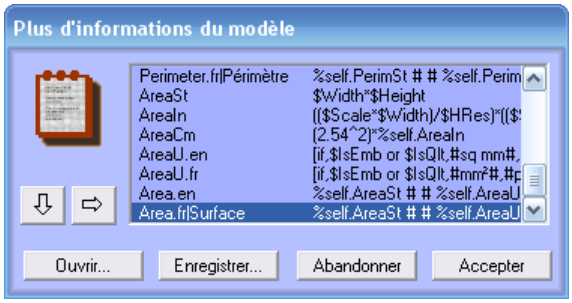
myriaCross editor – leçon 17 : Définir plus d'informations du modèle et évaluer des expressions mathématiques

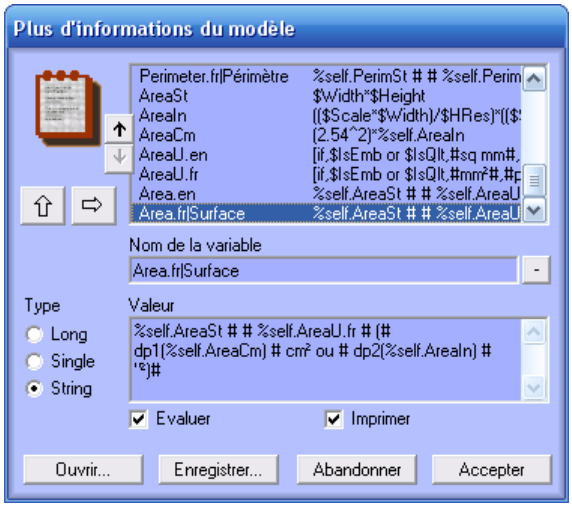
myriaCross editor version 1.51.01 et au dessus vous permet de définir vos propres informations du modèle grâce à des variables nommées dans des sections nommées ; ces données sont stockées dans votre modèle et peuvent apparaitre à l'impression des informations et des notes.
myriaCross editor version 1.51.01 et au dessus a aussi la capacité d'évaluer des expressions mathématiques définies par des nombres entiers ou réels, des opérateurs, des fonctions, des textes, des variables retournant des valeurs spécifiques au modèle et les valeurs de variables nommées.

Ajouter des données d'information du modèle :

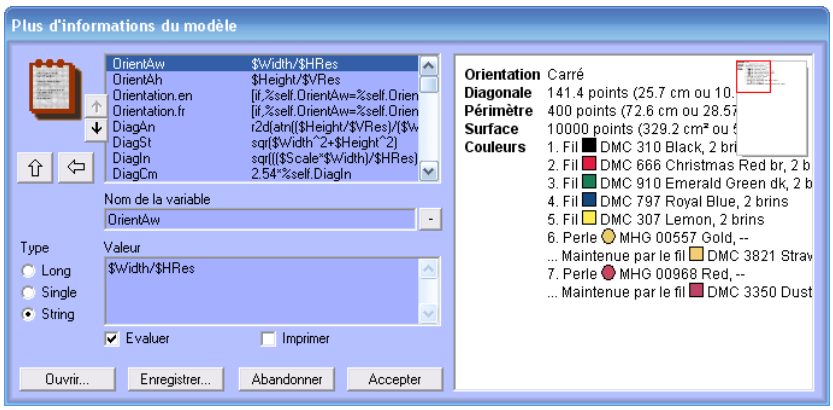
Lancez le dialogue à partir du menu *Edition/Plus d'informations du modèle* ou avec un clic droit sur l'icône  .



La liste montre les variables définies avec leur valeur associée. Evidemment, un nouveau modèle ne montre aucune variable tant que vous n'en avez pas défini. Cliquez la **flèche en bas** pour agrandir la fenêtre et commencer l'édition. Cliquer sur la **flèche en haut** terminera l'édition et réduira la fenêtre.



Les petites flèches vous permettent de changer l'ordre de la variable sélectionnée en la déplaçant vers le haut ou vers le bas.
Pour **créer** une variable, entrez son ***nom***, choisissez le ***type*** de donnée (long, single ou string) puis entrez une ***valeur***. Cochez l'option ***Evaluer*** pour les variables *string* dont le contenu doit être évalué à l'utilisation. Cochez l'option ***Imprimer*** pour inclure la variable courante dans l'impression des *Informations et des notes*. **Créez vraiment** votre variable avec le bouton ***+ (plus)*** ou **supprimez** la variable sélectionnée avec le bouton ***– (moins)***.



Cliquez **la flèche à droite** pour un aperçu de vos variables comme elles seront imprimées ; la **flèche à gauche** cachera l'aperçu.
Apparaissent ici seulement les variables déclarées imprimables.
Cliquez le bouton ***Abandonner*** pour ignorer toutes les modifications ou cliquez le bouton ***Accepter*** pour les garder. Cliquez le bouton ***Enregistrer*** pour sauver vos informations dans un fichier. Cliquez le bouton ***Ouvrir*** pour charger les informations à partir d'un fichier, comme nouvelles informations ou ajoutées aux existantes. Comme les fichiers d'information sont dans un format lisible, vous pouvez

définir vos informations dans un fichier et les charger d'un seul coup (voir l'appendice C ci-dessous).

Note : changer la valeur, le type ou les options (ou le nom local – voir plus loin) d'une variable sélectionnée la met à jour automatiquement. Entrer un nom correspondant à une variable existante la sélectionne automatiquement dans la liste. Les noms ne peuvent pas contenir les caractères & # @ $ < > = ( ) + - * ^ / \ [ ] { } , ; : ! ou ?

Titre	Daisies
Auteur	Ellen Maurer-Stroh
Droits d'auteur	2000 Ellen Maurer-Stroh
Taille	84 x 79 points (15.2 x 14.3 cm ou 6 x 5.6
Toile	14 count (5.5 points/cm) (14ct #357 Aida White)
1 point	1.81 x 1.81 mm ou 0.071 x 0.071 "
Notes	Full Stitches - use 3 strands of floss Backstitches - use 1 strand of floss
Orientation	Paysage
Diagonale	115.3 points (20.9 cm ou 8.24 ") à 43.2°
Périmètre	326 points (59.1 cm ou 23.29 ")
Surface	6636 points (218.4 cm² ou 33.86 "²)

Voici un exemple d'impression ; en plus de l'information standard, apparaissent maintenant les variables que vous avez déclarées imprimables dans le dialogue ci-dessus. Comme vous pouvez le voir, cette fonctionnalité vous permet d'imprimer presque tout ce dont vous avez besoin.

Appendice A. Enregistrer des données étendues nommées

Structure des données nommées :

Section1 *Variable1* ➔ *Valeur*
Variable2 ➔ *Valeur*
…
Section2 *Variable1* ➔ *Valeur*
Variable2 ➔ *Valeur*
…

Note : les données de vos informations sont stockées dans la section *MoreInfo* ; vous ne pouvez pas encore définir d'autres sections ; les prochaines versions pourraient vous permettre de le faire de façon à gérer de nouvelles fonctionnalités.

Types de données disponibles pour les variables :

Vous pouvez définir les variables comme entiers longs, réels en simple précision ou chaines de caractères. Les entiers longs sont dans la plage -2 147 483 648 à 2 147 483 647 alors que les réels sont dans la plage -3.402823e38 à -1.401298e-45 pour les valeurs négatives et 1.401298e-45 à 3.402823e38 pour les valeurs positives. Les variables chaine peuvent contenir des constantes, des variables ou des expressions mathématiques que le programme évaluera.

Un exemple simple d'informations du modèle :

```
MoreInfo
 MyLong    =    1234
 MySingle  =    17.99
 MyString  =    "PAT001"
```

Un exemple avancé d'informations du modèle :

```
MoreInfo
 AreaStitches =    "$Width*$Height"
 AreaUS       =    "($Width/$HRes)*($Height/$VRes)"
 AreaMetric   =    "(2.54^2)*%self.AreaUS"
 Display      =    "#Area = # %self.AreaStitches # stitches#
                    #Area US = # dp3(%self.AreaUS) # sq in#
```

#Area Metric = # dp2(%self.AreaMetric) # cm²#"

Si le modèle courant est de 40 x 50 points à 14 points par pouce et que vous avez déclaré seulement la variable *Display* comme imprimable, les notes et informations imprimées devraient également montrer :

Display Area = 2000 stitches
Area US = 10.204 sq in
Area Metric = 65.83 cm²

Note : l'expression *%self* fait référence à la section courante ; par exemple, *%self.AreaUS* est évalué comme *MoreInfo.AreaUS.*

Appendice B. Evaluer des expressions mathématiques

Opérateurs disponibles :

xor	Ou exclusif (bits)	(5 xor 2 returns 7	7 xor 2 retourne 5)
	Ou exclusif logique	(0 xor 0 = 0	0 xor -1 = -1
		-1 xor 0 = -1	-1 xor -1 = 0)
or	Ou (bits)	(5 or 2 retourne 7)	7 or 2 retourne 7)
	Ou logique	(0 or 0 = 0	0 or -1 = -1
		-1 or 0 = -1	-1 or -1 = -1)
and	Et (bits)	(5 and 2 retourne 0)	
	Et logique	(0 and 0 = 0	0 and -1 = 0
		-1 and 0 = 0	-1 and -1 = -1)
>=	Supérieur ou égal à	(5 >= 2 retourne -1	2 >= 5 retourne 0)
<=	Inférieur ou égal à	(5 <= 2 retourne 0	2 <= 5 retourne -1)
>	Supérieur à	(5 > 2 retourne -1	2 > 5 retourne 0)
<	Inférieur à	(5 < 2 retourne 0	2 < 5 retourne -1)
<>	Différent de	(5 <> 2 retourne -1	5 <> 5 retourne 0)
=	Egal à	(5 = 2 retourne 0	5=5 retourne -1)
+	Ajouter	(5 + 2 = 7)	
-	Soustraire	(5 – 2 = 3)	
mod	Modulo	(5 mod 2 = 1, reste de la division entière)	
\	Division entière	(5 \ 2 = 2, partie entière de la division)	
*	Multiplier	(5 * 2 = 10)	
/	Diviser	(5 / 2 = 2.5)	
^	Puissance	(5 ^ 2 = 5 * 5 = 25)	

Note : les operateurs *xor*, *or*, *and*, *mod* **doivent** être entourés d'espaces.

Fonctions disponibles :

abs	Valeur absolue	(abs(2.3) = 2.3	abs(-2.3) = 2.3)
int	Partie entière	(int(2.3) = 2	int(-2.3) = -3)
fix	Partie entière	(fix(2.3) = 2	fix(-2.3) = -2)
sgn	Signe	(sgn(2.3) = 1	sgn(0) = 0 sgn(-2.3) = -1)
sqr	Racine carrée	(sqr(3) = 1.732)	
exp	Exponentielle (base e)	(exp(1) = 2.718)	
log	Logarithme (base e)	(log(1) = 0	log(2.7182818) = 1)
log10	Logarithme (base 10)	(log10(5) = 0.698	log10(50) = 1.698)
rnd	Valeur aléatoire	(rnd(any) = réel entre 0 et 1)	
sin	Sinus	(sin(pi/6) = 0.5)	

cos	Cosinus	(cos(pi/6) = 0.866)
tan	Tangente	(tan(pi/6) = 0.577)
asn	Arc sinus	(asn(0.5) = pi/6)
acs	Arc cosinus	(acs(0.866) = pi/6)
atn	Arc tangente	(atn(0.5776) = pi/6)
not	Not logique/bits	(not(-1) = 0 not(0) = -1)

Note : les fonctions *sin*, *cos*, *tan* requièrent un argument en radians ; les fonctions *asn*, *acs*, *atn* retournent un angle en radians ; ces fonctions convertissent entre radians et degrés :

d2r	Degrés vers radians	(d2r(180) = 3.1415926)
r2d	Radians vers degrés	(r2d(3.1415926) = 180)

Fonctions de formatage disponibles (conversion en chaine) :

dp0	Pas de décimale	(dp0(0.5776) = "1")
dp1~dp8	1 à 8 décimales	(dp1(0.5776) = "0.6", (dp2(0.5776) = "0.58")
quote	Ajouter des marques	(quote(0.5) = "«0.5»", quote("ab") = "«ab»" quote("5 + 1") = "«6»")
a2s	Convertir en chaine	(a2s(10) = "10", a2s(1,2) = 1.2, a2s("ab") = "ab")

Note : vous ne pouvez pas mélanger des fonctions de formatage et des opérateurs ; par exemple, l'expression *dp1(3.25)*3* retourne 0 alors que *dp1(3.25*3)* retourne 9.8, ce qui est correct.

Spécifier des textes :

Utilisez la syntaxe *#TextString#* ; si vous voulez le caractère # dans un texte, spécifiez &#.
Exemple 1: *#Le résultat est # 5+3 # pouces#* retourne *Le résultat est 8 pouces*.
Exemple 2 : *#Le résultat &#1 est # 5+3 # pouces#* retourne *Le résultat #1 est 8 pouces*.
Vous pouvez aussi définir des textes spéciaux qui s'impriment comme des images :

 Utilisez la syntaxe *!Box:<couleur RGB>* pour créer un rectangle coloré.
 Utilisez la syntaxe *!Dot:<couleur RGB>* pour créer un rond coloré.
 Utilisez la syntaxe *!Pnt:<couleur RGB>* pour créer un point coloré.
 Utilisez la syntaxe *!Lin:<couleur RGB>* pour créer une ligne colorée.
 Utilisez la syntaxe *!Arr:<couleur RGB>* pour créer une flèche colorée.

Fonctions complexes disponibles :

[If, <condition>, <expression si vrai>, <expression si faux>]

Evalue *<condition>* puis retourne *<expression si vrai>* évaluée si la condition retourne *True* ou toute valeur sauf 0, ou retourne *<expression si faux>* si la condition retourne *False* ou 0.
Exemple : [if, X>5, Y+1, Y-1]
Note : les expressions ne doivent pas contenir de virgule sous peine de résultats imprévisibles; si vous avez besoin de virgules à l'intérieur d'une expression texte, faites référence à une variable qui les contient.

[Choose, <condition>, <expression si 1>, <expression si 2>, …, <expression si n>]

Evalue *<condition>* puis retourne l'expression évaluée dont la position est *<condition>* évaluée. Si inférieur à 1 ou supérieur au nombre d'expressions spécifiées, retourne 0.
Note : même remarque à propos de l'utilisation de virgules.

[Repeat, <variable>, <min>, <max>, <expression>]

Répète l'évaluation de *<expression>* pour *<variable>* dont la valeur varie entre *<min>* et *<max>* par pas de 1 ou -1 (déduit automatiquement des valeurs de *<min>* et *<max>*).
La variable doit être *%0*, *%1*, *%2*, …, ou *%9* ; cette variable peut être utilisée dans *<expression>* et même dans les variables appelées par *<expression>*.
Exemples :
*X = %1 * 10 # #* et *Y = [Repeat, %1, 1, 5, %self.X]* est évalué comme *10 20 30 40 50.*
*X = %1 * 2 # #* et *Y = [Repeat, %1, 5, 1, %self.X]* est évalué comme *10 8 6 4 2.*
*Y = [Repeat, %1, 1, 5, %1*2 # #]* est évalué comme *2 4 6 8 10.*
Note : même remarque à propos de l'utilisation de virgules. Vous pouvez aussi utiliser *%1p* qui est évalué comme *%1+1* ou *%1m* qui est évalué comme *%1-1* dans les variables appelées par *<expression>* mais pas dans *<expression>* elle-même.

[ExitRepeat]

Quitte la boucle *Repeat* la plus récente et retourne une chaine vide ou 0 suivant l'expression appelante. S'il n'y a pas de boucle en fonctionnement, la fonction ne fait rien.
Exemple :
X = [If, %1 > 5, [ExitRepeat], 10%1 # #]* et *Y = [Repeat, %1, 1, 15, %self.X]* sont évalués comme *10 20 30 40 50* ; les cas où *%1* égale 6 à 15 ne sont pas évalués du tout.

[DefArray, %<type><numéro>, <min>, <max> (, %<type><numéro>, <min>, <max>)]

Définit un tableau de valeurs de type *Lng* (pour *Long*), *Sng* (pour *Single*) ou *Str* (pour *String*). Le numéro est entre 0 et 3 (vous pouvez définir jusqu'à 4 tableaux de chaque type). Les expressions de position *min* et *max* sont évaluées pour définir le tableau. Vous pouvez définir plus d'un tableau à la fois. Ne retourne rien.
Exemple : *X = [DefArray, %Lng0, 1, 10]* définit un tableau de valeurs *long* aux positions 1 à 10. *Y = [DefArray, %Lng0, 1, %self.Size]* définit un tableau avec la position maximum calculée.

[SetArray, %<type><numéro>, <position>, <valeur> (, %<type><numéro>, <position>, <valeur>)]

Stocke *<valeur>* évaluée dans le tableau et à la *<position>* spécifiés. Vous pouvez stocker plus d'une valeur à la fois. Ne retourne rien.

Constantes disponibles :

Pi	Single	3.1415926
EmptyStr	String	"" (chaine vide)

Variables du modèle disponibles :

$Title	String	Titre du modèle courant
$Author	String	Auteur du modèle courant
$Copyright	String	Message de copyright du modèle courant
$Company	String	Société de l'auteur du modèle courant
$Website	String	Site de l'auteur du modèle courant
$Email	String	Courriel du modèle courant
$Width	Long	Largeur du modèle courant en points
$Height	Long	Hauteur du modèle courant en points
$HRes	Long	Résolution horizontale du modèle courant en points par pouce
$VRes	Long	Résolution verticale du modèle courant en points par pouce
$Scale	Single	Facteur d'échelle (broderie ou quilting : 10, autre :1)

$IsCrs	Long	Le modèle courant est du point de croix (0=faux, -1=vrai)		
$IsEmb	Long	Le modèle courant est de la broderie (0=faux, -1=vrai)		
$IsQlt	Long	Le modèle courant est du quilting (0=faux, -1=vrai)		
$IsKnt	Long	Le modèle courant est du tricot (0=faux, -1=vrai)		
$IsLhk	Long	Le modèle courant est du latch hook (0=faux, -1=vrai)		
$WhatIs	Long	Type de contenu du modèle courant (1=point de croix, 2=broderie, 3=quilting, 4=tricot, 5=latch hook)		
$WhatIsEx	Long	Mêmes valeurs plus 6=scrapbook s'il n'y a que des charms		
$EdTime	String	Temps total d'édition du modèle courant.		
$PropPal:i;j;k		Propriété de la palette du modèle courant		
	Long	Nombre d'entrées dans la palette	0; 0; 0	
	Long	Nombre de sous entrées	ME; 0 ; 1	
	Long	Numéro du symbole	ME; 0 ; 2	
	String	Police du symbole	ME; 0 ; 3	(exemple : Arial)
	Long	La sous entrée est définie	ME; SE; 0	(0= faux, -1= vrai)
	Long	Type de sous entrée	ME; SE; 1	(0=Fil, 1=Perle)
	String	Clé de la gamme de fil	ME; SE; 2	(exemple : DMC)
	String	Information sur la gamme de fil	ME; SE; 3	(exemple : DMC)
	String	Numéro de la couleur	ME; SE; 4	(exemple : 310)
	String	Description de la couleur	ME; SE; 5	(exemple : Black)
	Long	Valeur RGB de la couleur	ME; SE; 6	
	Long	Nombre de brins	ME; SE; 7	(0 to 6)

Note : Le nom de la variable est *$PropPal:* (2 points à la fin sans espace). *i*, *j* et *k* sont des paramètres séparés par des points-virgules ; ils peuvent être *@section.variable*, *%self.variable* ou *%0* à *%9*. *ME* signifie le numéro de l'entrée de (la position dans) la palette (1 pour la 1ère couleur, 2 pour la 2ème et ainsi de suite). *SE* signifie la sous entrée dans la palette (entre 1 et 4). Utiliser la fonction complexe *Repeat* vous permet de parcourir toute la palette.

Exemples :

$PropPal: 0; 0; 0 retourne le nombre de couleurs dans la palette (nombre d'entrées). *$PropPal: 5; 0; 1* retourne le nombre de sous entrées définies dans la 5ème entrée de la palette. *$PropPal: 3; 2; 4* retourne le numéro de couleur de la 2ème sous entrée de la 3ème entrée de la palette.

$PropCS:col;row		Propriété d'un point de croix du modèle courant	
	Long	Pointeur vers la table de types de point de croix	

Note : Le nom de la variable est *$PropCS:* (2 points à la fin sans espace). *col* et *row* sont des paramètres séparés par des points-virgules ; ils peuvent être *@section.variable*, *%self.variable* ou *%0* à *%9*. Utiliser la fonction complexe *Repeat* vous permet de parcourir tout le modèle.

$PropCT:i;j		Propriété d'un type de point de croix du modèle courant	
	Long	Nombre d'entrées dans la table	0; 0
	Long	Pointeur vers la palette du ¼ de point haut gauche (*TLQ*)	TE; 1
	Long	Pointeur vers la palette du ¼ de point haut droit (*TRQ*)	TE; 2
	Long	Pointeur vers la palette du ¼ de point bas gauche (*BLQ*)	TE; 3
	Long	Pointeur vers la palette du ¼ de point bas droit (*BRQ*)	TE; 4
	Long	Indicateur de petits points	TE; 5

Note : Le nom de la variable est *$PropCT:* (2 points à la fin sans espace). *i* et *j* sont des paramètres séparés par des points-virgules ; ils peuvent être *@section.variable*, *%self.variable* ou *%0* à *%9*. *TE* signifie le numéro de l'entrée dans la table (1 pour la 1ère entrée, 2 pour la 2ème entrée et ainsi de suite). L'indicateur de petits points est codé au niveau du bit (1 pour *TLQ*, 2 pour *TRQ*, 4 pour *BLQ*, 8 pour *BRQ*) ; exemple : 5 signifie que *TLQ* et *BLQ* sont des petits points alors que *TRQ* et *BRQ* sont des quarts de points. Utiliser la fonction complexe *Repeat* vous permet de parcourir toute la table.

$PropBS:i;j		Propriété d'une piqûre du modèle courant	
	Long	Nombre d'entrées dans la table	0; 0
	Long	Pointeur vers la palette	TE; 0
	Long	X de départ	TE; 1
	Long	Y de départ	TE; 2
	Long	X de fin	TE; 3
	Long	Y de fin	TE; 4
$PropFK:i;j		Propriété d'un nœud du modèle courant	
	Long	Nombre d'entrées dans la table	0; 0
	Long	Pointeur vers la palette	TE; 0
	Long	X	TE; 1
	Long	Y	TE; 2
$PropBD:i;j		Propriété d'une perle du modèle courant	
	Long	Nombre d'entrées dans la table	0; 0
	Long	Pointeur vers la palette	TE; 0
	Long	X	TE; 1
	Long	Y	TE; 2
$PropES:i;j		Propriété d'un point de broderie du modèle	
	Long	Nombre d'entrées dans la table	0; 0
	Long	Pointeur vers la palette	TE; 0
	Long	X	TE; 1
	Long	Y	TE; 2
$PropQS:i;j		Propriété d'un point de quilting du modèle courant	
	Long	Nombre d'entrées dans la table	0; 0
	Long	X	TE; 1
	Long	Y	TE; 2
$PropObj:i;j		Propriété d'un objet du modèle courant	
	Long	Nombre d'objets	0; 0
	Long	Type d'objet	TE; 0
	Long	Gauche	TE; 1
	Long	Haut	TE; 2
	Long	Largeur	TE; 3
	Long	Hauteur	TE; 4
	String	Nom (Nom d'article du fournisseur)	TE; 5
	String	Référence (Numéro d'article du fournisseur)	TE; 6
	String	Description (Type d'article du fournisseur)	TE; 7
	String	Nom du fournisseur	TE; 8
$LngStr:ID;i;j;k;l		Chaine de caractères provenant du fichier d'aide courant	

Note : *ID* est obligatoire, c'est le numéro d'identification de la chaine ; *i, j, k* et *l* sont les paramètres optionnels *%1*, *%2* ; *%3* et *%4* respectivement.

myriaCross editor – leçon 17 : Définir plus d'informations du modèle et évaluer des expressions mathématiques

Variables de fonctionnement disponibles :

%0 à *%9*	Long ou Single	Variable d'une fonction *Repeat*
%Lng<numéro>:<position>	Long	Elément d'un tableau de valeurs *Long*
%Sng<numéro>:<position>	Long	Elément d'un tableau de valeurs *Single*
%Str<numéro>:<position>	Long	Elément d'un tableau de valeurs *String*

Variables nommées (données étendues) :

La syntaxe générale est *@NomDeSection.NomDeVariable mais vous pouvez utiliser %self.NomDeVariable* pour faire référence à la section courante ; par exemple, *%self.Area* est évalué comme *MoreInfo.Area.* Les versions futures pourraient contenir plus de sections pour des fonctionnalités supplémentaires ; à ce moment là, la syntaxe générale sera utile. Si la variable n'est pas trouvée dans la section spécifiée, elle sera évaluée à zéro.

Plus d'exemples :

1+2	retourne 3
*(1+2)*3*	retourne 9
Sqr(3)	retourne 1.7320508
Sqr(2+1)	retourne 1.7320508
(2.5+1.5)(3-1)*	retourne 8
*2.54*$Width/$HRes*	retourne la largeur du modèle en centimètres
2@Size.Width*	retourne 2 fois la valeur stockée dans la section *Size*, variable *Width*.

Appendice C. Format des fichiers d'information (.mcmia)

Codage :

Déclaration	***Signification***
myriaCross editor	*Identifiant*
Section=MoreInfo	*Nom de la section, toujours MoreInfo*
Version=1	*Version du format*
Count=<nombre-de-variables>	*Nombre de définitions de variables qui suivent*
Pour *Count* variables :	
Lng:<nom-variable>=<valeur>	*Long*
Ou Lng/P:<nom-variable>=<valeur>	*Long Imprimable*
Ou Sng:<nom-variable>=<valeur>	*Single*
Ou Sng/P:<nom-variable>=<valeur>	*Single Imprimable*
Ou Str:<nom-variable>=<valeur>	*String*
Ou Str/E:<nom-variable>=<valeur>	*String Doit être évalué*
Ou Str/P:<nom-variable>=<valeur>	*String Imprimable*
Ou Str/EP:<nom-variable>=<valeur>	*String Imprimable Doit être évalué*

Note : Les variables *long* et *single* n'ont pas besoin d'être évaluées, ainsi, les expressions Lng/E et Sng/E ne feront rien de plus bien qu'elles soient reconnues par le programme. Pour les valeurs *string* multilignes, utilisez la séquence \0D\0A pour indiquer une nouvelle ligne ; exemple : *Str:MyString=Voici\0D\0Aune chaine\0D\0Amultiligne*. Vous pouvez ajouter des lignes de remarque mais seulement après la ligne de déclaration *Count* ; pour ce faire, utilisez ' comme premier caractère ; exemple : *'Ligne de commentaire*

myriaCross editor – leçon 17 : Définir plus d'informations du modèle et évaluer des expressions mathématiques

Localiser vos variables (en fonction de la langue) :

Le programme imprime toutes les variables que vous avez déclarées imprimables en utilisant le nom de variable comme nom d'information. Si vous voulez que vos variables soient imprimables dans plus d'une langue, voici comment faire :

Considérez cette variable :

Str/EP:Area=$Width*$Height # stitches#

Pour l'imprimer en Anglais comme en Français, définissez 2 variables ainsi :

Str/EP:Area.en=$Width*$Height # stitches# (Variable Anglaise)

Str/EP:Area.fr|Surface=$Width*$Height # points# (Variable Française)

Toutes les variables imprimables sans langue spécifiée sont ajoutées à l'impression.

Les variables dont la langue spécifiée est celle de l'interface sont ajoutées si elles existent; sinon, les variables dont la langue spécifiée est celle de remplacement, comme indiqué dans le fichier de langue, sont ajoutées. Les variables pour d'autres langues sont ignorées.

Note : vous pouvez aussi utiliser la syntaxe *NomDeVariable|NomImprimé* avec des variables non localisées.

Exemple de fichier :

myriaCross editor
Section=MoreInfo
Version=1
Count=4
Str/E:AreaSt=$Width*$Height
Str/E:AreaUS=($Width/$HRes)($Height/$VRes)*
Str/E:AreaMetric=(2.54^2)%self.AreaUS*
Str/EP:Area=%self.AreaSt # stitches (# dp2(%self.AreaUS) # sq in or # dp1(%self.AreaMetric) # sq cm)#

Fichiers d'information prédéfinis :

Nom de fichier *(dans Mes documents\myriaCross editor\MoreInfo)*	Variable Anglaise	Variable Française
en-fr-Property-Area	*Area*	*Surface*
en-fr-Property-Diagonal	*Diagonal*	*Diagonale*
en-fr-Property-Orientation	*Orientation*	*Orientation*
en-fr-Property-Perimeter	*Perimeter*	*Périmètre*
en-fr-Properties-Orientation-Diagonal-Perimeter-Area	*Toutes*	*Toutes*
en-fr-Property-Objects List	*Objects*	*Objets*
en-fr-Property-Stitch Types	*Types* *Stitches*	*Types* *Points*
br-de-en-fr-nl-Property-Edit Time	*Edited for*	*Edité durant*
br-de-en-fr-nl-Property-Content	*Content*	*Contenu*
br-de-en-fr-nl-Property-Colour	*Colours*	*Couleurs*
br-de-en-fr-nl-Properties-Backstitches-FrenchKnots-Beads	*Backstitches* *French knots* *Beads*	*Piqûres* *Nœuds* *Perles*
br-de-en-fr-nl-Property-Alternate Sizes	*Sizes*	*Tailles*

myriaCross editor – leçon 17 : Définir plus d'informations du modèle et évaluer des expressions mathématiques

Surface *affiche la surface du modèle en points, en centimètres carrés et en pouces carrés.*
Diagonale *affiche la taille de la diagonale en points, en centimètres et en pouces, plus l'angle en degrés.*
Orientation *affiche « Portrait », « Paysage » ou « Carré ».*
Périmètre *affiche le périmètre du modèle en points, en centimètres et en pouces.*
Objets *affiche le coin, la taille et le nom de chaque objet du modèle.*
Types *affiche le nombre de combinaisons de quarts de points des croix.*
Points *affiche les combinaisons de quarts de points.*
Edité durant *affiche le temps total d'édition du modèle.*
Contenu *affiche « Point de croix », « Broderie », « Quilting », « Tricot » ou « Latch Hook ».*
Couleurs *affiche la liste des couleurs de la palette.*
Piqûres, ***Nœuds*** et ***Perles*** *affichent le nombre respectif d'éléments par couleur.*
Tailles *affiche des tailles alternatives pour le modèle de point de croix courant.*

Un fichier prédéfini expliqué :

Nous allons maintenant expliquer le contenu du fichier prédéfini ***en-fr-Property-Area*** :

```
1 myriaCross editor
2 Section=MoreInfo
3 Version=1
4 Count=7
5 'Calculer les valeurs nécessaires
6 Str/E:AreaSt=$Width*$Height
7 Str/E:AreaIn=(($Scale*$Width)/$HRes)*(($Scale*$Height)/$VRes)
8 Str/E:AreaCm=(2.54^2)*%self.AreaIn
9 Str/E:AreaU.en=[if,$IsEmb or $IsQlt,#sq mm#,#stitches#]
10 Str/E:AreaU.fr=[if,$IsEmb or $IsQlt,#mm²#,#points#]
11 'Definir les variables imprimables
12 Str/EP:Area.en=%self.AreaSt # # %self.AreaU.en # (# dp2(%self.AreaIn) # sq in or #
dp1(%self.AreaCm) # sq cm)#
13 Str/EP:Area.fr|Surface=%self.AreaSt # # %self.AreaU.fr # (# dp1(%self.AreaCm) # cm² ou #
dp2(%self.AreaIn) # "²)#
```

La ligne 1 est l'identifiant du contenu du fichier ; entrer autre chose empêche le programme de charger le fichier.
La ligne 2 est le nom de la section des données ; entrer autre chose empêche le programme de charger le fichier.
La ligne 3 est la version du format de fichier ; entrer autre chose empêche le programme de charger le fichier ; les prochaines versions du programme pourraient améliorer le format et changer le numéro de version du format.
La ligne 4 est le nombre de variables dans le fichier ; les lignes de commentaires ne sont pas comptabilisées.
La ligne 5 est une ligne de commentaire ; elle rend le fichier un peu plus lisible ; ajoutez autant de lignes de commentaires que vous voulez mais pas avant la 5ème ligne. Le programme n'inclut pas de ligne de commentaire à l'enregistrement.
La ligne 6 calcule la surface du modèle en points :

$Width est le nombre de colonnes du modèle.
$Height est le nombre de lignes du modèle.

La ligne 7 calcule la surface réelle du modèle en pouces carrés :

$HRes est le nombre de points par pouce dans la direction des colonnes.
$VRes est le nombre de points par pouce dans la direction des lignes.
$Scale est un facteur d'échelle requis pour la compatibilité avec tous les types de contenu du modèle :

Point de croix, tricot, latch hook : 1 point par carré de grille.
Broderie, quilting : 10 points par carré de grille.

($Scale*$Width)/$HRes est la largeur du modèle en pouces.
($Scale*$Height)/$VRes est la hauteur du modèle en pouces.

La ligne 8 calcule la surface réelle du modèle en centimètres carrés :
Sachant qu'une longueur en cm est égale à 2.54 fois la longueur en pouces, 2.54^2 signifie 2.54*2.54 ; la largeur et la hauteur doivent être multipliés par 2.54
Pour éviter la répétition, la variable utilise le résultat de la variable *%self.AreaIn*, c'est à dire la variable *AreaIn* dans la même section.

Les lignes 9 et 10 définissent les unités à utiliser pour afficher la surface en points en fonction du contenu du modèle : broderie ou quilting en mm^2, sinon en points.

La ligne 9 définit les unités pour la version Anglaise.

La ligne 10 définit les unités pour la version Française.

La ligne 11 est une autre ligne de commentaire.

Les lignes 12 et 13 définissent les variables imprimables qui mélangent textes et nombres.

La ligne 12 définit la variable imprimable *Area* pour la version Anglaise.

%self.AreaSt	Nombre : surface en points
# #	Texte : 1 espace
%self.AreaU.en	Texte de l'unité pour la surface en points
# (#	Espace plus texte
dp2(%self.AreaIn)	Nombre : surface en pouces carrés avec 2 décimales
# sq in or #	Espace, texte d'unité, espace
dp1(%self.AreaCm)	Nombre : surface en cm^2 avec 1 décimale
# sq cm)#	Espace, texte d'unité

La ligne 13 définit la variable imprimable *Surface* pour la version Française.

%self.AreaSt	Nombre : surface en points
# #	Texte : 1 espace
%self.AreaU.fr	Texte de l'unité pour la surface en points
# (#	Espace plus texte
dp1(%self.AreaCm)	Nombre : surface en cm^2 avec 1 décimale
# cm² ou #	Espace, texte d'unité, espace
dp2(%self.AreaIn)	Nombre : surface en pouces carrés avec 2 décimales
# "²)#	Espace, texte d'unité

Sections nommées privées :

Notre jeu d'échecs et notre puzzle contiennent des sections privées pour gérer des comportements spécifiques. Les sections *Object()* sont éditables à partir du menu *Edition/Propriétés des objets* ; les autres sections ne sont pas éditables mais pourraient le devenir dans les versions futures du programme.

Object(1)	**Données spécifiques à l'objet principal (modèle)**
Lock	Verrous du modèle (voir ci-dessous)
V3dBi	Données de régions pour vue comme biscornu
V3dCu	Données de régions pour vue comme cube
V3dCy	Données de régions pour vue comme cylindre
V3dPe	Données de régions pour vue en perspective
V3dSp	Données de régions pour vue comme sphère

myriaCross editor – leçon 17 : Définir plus d'informations du modèle et évaluer des expressions mathématiques

Object(<handle>)	**Données spécifiques à l'objet spécifié par son *handle***
X, Y	Position de départ de l'objet
X1, Y1	Autre position de l'objet
Lock	Verrous de l'objet (voir ci-dessous)

ScreenButton<#>	**Définition d'un bouton dans le modèle**
Left, Top	Position
Width, Height	Taille
Picture	Image
Shape	Forme (voir ci-dessous)
Action	Action lors d'un clic (voir ci-dessous)
[X1, Y1, X2, Y2]	Etendue de l'action
Tooltip.<lang>	Texte d'info-bulle pour la langue *lang*

Valeur des bits des verrous :

1	Interdire l'édition de l'objet (ni barre d'outils, menu d'édition, menu contextuel) Désélectionner automatiquement l'objet quand le bouton de la souris est relâché
2	Interdire la sélection de l'objet (ni édition, ni déplacement)
4	Interdire la génération du contour de l'objet
8	Interdire le déplacement de l'objet
16	Interdire l'adoucissement du contour de l'objet
32	Interdire la génération de la lumière de l'objet

1 * 65536	Interdire le changement de taille du modèle
2 * 65536	Interdire la standardisation de la palette du modèle
Autre	Actuellement inutilisé

Codes de forme :

0	Rectangle
1	Rectangle arrondi
2	Ellipse
3	Gérer la couleur de transparence (0xFF00FF)
4	Ne garder que les zones non transparentes

Codes d'action :

1	Changer le verrou de tous les objets
10	Mémoriser la position de tous les objets dans X, Y
11	Mémoriser la position de tous les objets dans X1, Y1
20	Tourner les objets trouvés dans l'étendue X1, Y1, X2, Y2
30	Restaurer la position de tous les objets à partir de X, Y
31	Restaurer la position de tous les objets à partir de X1, Y1
40	Fixer aléatoirement la position de tous les objets en utilisant X, Y