La méthode OnTime : question de temps ?
Catégorie : VBA par l'exemple
Publié par myDearFriend! le 12-06-2006

Lancer une procédure à une heure précise ou dans un délai prédéterminé, déclencher une action à intervalle régulier (un clignotement de cellule par exemple), comment gérer cette notion de temps en VBA sans pour autant suspendre le déroulement du programme ? Voilà un problème auquel on se confronte tôt ou tard.

A ma connaissance, il existe deux façons de procéder :

1. L'utilisation de la méthode OnTime que nous allons voir ci-dessous. C'est la procédure la plus simple à mettre en oeuvre et c'est aussi la plus documentée (voir l'aide VBA). On peut lui reprocher 2 inconvénients toutefois : d'une part, cette méthode peut provoquer un léger scintillement de l'écran pas très esthétique pendant le déroulement du décompte de temps et d'autre part, il est impossible d'obtenir une précision inférieure à la seconde.
2. Une autre manière d'aborder ces problèmes de gestion du temps consiste à recourir à l'utilisation d'API Windows. Une façon beaucoup plus évoluée d'obtenir satisfaction, mais aussi beaucoup plus sensible en terme de mise au point. Nous le verrons dans un autre article, cette façon de faire permet de passer outre les inconvénients de la méthode OnTime, mais en revanche, sa mise en oeuvre reste toujours un exercice délicat voir "hasardeux" dans certaines situations. Plantages et autres bonnes surprises du genre sont le lot quotidien de ceux qui, comme moi non spécialiste, s'essayent au bricolage des API Windows !

La méthode OnTime

Cette méthode permet de programmer l'exécution d'une procédure à une heure précise sans suspendre l'exécution du programme.

Elle s'applique à l'objet Application et se présente comme suit :

Application.OnTime EarliestTime, Procedure, [LatestTime], [Schedule]

- EarliestTime : Argument de temps. Il représente l'horaire "Top Départ" pour la procédure à exécuter.
- Procedure : Argument chaine de caractères. Il s'agit du nom de la procédure à lancer.
- LatestTime : Argument de temps (facultatif). Il sert à définir l'heure maximale à laquelle la procédure peut être exécutée. Si l'évènement n'a pu s'exécuté en temps voulu (car Excel n'était pas prêt), LatestTime indique si il est encore temps de le déclencher ou non. Cet argument est généralement omis pour que l'évènement programmé ait toujours lieu.
- Schedule : Argument boolean (facultatif). Il vaut True par défaut et sert à "activer" ou "désactiver" la gestion de l'évènement programmé.

Le principe de fonctionnement est simple :

Pour lancer la procédure "MaMacro" à 14H30 précise

```
Application.OnTime TimeValue("14:30:00"), "MaMacro"
```

Pour désactiver l'évènement prévu à 14H30 avant son exécution

```
Application.OnTime TimeValue("14:30:00"), "MaMacro", , False
```

Pour lancer la procédure "MaMacro" dans 2 minutes et 45 secondes

```
Application.OnTime Now + TimeValue("00:02:45"), "MaMacro"
```

EXEMPLE

Afficher l'heure qui défile (à la seconde près) dans une cellule de la feuille de calcul.
Bien évidemment, il ne s'agit que d'un exercice. Je vous déconseille d'ailleurs d'appliquer une telle procédure dans un projet dit "sérieux" car vous verrez qu'en utilisant cette méthode en boucle (toutes les secondes !), on peut apercevoir un léger scintillement de l'écran qui devient très pénible à la longue...

- Dans un module de code standard, on va mettre le code suivant :

DANS UN MODULE DE CODE STANDARD

```
Option Explicit
Dim Tps As Date

Sub Tempo()
    'Programmation de l'évènement toutes les secondes
    Tps = Now + TimeValue("00:00:01")
    Application.OnTime Tps, "Tempo"
    'Traitement
    Sheets(1).Range("A1").Value = Format(Now, "hh:nn:ss")
End Sub

Sub StopTempo()
    On Error Resume Next
    'Stopper la gestion de l'évènement OnTime en cours
    Application.OnTime Tps, "Tempo", , False
End Sub
```

- Puis, dans le module de code de l'objet ThisWorkbook, on place :

DANS LE MODULE DE CODE DE L'OBJET THISWORKBOOK

```
Option Explicit

Private Sub Workbook_BeforeClose(Cancel As Boolean)
    StopTempo
End Sub
```

Il suffira donc de lancer la procédure Tempo pour voir l'heure défiler dans la cellule A1 de la feuille 1 du classeur actif.
Pour arrêter le chronomètre, il conviendra de lancer la procédure StopTempo.

Dans ce morceau de code, notez l'importance de "Dim Tps As Date" déclaré en tête de module. La variable Tps prend une nouvelle valeur à chaque seconde dans la procédure Tempo. Lorsqu'il y a lieu de désactiver le dernier évènement OnTime en cours (par la procédure StopTempo), il est impératif de lui fournir à nouveau l'exacte valeur Tps ayant servie à son appel. Si vous ne prenez pas garde de désactiver OnTime avec une valeur EarliestTime strictement identique à son appel, le système ne saura pas retrouver l'évènement programmé correspondant et la désactivation n'aura pas lieu...

Remarquez également l'appel de StopTempo dans l'évènement Beforeclose() de l'objet ThisWorkbook. Cet appel est nécessaire car il permet de forcer la désactivation de l'évènement OnTime au moment de quitter le classeur. Sans ce garde-fou, l'évènement déclenché toutes les secondes risque fort de vous empêcher de fermer votre document !

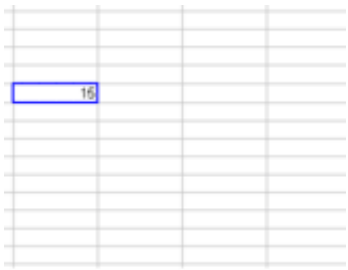
Bien évidemment, si on souhaite l'affichage de cette horloge dès l'ouverture du classeur, on peut rajouter dans le module de code de ThisWorkbook :

AJOUT DANS LE MODULE DE CODE DE L'OBJET THISWORKBOOK

```
Private Sub Workbook_Open()
    'Activation de l'heure
    Tempo
End Sub
```

Pour illustrer l'utilisation de cette méthode OnTime, vous trouverez en Section Téléchargements et plus particulièrement dans la catégorie Classeurs Exemples/Horloges, Timers et Clignotements , les 4 fichiers suivants :

Cellules et Alertes clignotantes : dans ce fichier, 4 exemples d'alertes et clignotements de cellules rendues possibles grâce à cette méthode.



Horloge Digitale : une horloge "classique" dans ce genre d'exercice.

Horloge Analogique Rudimentaire : une horloge à l'esthétique "brute de brute", sans aucun artifice, mais dont la particularité réside dans son mode d'affichage du cadran et des aiguilles. En effet, l'ensemble prend forme exclusivement grâce aux changements de couleur de fond des cellules de la feuille de calcul. Aucun objet Dessin ou autre n'a été utilisé pour sa conception ici !

Horloge Analogique : une autre horloge avec cadran et aiguilles, mais cette fois un peu plus évoluée car entièrement réalisée à base d'objets de la barre d'outils Dessin. En prime, un petite fenêtre calendrier... comme sur les vraies !