The REBOL Documentation Project

-- FR - Documentation REBOL - Articles Techniques --

Articles Techniques

Utilisation des Layouts dans REBOL/View

Philippe Le Goff

Première publication : 20 juin 2006, et mis en ligne le jeudi 22 juin 2006

Résumé :

De l'usage des layouts dans REBOL/View

Utilisation des Layouts dans REBOL/View

Auteur : Carl Read, traduit avec son autorisation.
Date Création : April 02, 2005, 08:50:55 PM
Version : 1.0.0

Ouvrir une fenêtre avec une interface symphatique est très simple à réaliser avec View, mais les nouveaux utilisateurs peuvent être surpris, lorsqu'ils souhaitent en ouvrir plus d'une.

Pour illustrer cela, considérez ce petit script ...

```
rebol []

view/title layout [   button "New Window" [    view/new/title layout [     box blue "Window 2"    ] "Window 2"   ]   button "Quit" [unview/all]  ] "Main"
```

(Note : La manière la plus rapide d'utiliser ces exemples est de les copier-coller dans une console View.)

Avec ce code, vous pouvez ouvrir une deuxième fenêtre simplement en cliquant sur le bouton "New Window", dans la fenêtre principale (Main). Il y a un piège cependant, car si vous déplacez votre fenêtre "Window2" placée au-dessus de la fenêtre principale, et que vous cliquez à nouveau sur le bouton "New Window", une troisième fenêtre apparaitra, également appelé "Window 2", etc. Ce n'est pas ce que vous vouliez !!

For heureusement, il y a une façon simple de contourner ce probléme - juste en créant un layout fixe, et en utilisant ensuite **view/new**. Par exemple :

```
rebol []

win2 : layout [   box blue "Window 2"  ]

view/title layout [   button "New Window" [    view/new/title win2 "Window 2"   ]   button "Quit" [unview/all]  ] "Main"
```

A présent, après avoir ouvert la fenêtre "Window 2", vous constaterez que vous ne pourrez plus ouvrir plusieurs fenêtres identiques. Cette façon de faire a aussi un petit "plus" : si vous fermez la fenêtre et que vous l'ouvrez à nouveau, elle apparaîtra à la même place dans l'écran que là où elle se trouvait auparavant, ce dont vos utilisateurs vous remercieront grandement.

C'est une excellente méthodologie que de gérer des fenêtres uniques dans votre programme. Cependant, il y aura des moments où vous voudrez ouvrir un nombre quelconque de fenêtres toutes plus ou moins identiques - comme celles affichant des images dans un programme de dessin, par exemple. Est-ce que notre premier script est Ok pour cela ? Voyons voir ...

```
rebol []

view/title layout [ button "New Window" [ view/new/title layout [ b : box blue "A Box" button
"Change !" [ b/color : random 255.255.255 show b ] ] "A Box" ] button "Quit" [unview/all]
] "Main"
```

Exécutez ce code et vous constaterez que vous pouvez ouvrir un nombre quelconque de fenêtre avec une boîte dedans, mais si vous cliquez sur le bouton "Change !", pour modifier aléatoirement la couleur de la boîte, vous remarquerez que c'est dans la *dernière fenêtre ouverte* que s'effectuent les changements. C'est parce que le mot **B** est global et donc peut uniquement faire référence à une boîte (box) à la fois - la dernière ouverte.

Une fois encore, il y a une solution de contournement à notre problème (et, plus exactement, plusieurs solutions), bien qu'il ne soit quand même pas aussi facile de résoudre ceci que d'empêcher des fenêtres multiples de s'ouvrir. Nous résoudrons ce probléme en utilisant le mot VAR contenu dans les attributs des faces, VAR permettant aux mots (tel que B dans notre boîte) d'être utilisés comme des références dans les layouts.

Un layout (qui est juste un objet REBOL standard) possède un bloc PANE qui contient toutes les *sous-faces* présentes dans le layout, tels que les boîtes et les boutons utilisés dans les exemples précédents. Ceci permet de rechercher dans un layout une face référencée par un mot spécifique et c'est ce que fera la fonction GET-PANE dans notre prochain exemple.

Voici résolu, mais à moitié seulement, le probléme, car nous avons encore besoin de trouver le layout avec lequel travailler (celui qui a le bouton sur lequel nous allons cliquer), pour pouvoir appeler cette fonction GET-PANE. Ceci semble raisonnablement facile, le mot FACE référençant le bouton et FACE/PARENT-FACE référençant le layout qui le contient.

Ce point étant résolu, nous pouvons utiliser les syntaxes : GET IN face word (dans lequel "word" est 'text, ou 'color, etc...) pour accéder à la valeur du mot (comme avec B/color), et : SET IN face word , pour modifier une valeur. C'est plus long que d'utiliser juste B/text ou B/color, mais cela permet d'utiliser les mêmes références dans des fenêtres multiples, et c'est bien là une chose symphatique que nous pouvons faire.

Et voici, une illustration de tout cela ...

```
rebol []

get-pane : func [face [object !] word [word !]][ foreach pane face/pane [if pane/var = word [return
pane]] make error ! rejoin ["Word '" word " not found in layout !"] ]

view/title layout [ button "New Window" [ view/new/title layout [ b : box blue "A Box" button
"Change !" [ set in get-pane face/parent-face 'b 'color random 255.255.255 show get-pane
face/parent-face 'b ] button "Print Color" [ print get in get-pane face/parent-face 'b 'color ]
] "A Box" ] button "Quit" [unview/all] ] "Main"
```

Ceci pourrait encore être simplifié un petit peu en écrivant des fonctions dédiées pour manipuler SET, GET et SHOW - c'est probablement le mieux à faire.

Vos commentaires ou questions sont les bienvenues.

(Traduction : Philippe Le Goff, avec l'autorisation de carl Read).