DIDACTICIEL

Écrire son quatrième script

par Jean-Paul Verpeaux

Ce didacticiel utilise MyrScript, le langage permettant de piloter les fonctionnalités d'Harmony Assistant et inclus par défaut dans celui-ci.

Les scripts avec boite de dialogue sont intéressants parce qu'ils permettent à l'utilisateur d'exprimer ses choix ou prendre des décisions. Il y a une interaction possible entre le musicien et son script. Concrètement, il existe deux façons de créer un script avec boîte de dialogue. Soit c'est le script lui-même qui construit la boîte, via une suite d'instructions précises, soit c'est le concepteur du script qui la dessine en plaçant un à un les objets dont il a besoin dans une fenêtre de dialogue vierge.

Le script que nous allons étudier utilise la première méthode. Ce script sert à remplacer les notes des portées sélectionnées par leurs enharmoniques (par exemple, remplacer les Do# par des Ré bémol). En plus de sa vocation pédagogique, ce script a donc une réelle utilité puisqu'il permet de corriger des erreurs qui se produisent assez souvent lorsqu'on importe un midifile ou que l'on transpose une partition. Les fautes d'enharmonie n'ont aucune incidence sur l'écoute de la musique, elles sont purement théoriques et graphiques, mais en plus de leur effet disgracieux, elles perturbent la logique de lecture des partitions.

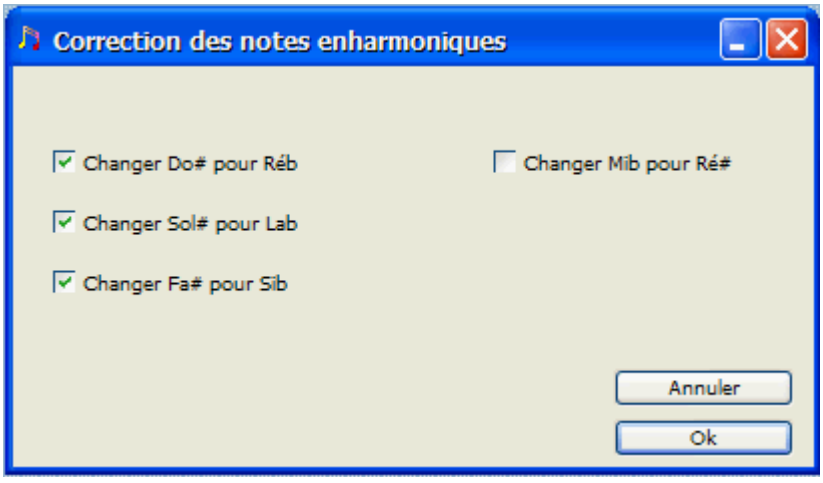
```
--COMMON SECTION
--CREATOR: VERPEAUX Jean-Paul
--DIFFUSION_MODE: 1
--VERSION: 1.1.1
--DATE-FR: Octobre 2007
--ABSTRACT-FR:Changement des altérations des notes au profit de leur écriture enharmonique
--NAME_IN_MENU: Correction des enharmoniques
Include "MSDefine"
Include "MSSounds"

myScore=FrontScore()
if myScore ~= nil then
	myScore.Preserve("Correction des enharmoniques")
	-- Creation de la boite de dialogue
	dialog=NewDialog("Correction des notes enharmoniques",400,200)
	if dialog then
		itemOk=dialog.NewPushButton("Ok",300,175)
		itemCancel=dialog.NewPushButton("Annuler",300,150)
		itemOk.IsDefault=true
		itemCancel.IsCancel=true
		item1=dialog.NewCheckBox("Changer Do# pour Réb",20,40)
		item2=dialog.NewCheckBox("Changer Sol# pour Lab",20,70)
		item3=dialog.NewCheckBox("Changer Fa# pour Solb",20,100)
		item4=dialog.NewCheckBox("Changer Mib pour Ré#",240,40)
		dialog.Show()
		-- attendre action
		while (item ~= itemOk and item ~= itemCancel) do
			item=dialog.Select()
		end
		if item==itemCancel then return end
	end -- if dialog

	myStaff=myScore.FirstSelectedStaff
	while myStaff~=nil and myStaff.IsSelected == true do
		symbol=myStaff.FirstSymbol
		while symbol do
			if symbol.IsNote== true then
				if item1.Value==true and mod (symbol.Pitch,12)==1 then
					symbol.Accidental=3
				end --if item 1
				if item2.Value==true and mod (symbol.Pitch,12)==8 then
					symbol.Accidental=3
```

```
                                    end --if item 2
                                    if item3.Value==true and mod (symbol.Pitch,12)==6 then
                                        symbol.Accidental=3
                                    end --if item 3
                                    if item4.Value==true and mod (symbol.Pitch,12)==3 then
                                        symbol.Accidental=1
                                    end --if item 4
                                end -- if symbol is note
                                symbol=symbol.Next
                            end -- while symbol
                            myStaff=myStaff.Next
                        end -- while
end – if
```

Voici un exemple concret.

Dans une partition en Fa majeur, un passage en si bémol mineur est assez fréquent. Pas question de changer d'armure pour une simple modulation passagère de deux mesures. Si le passage modulant concerne plusieurs portées, les corrections manuelles seraient fastidieuses. D'où l'intérêt d'un petit script.

Le Do# est illogique, ainsi que les Sol# et Fa# de la partie descendante de la gamme de Si bémol mineur.
Lançons le script.



Cochons les bonnes cases et cliquons sur OK.
Voici le résultat de l'opération.

Qu'en pensez-vous ?

Explications du script.

Le script étant un peu long, nous nous contenterons de ne commenter que les instructions que nous n'avons pas encore étudiées et, pour la même raison, ignorerons totalement l'entête.

myScore=FrontScore()	Attribuons un nom de variable à notre partition.
if myScore ~= nil then	Si le script ne trouve pas de partition les instructions suivantes seront ignorées. **nil**= inexistant, inconnu, non défini ...
myScore.Preserve("Correction des enharmoniques")	Petite précaution pour pouvoir annuler l'action du script.
-- Creation de la boite de dialogue	Cette première partie du script est consacrée à la création de la boîte de dialogue.
dialog=NewDialog("Correction des notes enharmoniques",400,200)	**NewDialog** élabore (en mémoire) une boîte de dialogue qui aura pour titre "Correction des notes enharmoniques" et pour dimensions 400 x 200. Si l'opération a réussi, l'objet **dialog** existe, on peut placer des boutons dedans, mais il restera invisible jusqu'à nouvel ordre.
if dialog then	Avant de travailler avec **dialog**, il faut s'assurer que myrScript a bien réussi à créer cet objet. **if dialog then** est equivalent à **if dialog ~= nil then.**
itemOk=dialog.NewPushButton("Ok",300,175)	Nous allons poser sur notre objet **dialog** un premier bouton. Il affichera le texte « OK » et sera placé aux coordonnées 300 (position horizontale) et 175 (verticale). Cet objet aura pour nom **itemOK**.
itemCancel=dialog.NewPushButton("Annuler",300,150)	Notre second bouton, placé au dessus du premier, affichera « Annuler » et répondra au nom de **itemCancel**.
itemOk.IsDefault=true	**ItemOK** sera le bouton « OK » par défaut, c'est à dire qu'il pourra être remplacé par l'appui sur la touche « Ok » du clavier.
itemCancel.IsCancel=true	De même on pourra annuler avec le bouton de même nom ou la touche ESC.
item1=dialog.NewCheckBox("Changer Do# pour Réb",20,40)	Maintenant nous allons placer des « boîtes à cocher ». Le premier objet de ce titre s'appellera **item1**, sera placé en X=20 et Y=40 et aura comme légende : Changer Do# pour Réb".
item2=dialog.NewCheckBox("Changer Sol# pour Lab",20,70)	Deuxième objet, un peu plus bas.
item3=dialog.NewCheckBox("Changer Fa# pour Solb",20,100)	Troisième objet.
item4=dialog.NewCheckBox("Changer Mib pour Ré#",240,40)	Quatrième objet, placé à la droite de **item1**.
dialog.Show()	Maintenant que les objets sont en place, nous pouvons afficher la boîte de dialogue. C'est le rôle de l'instruction **dialog.Show**().

-- attendre action	Puisque la boîte **dialog** est visible, il faut voir ce que l'utilisateur va faire avec. Nous allons guetter son action sur les boutons au moyen d'une boucle **while ... do**.
while (item ~= itemOk and item ~= itemCancel) do	Nous allons tester la valeur d'une variable **item** pour savoir si l'utilisateur a cliqué sur « Ok » ou sur « annuler ». Pour l'instant cette variable n'existe pas encore. Elle retourne comme valeur **Nil** qui signifie « variable non initialisée ». Comme **Nil** est forcement différent de ce qu'on attend, les conditions pour exécuter la boucle **while** sont réunies.
item=dialog.Select()	Nous déclarons ici la variable **item** en l'associant à la propriété **Select** de l'objet **dialog**. Elle nous retourne le nom du bouton sur lequel l'utilisateur a cliqué en dernier.
end	Fin de la boucle **while.**
if item==itemCancel then return end	Si on a cliqué sur Annuler, alors le script s'arrête ici.
end -- if dialog	On sort ici des instructions relatives à la création et l'exploitation de la boîte de dialogue et ce ne peut être que parce que l'utilisateur a choisi le bouton OK. Donc il va falloir regarder quelles cases ont été cochées et agir en conséquence.
myStaff=myScore.FirstSelectedStaff	Nous allons explorer une à une les portées sélectionnées. Pour cela nous avons besoin d'une variable. Ce sera **myStaff**.
while myStaff~=nil and myStaff.IsSelected == true do	Ici commence la boucle d'instructions pour passer d'une portée à la suivante. Disons que c'est la « grande boucle » ou « boucle extérieure » pour la différencier des boucles qui pourraient s'imbriquer.
symbol=myStaff.FirstSymbol	**symbol** est le nom choisi pour la variable qui va servir à lire séquentiellement les symboles d'une portée.
while symbol do	Cette boucle qui explore les symboles est imbriquée dans la grande boucle. Baptisons-la (mentalement) boucle intérieure ou petite boucle.
if symbol.IsNote== true then	Nous ne nous intéressons qu'aux notes, donc on teste la nature de chaque symbole. Dans le cas où ce ne serait pas une note, mais un silence par exemple, on ne fait rien.
if item1.Value==true and mod (symbol.Pitch,12)==1 then	**symbol** est une note. Reste à savoir si l'on souhaite « changer les Do# en Ré bémol » et si la note est un Do#. Pour savoir si la boîte « changer Do# pour Réb » à été cochée, nous testons la valeur (**Value**) de la variable **item1**. Pour savoir si la note est un do#, c'est un peu plus compliqué. Le numéro de la note (n° Midi) est donné par la propriété **Pitch** d'un objet **symbol**. Les notes sont numérotées de 0 à 127, la série commençant par un DO. La gamme chromatique comportant 12 notes, tous les multiples de 12 seront des DO. Les Do# (comme les Ré^{b}) seront des multiples de 12 + 1 comme 1, 13, 25, 37... Pour les identifier par calcul, il faut appliquer l'opération « modulo » qui retourne le reste d'une division par un nombre « n ». Nous allons calculer **« symbol.Pitch** modulo 12 » et si le résultat vaut 1, la note sera bien un Do#, quelque soit sont octave. La syntaxe exacte à employer est : (symbol.Pitch,12)==1.
symbol.Accidental=3	**Accidental** est la propriété de l'objet **symbol** qui identifie une altération appliquée à cet objet. Par définition 3 signifie « bémol». NATURAL=0 SHARP=1 AUTO=2 FLAT=3 DOUBLE_SHARP=4 DOUBLE_FLAT=5 Cette instruction remplace donc l'altération d'un Do# par un bémol.

	Le programme Harmony-assistant se charge tout seul de déplacer la note pour qu'elle soit sur la ligne ou l'interligne qui corresponde à un Ré. NOTA. Si on a pris le soin d'inclure MSDefine en début de script, nous pouvons remplacer symbol.Accidental=3 par symbol.Accidental=FLAT. Les deux instructions sont équivalentes.
end --if item 1	Fin du test de la première boîte à cocher.
if item2.Value==true and mod (symbol.Pitch,12)==8 then	Même chose pour la seconde boîte à côcher. Si le n° de **Pitch** est égal à 8, la note est un Sol# puisque Sol# est la 9ème note de la gamme chromatique.
symbol.Accidental=3	Forcer l'altération à être un bémol.
end --if item 2	Fin du test de la seconde boîte à cocher.
if item3.Value==true and mod (symbol.Pitch,12)==6 then	Un **Pitch** modulo 12 égal à 6 identifie un fa# (ou Sol^{b}).
symbol.Accidental=3	Transformons Fa# en Sol^{b}
end --if item 3	Fn du test
if item4.Value==true and mod (symbol.Pitch,12)==3 then	Voir si la quatrième note de la gamme (n°=3) est un Ré# ou Mi^{b}.
symbol.Accidental=1	Cette fois on force la note à s'afficher avec un bémol.
end --if item 4	
end -- if symbol is note	
symbol=symbol.Next	Passons au symbole suivant.
end -- while symbol	Fin de la petite boucle.
myStaff=myStaff.Next	Passage à la portée suivante.
end -- while	Fin de la grande boucle
end -- if	Sortie du script.

Travaux pratiques.

Maintenant je vais vous demander de faire un petit exercice:

- Agrandissez un peu la boîte de dialogue.
- Ajoutez des objets « case à cocher » pour les notes accidentées que nous n'avons pas encore prises en compte (par exemple »remplacer Si# par Do naturel » ou l'inverse.
- Écrivez les lignes de code nécessaires pour activer ces objets. Vous pouvez copier-coller les lignes d'instruction d'un autre objet et modifier seulement les noms ou valeurs qui changent.

Ensuite, si votre script est particulièrement réussi et complet, proposez-le à Myriad qui l'inclura peut-être dans sa rubrique Bonus.