de base* est utilisée pour indiquer que le signe doit apparaître dans la palette de signes même si *afficher tout* n'est pas sélectionné.

À partir de cette fenêtre, vous pouvez modifier les translittérations des signes et bien d'autres choses. La partie inférieure de la fenêtre donne accès à trois types d'informations : les relations entre ce signe et les autres signes, la description en texte libre du signe et les tags.

Translittération

Il est possible d'associer plusieurs translittérations à un signe donné. Les translittérations peuvent être utilisées dans divers contextes : pour rechercher un signe (dans la palette, ou en tapant la translittération directement au clavier), ou simplement à titre informatif, pour des translittérations peu courantes.

L'éditeur de translittération



Pour saisir une nouvelle translittération, appuyez simplement sur le bouton "+". Si vous saisissez du texte dans le champ *translittération*, ce texte sera utilisé comme nouvelle translittération.

Si vous souhaitez supprimer une de vos translittérations, sélectionnez la ligne et appuyez sur le bouton "moins". Ce bouton est grisé si aucune ligne n'est sélectionnée, ou si la ligne sélectionnée ne peut pas être supprimée. Vous pouvez alors ajuster les valeurs *type* et *utilisation*.

Je ne sais plus si le *type* est utile ou non, et il n'est actuellement pas utilisé par JSesh. La colonne *utilisation* est en revanche assez importante.

Les *utilisations* expliquent dans quel contexte cette translittération est utilisée dans le logiciel. Notez que chaque niveau d'utilisation est inclus dans le suivant. Par exemple, si vous choisissez *clavier*, le signe sera également utilisé dans les contextes *palette* et *informatif*.

Types

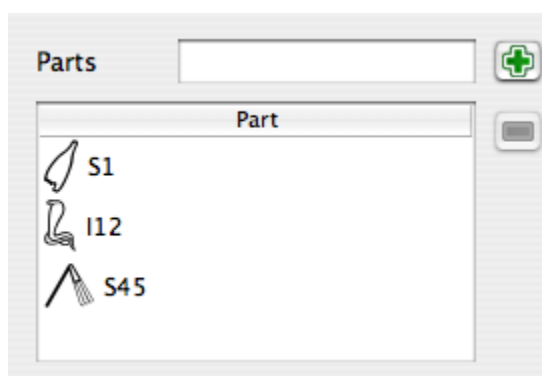
- phonogramme :
- idéogramme : doit être utilisé à la fois pour les idéogrammes simples (ceux suivis de Z1), et pour les déterminants dits *phonétiques*, qui sont en réalité des idéogrammes déguisés.
- abréviation : à utiliser pour les signes qui sont de véritables abréviations de mots. Ils ne sont normalement pas suivis de Z1 par écrit. Exemple : G37 peut être utilisé comme abréviation pour *śrī*.
- typique : le signe est typique pour un mot. Souvent utile pour certains déterminants. Je l'utilise pour la valeur "bin" de G37.
- clavier : le signe sera accessible via cette translittération dans JSesh

lors de l'utilisation du clavier pour saisir des signes. Par exemple, la translittération "iw" de D54.

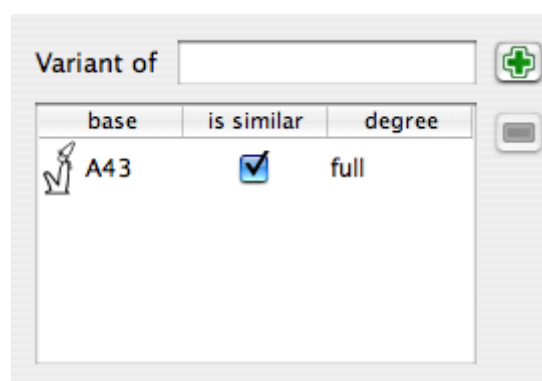
- palette : le signe ne sera pas accessible depuis le clavier via cette translittération, mais sera accessible via la palette. Notez que si un utilisateur utilise la translittération pour accéder à un signe dans la palette, il pourra y accéder par sa translittération par la suite.
- informatif : la translittération n'est donnée qu'à titre informatif. Elle apparaîtra dans le champ "valeur" de la palette, mais c'est tout.

Éditeur de parties de signes

L'éditeur de parties de signes permet de décrire les éléments d'un signe en tant que composants d'autres signes. Par exemple, le roi de Haute-Égypte A44 porte une couronne blanche, tient un sceptre nekhakha et porte également un uraeus. Tous ceux-ci, à leur tour, sont des hiéroglyphes. Notez que vous n'avez pas besoin de lister toutes les pièces. Par exemple, si un dieu porte la couronne Atef, qui à son tour contient la plume de Maat, ne mentionnez pas la plume. Elle est déjà incluse dans le signe de l'Atef (Merci à J. Hallof pour cette remarque pendant la conférence d'Oxford 2006).



Editeur de variantes



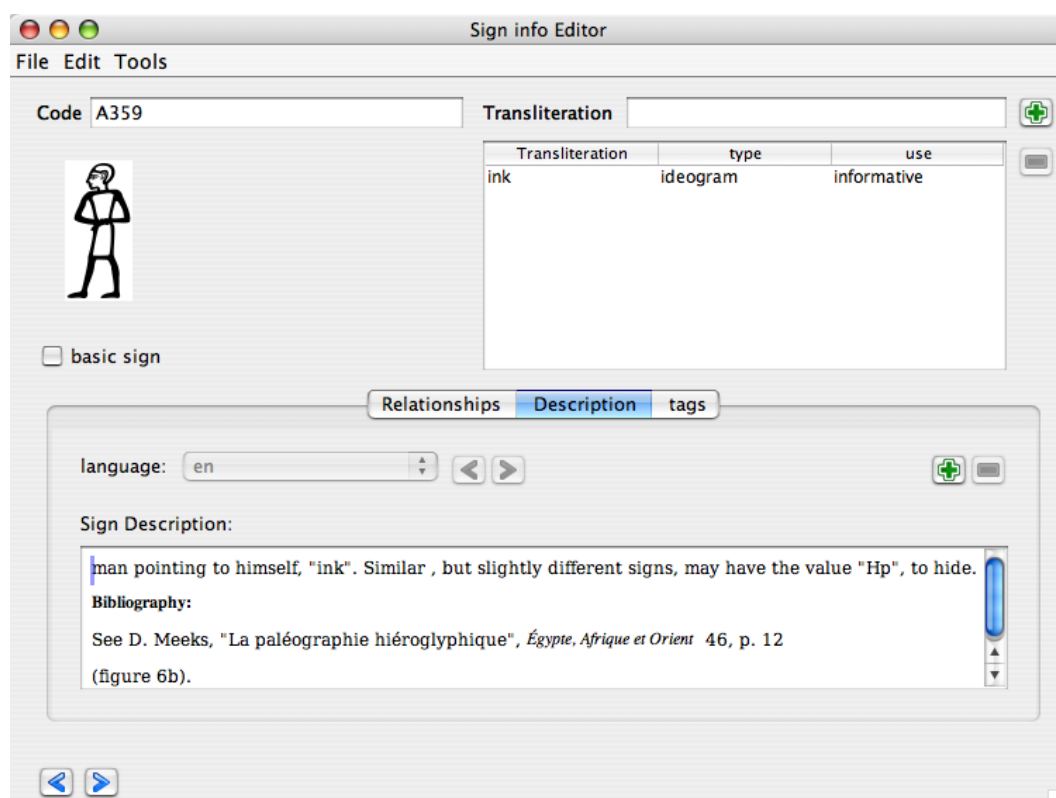
La notion de variante est délicate. En fait, ce n'est pas si bien défini et, pire encore, sa signification utile peut dépendre du contexte. Il y a en fait deux notions. L'un est graphique. Une *variante graphique* d'un signe est un signe qui *ressemble* à un autre. Une autre notion est linguistique. Un signe est une *variante linguistique* d'un autre s'il a les *mêmes valeurs et usages*. Les deux notions se chevauchent souvent, mais pas toujours. Par exemple, Y2

est à la fois une variation graphique de Y1 et une variante linguistique de celui-ci. D'autre part, A17A n'est qu'une variante graphique de A17. Il n'a pas du tout les mêmes utilisations; et Z7 et G43 sont des variantes linguistiques, mais pas du tout des variantes graphiques.

Ces notions seront utilisées par JSesh à la fois pour la palette de signes (avec son bouton *variante de*) et pour le système de recherche. Il peut être utile, lors de la recherche de mots avec G43, de retrouver ceux avec Z7.

- *base* : signe original
- *est similaire* : à cocher si les signes se ressemblent.
- *niveau de similitude* : indiquez à quel point la relation entre les deux signes est étroite. Il peut s'agir de :
 - *complet* : une variante complète d'un signe S est un signe avec exactement les mêmes usages et valeurs que S.
 - *partiel* : les usages se chevauchent de manière significative. Habituellement, la variante couvrirait certaines des utilisations du signe original.
 - *autre* : autres types de variantes. Par exemple, D36 (le bras) peut être considéré comme une variante de D37 (le signe "rdi") dans certains contextes, mais les deux signes ont une identité bien distincte.
 - *non* : le signe n'est pas du tout une variante linguistique
 - *non précisé* : vous ne savez pas vraiment, ou n'avez pas le temps de vous en soucier. 😊

Éditeur de descriptions



Ce champ permet des commentaires en texte libre pour les signes.

L'utilisation la plus importante pour cela est de documenter les signes peu communs, en donnant des références bibliographiques à leur sujet, et en général en aidant l'utilisateur à sélectionner le meilleur signe possible. Si vous souhaitez que la description de votre signe soit incluse dans la version distribuée de JSesh, ils doivent avoir des références bibliographiques pour étayer vos affirmations. Veuillez également noter que JSesh est un logiciel libre, mais qu'il doit respecter les droits d'auteurs. Donc, ce n'est pas le lieu pour y copier aveuglément des listes de signes des grammaires usuelles (cela ne veut pas dire bien sûr que vous ne pouvez pas les utiliser comme sources). Vous pouvez écrire des descriptions dans plusieurs langues, même s'il serait préférable qu'il y ait au moins une version anglaise pour chaque signe.

Éditeur de listes de signes



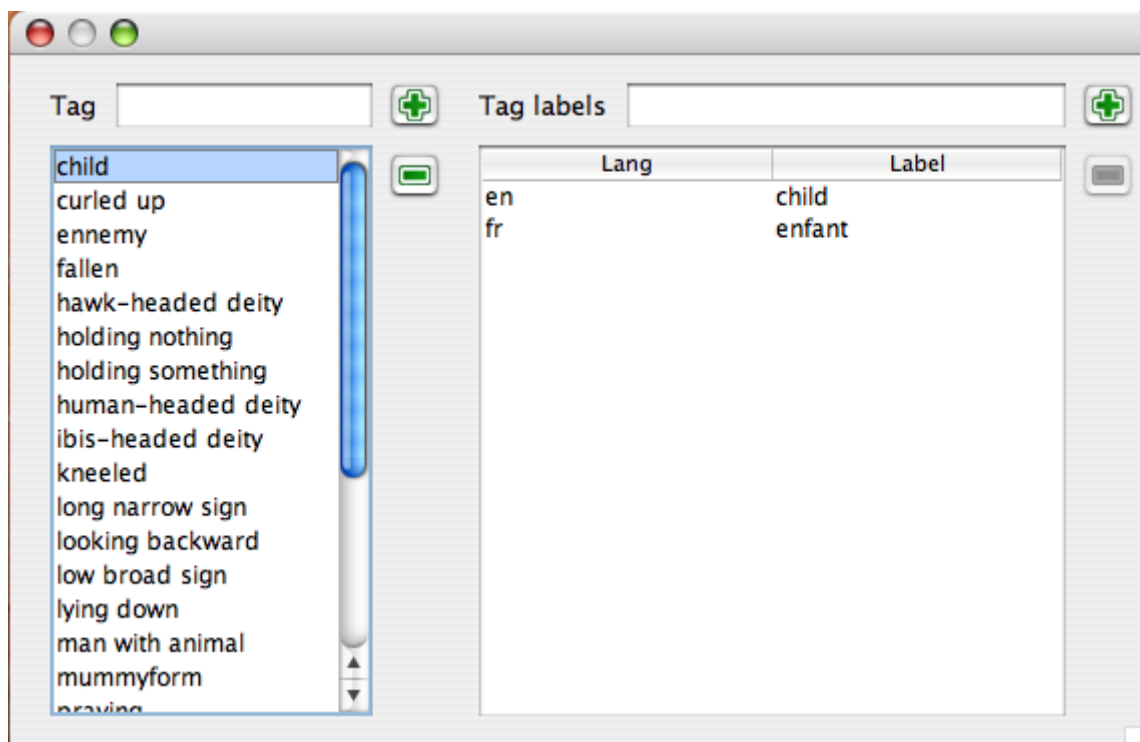
Cette fenêtre permet de décrire un signe selon divers mots descriptifs courts appelés "tags". Ceux-ci sont disponibles dans la palette pour sélectionner des sous-catégories d'une famille donnée.

Exemple : les caractères *debout* (évidemment, les tags utilisés ici ne s'appliquent pas à A44). La fenêtre de gauche affiche tous les tags disponibles, listant d'abord ceux qui sont déjà utilisés pour cette famille, puis les autres tags. La meilleure façon de marquer des signes avec des tags est de regarder des signes similaires et de voir comment cela a été fait.

J'aimerais attirer votre attention sur des tags très utiles, qui sont un peu différents. Il s'agit du *signe grand et étroit*, du *signe bas et large* et du *signe long et étroit*. Ces tags sont utilisés en tant que familles spéciales dans la palette JSesh.

Pour ajouter un nouveau tag à un signe, sélectionnez simplement le tag dans la liste de gauche et cliquez sur la *flèche de gauche à droite*. Pour supprimer un tag, utilisez le même principe. Notez que vous ne pouvez pas supprimer les tags qui sont fournis dans la liste JSesh officielle.

La fenêtre de création de tags



Cette fenêtre (accessible via le menu *Outils*) permet de créer de nouveaux tags et de leur associer une traduction multilingue. Ces traductions ne sont pas utilisées actuellement.

Les menus

Fichier

- Ouvrir le fichier par défaut de l'utilisateur : ouvre le fichier qui contient vos propres définitions de signes. Ceux-ci seront automatiquement utilisés par JSesh lors de son prochain lancement.
- Enregistrer : enregistre votre travail dans votre fichier de définition personnel. Tant que cela n'a pas été fait, votre fichier est inchangé.
- Effacer : créer un tout nouvel espace de travail, sans aucune donnée autre que les "officielles". Votre fichier utilisateur ne sera de toute façon modifié que si vous enregistrez votre travail.

Modifier

- copier : copier la définition de ce signe pour une utilisation future
- coller : collez les données copiées pour un autre signe dans cette définition de signe. Très utile pour traiter des variantes proches.

Outils

- Afficher/Masquer la palette : permet d'ouvrir la palette des signes, pour sélectionner les signes (par glisser-déposer)
- Afficher l'éditeur de signes : ouvre (ou ferme) la fenêtre de l'éditeur de signes, pour créer de nouveaux signes.

Contribution de votre description de signes à JSesh

Vos descriptions de signes sont stockées dans un fichier appelé *sign_definition.xml*, qui est placé dans :

- (votre répertoire personnel)/Library/Preferences/JSesh sur Macintosh. Par exemple :
/Users/rosmord/Library/Preferences/JSesh/signs_definition.xml sur ma machine.
- *C:\Documents and Settings\YOUR LOGIN\JSeshData* sur Windows (en gros, JSeshData dans votre dossier personnel). Normalement, le répertoire JSeshData est créé par JSesh, vous pouvez donc le rechercher si vous avez des doutes. Par exemple, *C:\Documents and Settings\Rosmord\JSeshData*.
- \$HOME/.jsesh sur Linux.

Pour apporter votre contribution dans la description de signes pour JSesh, envoyez-moi simplement ce fichier. Je déciderai de ce qui peut aller dans la distribution générale de JSesh. Il y a beaucoup de problèmes que je dois prendre en compte : le logiciel doit rester assez général, assez correct, et je dois éviter les violations du droit d'auteur.

Annexe B : Informations techniques sur le fichier de description des signes

Attention : contenu techniquement explicite. Âmes pures, détournez les yeux. Un système plus convivial a été créé.

Alternativement, vous pouvez éditer directement votre fichier de description. Vous n'avez besoin que d'un éditeur simple pour faire cela : le bloc-notes peut faire l'affaire sur Windows, et des logiciels comme TextWrangler peuvent être utilisés sur Mac OS X. Les fichiers XML sont constitués de texte brut.

JSesh n'acceptera pas les fichiers mal formés, vous risquez donc de ne pas pouvoir lancer JSesh. Si c'est le cas, corrigez le fichier *sign_definition.xml* ou renommez-le en quelque chose d'autre, afin qu'il soit ignoré. À l'avenir, j'ajouterai un éditeur convivial, mais je ne le ferai pas tant que le format ne sera pas complètement défini. Le fichier doit avoir la forme suivante :

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE signs PUBLIC "-//ORG/QENHERKHOPESHEF//DTD SIGNDESCRIPTION 1.0"
"sign_description.dtd">
<signs>

<!-- placez ici vos définitions de signes -->

</signs>
```

Il est important d'avoir exactement ce contenu, en particulier la ligne DOCTYPE.

Voici un petit exemple (en fait, une partie du fichier de description des signes standard de JSesh). Ce dossier décrit les signes C1 et C1A. Vous voyez qu'ils sont classés dans un certain nombre de catégories. Ce sont à la fois des divinités à tête humaine et des personnages assis. La translittération de C1 est donnée. Nous en avons également fourni un pour C1A. Le code *relevance='1'* signifie que cette translittération est ici uniquement à titre informatif. En fait, le format XML a été préparé pour accueillir beaucoup de données différentes, ce qui n'est pas encore vraiment utilisé par JSesh, et je suis très intéressé de recevoir des suggestions à ce sujet. La définition du format (son "dtd") est donnée juste après cette annexe.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE signs PUBLIC "-//ORG/QENHERKHOPESHEF//DTD SIGNDESCRIPTION 1.0"
"sign_description.dtd">
<signs>

<!-- Comme C est actuellement la partie la plus complète des polices JSesh,
```


nous essayons de la couvrir entièrement. -->

```
<tagCategory tag="divinité à tête humaine" label="divinité à tête humaine"/>
<tagCategory tag="divinité à tête de faucon" label="divinité à tête de
faucon"/>
<tagCategory tag="divinité à tête d'ibis" label="divinité à tête d'ibis"/>
<tagCategory tag="divinité à tête de bélier" label="divinité à tête de
bélier"/>

<sign sign="C1">
  <hasTransliteration sign="C1" transliteration="ra"/>
  <hasTag tag="divinité à tête humaine"/>
  <hasTag tag="assis"/>
  <contains partCode="N6"/>
</sign>

<sign sign="C1A">
  <similarTo baseSign="C1"/>
  <hasTransliteration transliteration="ra" relevance="1"/>
  <hasTag tag="divinité à tête humaine"/>
  <hasTag tag="assis"/>
  <contains partCode="N6"/>
  <contains partCode="S40"/>
</sign>
</signs>
```

Notez que les balises doivent être définies avant d'être utilisées (comme `tagCategory`). Un tag a un nom et une étiquette ; il est en effet possible de définir des étiquettes dans plusieurs langues, bien que cela ne soit pas vraiment utilisé par JSesh maintenant.

DTD des descriptions de signes

Pour les Geeks, voici la DTD¹⁴ actuelle utilisée pour les descriptions des signes. Elle est encore expérimentale, et a déjà changé depuis la version 2.4.13.

```
<!-- DTD utilisée pour décrire les caractéristiques des signes. -->
<!-- NOM DU CATALOGUE : "-//ORG/QENHERKHOPESHEF//DTD SIGNDESCRIPTION 1.0" -
-->

<!ENTITY % signInfo
"variantOf|hasTransliteration|partOf|contains|signDescription|isDeterminati
ve|hasTag|phantom"
>

<!ELEMENT signs
(sign|determinativeCategory|tagCategory|tagLabel|%signInfo;)*>

<!-- L'élément sign est facultatif, mais permet d'avoir un fichier mieux
structuré. -->

<!ELEMENT sign (%signInfo;)* >

<!ATTLIST sign
  sign CDATA #REQUIRED
  alwaysDisplay (y|n) 'n'
>

<!--
```

¹⁴ Description de type de document.

La notion de variante utilisée ici est en quelque sorte ad-hoc.

Le problème des variantes est qu'il y a deux notions différentes derrière, toutes deux utiles dans notre logiciel.

La première notion est la variante LINGUISTIQUE. Un signe est une variante linguistique d'un autre s'il a les mêmes usages.

Par exemple, Y2 est une variante linguistique de Y1. Maintenant, Y2 "ressemble" également à Y1. Nous l'appellerons une "variation graphique".

Les deux notions sont indépendantes, bien que statistiquement liées. Par exemple, Z7 est une variante linguistique de G43, mais pas une variation graphique de celui-ci.

la notion de "ressembler" à un autre signe est couverte par l'attribut "isSimilar".

Dans de nombreux cas, notamment pour les déterminatifs, les signes ne sont pas toujours entièrement substituables les uns aux autres.

Pour permettre l'utilisation d'informations « variantes » dans les recherches, nous introduisons l'attribut « linguistique ».

Soit B une variante de A.

"complet" signifie que toutes les utilisations de B sont également des utilisations possibles de A, et toutes les utilisations de A sont des utilisations de B.

"autre" signifie que B est plus spécifique que A, ou que le degré est inconnu

"partiel" signifie que les utilisations de A et B se recoupent, mais elles ont également toutes deux des utilisations très différentes.

Par exemple, le signe D36 (ayin) est une variante partielle de D37 (di), car D36 peut écrire "di". Pourtant,

dans ce cas, je ne considérerai pas le D37 comme une variante du D36, car cela ferait plus de mal que de bien.

"non" est utilisé lorsque le signe n'est pas du tout une variante linguistique. Dans ce cas, isSimilar est normalement "y".

-->

```
<!ELEMENT variantOf EMPTY>
<!ATTLIST variantOf
  sign CDATA #IMPLIED
  baseSign CDATA #REQUIRED
  isSimilar (y|n) 'y'
  linguistic (full|partial|other|no|unspecified) 'unspecified'
>
```

```
<!ELEMENT hasTransliteration EMPTY>
<!-- le but principal de la translittération est d'aider quelqu'un à
trouver un signe. -->
<!-- quelques informations supplémentaires ci-dessous -->
<!--
```

l'attribut "use" explique où la translittération sera visible dans JSesh.

"clavier" signifie que le signe est typique de cette translittération, c'est-à-dire qu'il doit être utilisé

dans le logiciel principal lorsque vous utilisez « espace » pour faire le tour des signes possibles.

"palette" signifie que le signe est une valeur pas trop inhabituelle pour une translittération donnée.

il doit être accessible via la palette.

"informatif" signifie que la valeur est ici à des fins d'information uniquement.

Type permet de spécifier d'où vient la valeur. Il se peut qu'un signe soit un vrai phonogramme (par exemple G1 pour aleph), ou un idéogramme, ou une abréviation, ou simplement être typique de certains mots (par exemple "bin" n'est pas vraiment une valeur pour G37 ; mais c'est typique. G37 est

cependant une abréviation connue pour Sri.

```
-->
```

```
<!ATTLIST hasTransliteration
  sign CDATA #IMPLIED
  transliteration CDATA #REQUIRED
  use (keyboard|palette|informative) 'keyboard'
  type (phonogram|ideogram|abbreviation|typical) 'phonogram'
>
```

```
<!ELEMENT hasShape EMPTY>
<!ATTLIST hasShape
  sign CDATA #IMPLIED
  shape (tallNarrow|lowBroad|lowNarrow) #REQUIRED
  order CDATA #IMPLIED
>
```

```
<!ELEMENT partOf EMPTY>
<!ATTLIST partOf
  sign CDATA #IMPLIED
  baseSign CDATA #REQUIRED
>
```

```
<!-- Plus facile à utiliser (et à déclarer) que isPartOf -->
```

```
<!ELEMENT contains EMPTY>
<!ATTLIST contains
  sign CDATA #IMPLIED
  partCode CDATA #REQUIRED
>
```

```
<!ELEMENT determinativeCategory EMPTY>
<!ATTLIST determinativeCategory
  category CDATA #REQUIRED
  lang NMTOKEN 'en'
  label CDATA #REQUIRED
>
```

```
<!ELEMENT isDeterminative EMPTY>
<!ATTLIST isDeterminative
  sign CDATA #IMPLIED
  category CDATA #REQUIRED
>
```

```
<!ELEMENT hasTag EMPTY>
<!ATTLIST hasTag
  sign CDATA #IMPLIED
  tag CDATA #REQUIRED
>
```

```
<!-- Déclare une balise (sans aucune étiquette) -->
<!ELEMENT tagCategory (tagLabel)*>
<!ATTLIST tagCategory
  tag CDATA #REQUIRED
>
```

```
<!-- Declares a label for a tag. -->
<!ELEMENT tagLabel EMPTY>
<!ATTLIST tagLabel
  tag CDATA #IMPLIED
  lang NMTOKEN 'en'
  label CDATA #REQUIRED
>
```

```
<!-- description du signe, au format du Manuel de codage.
```

- lang peut être utilisé pour décrire la langue. Utilisateur "fr" pour le français, "de" pour l'allemand...

-->

```
<!ELEMENT signDescription (#PCDATA)>
```

```
<!ATTLIST signDescription
```

```
  sign CDATA #IMPLIED
```

```
  lang CDATA 'en'
```

```
>
```

<!-- Un fantôme est un code redondant. Il précise qu'un code donné est l'équivalent exact d'un autre.

Cela peut être utilisé à des fins de normalisation. Par exemple, il y a quelques signes qui ont des encodages différents dans winglyph, JSesh et Inscribe. L'utilisation du fantôme a) évite d'avoir plusieurs signes et b) permet de créer un texte normalisé.

-->

```
<!ELEMENT phantom EMPTY>
```

```
<!ATTLIST phantom
```

```
  baseSign CDATA #REQUIRED
```

```
  existsIn CDATA 'jsesh'
```

```
>
```

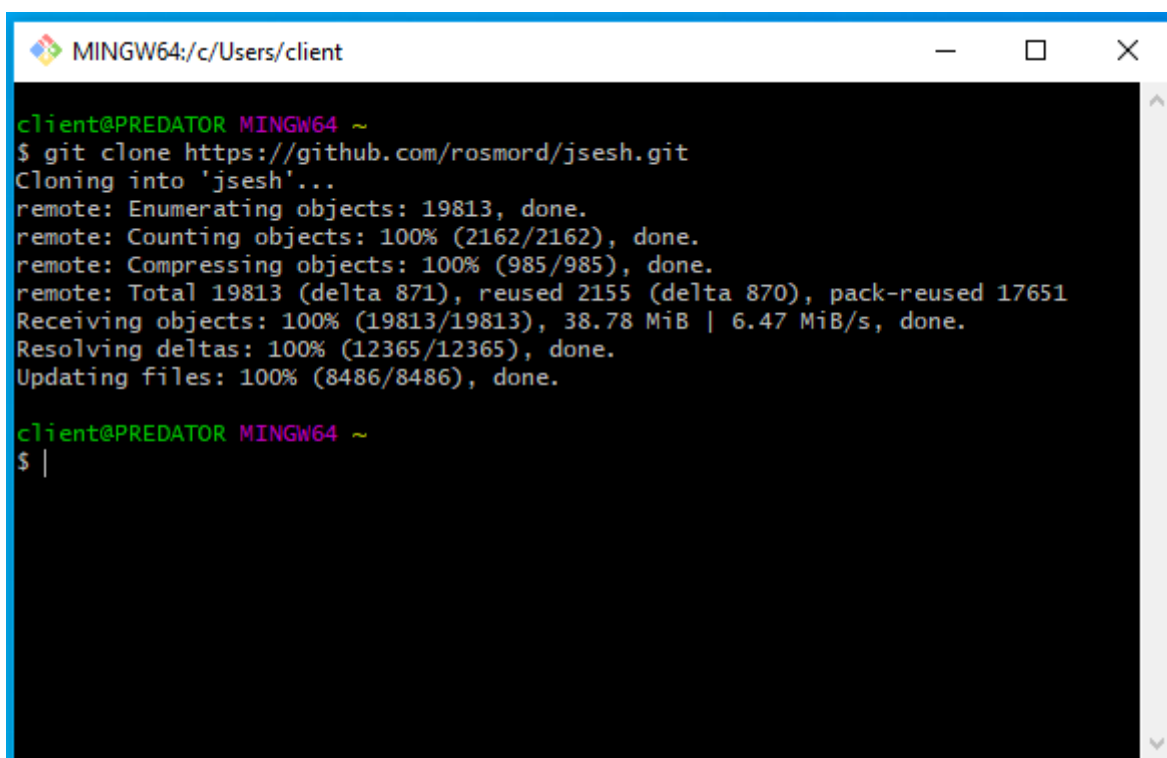
Annexe C : Guide du développeur

Si vous savez programmer en Java, vous pouvez utiliser JSesh comme bibliothèque pour vos propres programmes.

Obtenir le code

La méthode préférée est d'utiliser *git*, avec la commande :

```
git clone https://github.com/rosmord/jsesh.git
```

A screenshot of a terminal window titled 'MINGW64:/c/Users/client'. The terminal shows the execution of the command 'git clone https://github.com/rosmord/jsesh.git'. The output indicates that the repository was successfully cloned into a directory named 'jsesh'. The terminal text is as follows:

```
client@PREDATOR MINGW64 ~
$ git clone https://github.com/rosmord/jsesh.git
Cloning into 'jsesh'...
remote: Enumerating objects: 19813, done.
remote: Counting objects: 100% (2162/2162), done.
remote: Compressing objects: 100% (985/985), done.
remote: Total 19813 (delta 871), reused 2155 (delta 870), pack-reused 17651
Receiving objects: 100% (19813/19813), 38.78 MiB | 6.47 MiB/s, done.
Resolving deltas: 100% (12365/12365), done.
Updating files: 100% (8486/8486), done.

client@PREDATOR MINGW64 ~
$ |
```

Normalement, la branche *master* contient du code à jour et compilable. C'est le seul que je suggère de cloner. D'autres branches sont :

- *production* : corrections en cours et petits changements, qui seront appliqués à la branche *master*. C'est un travail en cours, alors ne vous attendez pas à ce qu'il se laisse compiler tout le temps.
- *développement* : travail en cours pour la prochaine version de JSesh
- *jfx-test* : quelques travaux pour voir comment adapter JSesh à Java FX.

Compilation du code

C'est par la commande :

```
mvn install
```

Avant cela, vous pouvez choisir la version de JSesh que vous souhaitez compiler. Diverses versions sont référencées. Alors, vous pourriez faire

quelque chose comme

```
git checkout version-5.3
mvn clean install
```

Pour compiler JSesh v. 5.3, tous les tags peuvent être répertoriés en tapant

```
git tag
```

Ensuite, vous pouvez utiliser les bibliothèques JSesh dans votre programme en les référençant dans votre fichier *pom.xml*. Par exemple:

```
<dependency>
  <groupId>org.qenherkhopeshef</groupId>
  <artifactId>jseshGlyphs</artifactId>
  <version>5.3</version>
</dependency>
<dependency>
  <groupId>org.qenherkhopeshef</groupId>
  <artifactId>jsesh</artifactId>
  <version>5.3</version>
</dependency>
<dependency>
  <groupId>org.qenherkhopeshef</groupId>
  <artifactId>qenherkhopeshefUtils</artifactId>
  <version>5.3</version>
</dependency>
```

Notez qu'à partir de JSesh 6.7, le *groupId* sera remplacé par

```
<groupId>org.qenherkhopeshef.jsesh</groupId>
```

afin de simplifier la gestion des dépôts *maven* (je veux pouvoir supprimer facilement toutes les anciennes versions de JSesh avec un simple *rm* sur mon ordinateur).

Si vous voulez *exécuter* JSesh, le module est *jseshAppli*. La dernière version de *jsesh-installer* fournit deux dossiers, un pour Mac et un pour Windows, avec des distributions presque prêtes - la fin de la construction de la production est actuellement manuelle, voir le README.md à la racine du projet JSesh.

Comment faire

Pour ajouter un champ d'édition hiéroglyphique dans une interface SWING

En fait, c'est assez facile à faire. Vous devez avoir *jsesh.jar* dans votre chemin de classe, et probablement aussi *jseshGlyphs.jar* si vous voulez les polices complètes. Ensuite, avoir un champ hiéroglyphique dans votre application est aussi simple que ça :

```
// Le package peut changer un jour dans un (lointain) avenir.
import package jsesh.mdcDisplayer.swing.editor.*;
public MyClass .... {
    void buildInterface() {
```

```

    // A Large editor, better placed in a JScrollPane
    JMDCEditor editor= new JMDCEditor();
    // A TextField-like editor
    JMDCField mdcField= new JMDCField();
}
}

```

Maintenant, vous pouvez manipuler le texte directement via la classe *HieroglyphicTextModel*, qui représente le texte sous forme de liste d'objets, ou, si vous n'avez besoin que de fonctionnalités simples, utilisez les méthodes *setMDCText(String mdc)* et *getMDCText()* pour définir et récupérer le contenu du champ selon le Manuel de codage.

Vous pouvez interdire l'édition du texte avec *setEditable(false)*.

Une chose qui manque actuellement dans les bibliothèques (mais qui devrait être corrigée bientôt) est un moyen de diriger facilement les informations de la palette vers divers widgets JSesh (cela peut être fait avec la structure globale, mais ce n'est pas automatique du tout).

Pour produire une image bitmap à partir d'un texte MDC

Il existe un certain nombre de raisons pour lesquelles vous pouvez souhaiter produire une image à partir d'un texte MDC. Par exemple, vous pouvez utiliser JSesh comme bibliothèque dans une application Web.

Bien sûr, vous devez avoir à la fois *jsesh.jar* et *jseshGlyphs.jar* dans votre chemin de classe. A partir de JSesh 2.13.7, vous avez également besoin de *jvectClipboard-1.0.jar*, mais c'est une dépendance que je supprimerai dans peu de temps. Vous aurez besoin de *jvectClipboard-1.0.jar* si vous souhaitez produire du SVG, WMF ou similaire. Ensuite, le code Java suivant fera l'affaire¹⁵ :

```

public static BufferedImage buildImage(String mdcText) throws MDCSyntaxError
{
    // Create the drawing system:
    MDCDrawingFacade drawing = new MDCDrawingFacade ();
    // Create the picture
    BufferedImage result = drawing.createImage(mdcText);
    return result
}

```

C'est tout. Une fois que vous avez une *BufferedImage*, elle peut être affichée à l'écran ou écrite en JPEG ou PNG en utilisant *ImageIO*.

En option, il est possible de personnaliser le rendu. Voici un exemple complet prêt à l'emploi :

```

/**
 * How to use JSesh to create bitmaps in Java.
 * compile: javac -cp ./FOLDER_CONTAINING/jsesh.jar TestJSeshBitmap.java
 * run: java -cp ./FOLDER_CONTAINING/jsesh.jar TestJSeshBitmap

```

¹⁵ contrairement aux Datari républicaines :

https://www.youtube.com/watch?v=Wp872x3_vc0 (Note du traducteur).

```

*
* jseshGlyphs.jar and jvectClipboard-1.0.jar should be in the same folder as
jsesh.jar.
* (normally, there is no need to add them explicately to the class path , as
jsesh.jar contains the necessary
* information in its manifest.
*/

import javax.imageio.ImageIO;
import java.io.*;
import java.awt.image.* ;
import jsesh.mdcDisplayer.preferences.*;
import jsesh.mdcDisplayer.draw.*;
import jsesh.mdc.*;

public class Test {
    public static BufferedImage buildImage(String mdcText) throws
MDCSyntaxError {
        // Create the drawing system:
        MDCDrawingFacade drawing = new MDCDrawingFacade();
        // Change the scale, choosing the cadrat height in pixels.
        drawing.setCadratHeight(60);
        // Change a number of parameters
        DrawingSpecification drawingSpecifications = new
DrawingSpecificationsImplementation();
        PageLayout pageLayout= new PageLayout();
        pageLayout.setLeftMargin(5);
        pageLayout.setTopMargin(5);
        drawingSpecifications.setPageLayout(pageLayout);
        drawing.setDrawingSpecifications(drawingSpecifications);
        // Create the picture
        BufferedImage result = drawing.createImage(mdcText);
        return result;
    }

    public static void main(String args[]) throws MDCSyntaxError, IOException {
        // Create the picture
        BufferedImage img= buildImage("i-w-r:a-C1-m-p*t:pt");
        File f = new File("example.png");
        // save it in png (better than jpeg in this case)
        ImageIO.write(img, "png", f);
    }
}

```

La taille des signes est contrôlée à l'aide de `drawing.setCadratHeight()`;