eBook Gratuit

APPRENEZ AutoHotkey

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#autohotkey

Table des matières

À propos

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: autohotkey

It is an unofficial and free AutoHotkey ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official AutoHotkey.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapitre 1: Démarrer avec AutoHotkey

Remarques

AutoHotkey est un langage de script personnalisé open source gratuit pour Microsoft Windows, initialement destiné à fournir des raccourcis clavier ou des raccourcis clavier faciles à utiliser, une macro- création rapide et une automatisation logicielle permettant aux utilisateurs de la plupart des compétences informatiques d'automatiser des tâches répétitives. Les interfaces utilisateur peuvent facilement être étendues ou modifiées par AutoHotkey (par exemple, remplacer les commandes de clé de contrôle Windows par défaut par leurs équivalents Emacs). L'installation d'Autohotkey inclut son propre fichier d'aide complet avec une version Web toujours mise à jour.

Vous pouvez écrire des macros de souris ou de clavier , remapper des touches , créer des raccourcis clavier , développer des abréviations, modifier le contenu du presse-papiers et créer des fichiers exécutables pour exécuter des scripts de raccourci sur des ordinateurs sur lesquels AutoHotkey n'est pas installé.

Versions

AutoHotkey 1.0. * - également connu sous le nom rétroactif d'AutoHotkey Basic, Classic, Vanilla, etc.

(Le développement a été interrompu en 2011; dernière stable: 2009)

Version	Date de sortie
v1.0.48.05	2009-09-26
v1.0.97.02	2011-04-14

AutoHotkey 1.1. * - précédemment appelé AutoHotkey_L.

(Stable et reçoit des mises à jour régulièrement)

Version	Date de sortie
v1.1.24.00	2016-05-22
v1.1.24.01	2016-08-02

AutoHotkey 2.0-a *

(Toujours en phase alpha)

Version	Date de sortie
v2.0-a069	2015-10-24
v2.0-a070	2015-11-09
v2.0-a071	2015-12-25
v2.0-a072	2015-12-25
v2.0-a073	2016-02-05
v2.0-a074	2016-03-11
v2.0-a075	2016-06-03

Examples

Installation ou configuration

De la documentation du site Autohotkey

1. Accédez à la page d'accueil AutoHotkey .
2. Cliquez sur Télécharger , une fois téléchargé, exécutez l'exécutable.
3. Lors de l'installation d'AutoHotkey, vous serez invité à choisir entre UNICODE ou ANSI. En bref, vous voudrez probablement choisir UNICODE. Il prend en charge les lettres et les chiffres non anglais (caractères).
4. Continuez jusqu'à ce que vous voyez un bouton Installer.
5. Une fois fait, génial!

Utiliser comme logiciel portable

1. Accédez à la page de téléchargement de AutoHotkey.
2. Recherchez la section Portable, choisissez UNICODE 32, 64 ou ANSI et téléchargée.
3. Lorsque vous choisissez le dossier de destination, sélectionnez tout périphérique de stockage externe correct ou non.
4. Maintenant, vous pouvez choisir d'associer des fichiers .ahk avec Autohotkey.exe
5. Créez un fichier texte brut et donnez-lui l'extension .ahk
6. Cliquez ensuite avec le bouton droit sur le fichier .ahk dans l'explorateur, puis cliquez sur Propriétés.
7. Dans les propriétés du fichier, cliquez sur le bouton Modifier situé à côté de l'option "Ouvre avec".
    - Après avoir cliqué sur Modifier, vous recevrez une liste de programmes pour ouvrir le fichier, sélectionnez le programme que vous souhaitez utiliser, puis cliquez sur OK ou sur Appliquer.

    - Si le programme que vous souhaitez sélectionner n'est pas répertorié, cliquez sur le bouton Parcourir et recherchez le fichier exécutable Autohotkey (.exe), puis cliquez sur OK pour sélectionner ce programme.
8. Maintenant, les fichiers .ahk fonctionneront comme si autohotkey était installé, génial!

Si vous avez installé Chocolatey, exécutez la commande suivante en tant qu'utilisateur admin

```
choco installer autohotkey
```

Alternativement, il peut être construit à partir du code source. Voir ici pour plus de détails: https://github.com/Lexikos/AutoHotkey_L/

Bonjour le monde

Montrez un "Hello World!" dans la boîte de message.

```
MsgBox, Hello World!
```

Montrez un "Hello World!" dans info-bulle.

```
#Persistent
Tooltip, Hello World!
```

Montrez un "Hello World!" message dans la barre de modification

```
#Persistent
TrayTip,, Hello World!
```

Imprime "Hello, World" en sortie standard (stdout).

```
FileAppend, % "Hello, World", *
```

Montrer "Hello World" dans une interface graphique

```
Gui, Add, Text,, Hello World!
Gui, Show, w200 h200
return

GuiClose:
ExitApp
```

Obtenir un effet similaire à SplashTextOn

```
Gui, +AlwaysOnTop +Disabled -SysMenu +Owner  ; +Owner avoids a taskbar button.
Gui, Add, Text,, Some text to display.
Gui, Show, NoActivate, Title of Window  ; NoActivate avoids deactivating the currently active
window.
```

Comment créer un script

Une fois AutoHotkey installé, vous voudrez probablement qu'il fasse des choses. AutoHotkey n'est pas magique, nous le souhaitons tous, mais ce n'est pas le cas. Nous devrons donc lui dire quoi faire. Ce processus s'appelle "Scripting".

1. Clic droit sur votre bureau.
2. Trouvez "Nouveau" dans le menu.
3. Cliquez sur "AutoHotkey Script" dans le menu "Nouveau".
4. Donnez un nouveau nom au script. Note: Il doit se terminer par une extension .ahk. Ex. MyScript.ahk
5. Recherchez le nouveau fichier créé sur votre bureau et cliquez dessus avec le bouton droit de la souris.
6. Cliquez sur "Modifier le script".
7. Une fenêtre aurait dû apparaître, probablement Notepad. Si oui, SUCCESS!

Donc, maintenant que vous avez créé un script, vous devez ajouter des éléments dans le fichier. Pour obtenir une liste de toutes les commandes, fonctions et variables intégrées, reportez-vous à la section 5. Voici un script très simple contenant un raccourci clavier qui tape du texte à l'aide de la commande Envoyer lorsque vous appuyez sur la touche d'activation.

```
^j::
   Send, My First Script
Return
```

Nous approfondirons plus tard. En attendant, voici une explication du code ci-dessus.

- La première ligne `^j::` est le raccourci clavier. `^` signifie `CTRL` , `j` est la lettre j. Tout ce qui se trouve à gauche de `::` sont les touches sur lesquelles vous devez appuyer.
- La deuxième ligne `Send, My First Script` est la façon dont vous `SEND` frappes. `SEND` est la commande, tout ce qui est après la virgule (,) sera saisi.
- La troisième ligne. `Return` Le retour deviendra votre meilleur ami. Il littéralement STOPS code d'aller plus loin, aux lignes ci-dessous. Cela évitera de nombreux problèmes lorsque vous commencerez à avoir beaucoup de choses dans vos scripts.

8. Enregistrez le fichier.
9. Double-cliquez sur le fichier / icône sur le bureau pour l'exécuter. Ouvrez le bloc-notes ou (tout ce que vous pouvez taper) et appuyez sur Ctrl et J.
10. Hip Hip Hourra! Votre premier script est terminé. Allez chercher des snacks récompensants puis revenez à la lecture de la suite de ce tutoriel.

Lire Démarrer avec AutoHotkey en ligne: https://riptutorial.com/fr/autohotkey/topic/3532/demarrer-avec-autohotkey

Chapitre 2: Bonjour le monde

Examples

Bonjour tous les exemples

Montrez un "Hello World!" dans la boîte de message.

```
MsgBox, Hello World!
```

Montrer "Bonjour le monde!" stocké dans la variable MyString dans la boîte de message.

```
MyString := "Hello World!"
MsgBox, %MyString%
```

Montrez un "Hello World!" dans info-bulle.

```
#Persistent
Tooltip, Hello World!
```

Montrez un "Hello World!" message dans la barre de modification

```
#Persistent
TrayTip,, Hello World!
```

Montrer "Hello World!" dans une fenêtre graphique.

```
Gui, Add, Text,, Hello World!
Gui, Add, Edit,, Hello World!
Gui, Show
return

GuiClose() {
    ExitApp
}
```

Estampes "Bonjour, Monde!" à la sortie standard (stdout).

```
FileAppend, % "Hello, World!", *
```

Simuler en tapant "Hello, World!" en appuyant sur enter

```
Enter::Send, Hello World{!}
```

Créez un nouveau fichier appelé HelloWorld.txt

```
FileAppend,, HelloWorld.txt
```

Lorsque vous tapez le mot "Hello" suivi d'un espace ou entrez, vous serez remplacé par "Hello World"

```
::Hello::Hello World
```

Lire Bonjour le monde en ligne: https://riptutorial.com/fr/autohotkey/topic/3547/bonjour-le-monde

Chapitre 3: Champ de saisie

Introduction

Pour obtenir la saisie d'un utilisateur et la stocker dans une variable, vous pouvez utiliser la commande InputBox. Le script ne continuera pas d'exécuter des commandes tant que l'utilisateur n'appuie pas sur «OK» ou «Annuler».

'OK' fermera la fenêtre et sauvegardera la saisie de l'utilisateur 'Cancel' fermera la fenêtre, supprimant la saisie de l'utilisateur

Paramètres

InputBox, OutputVar [, Titre, Invite, HIDE, Largeur, Hauteur, X, Y, Délai d'expiration, Par défaut]	Que signifie chaque option
OutputVar	La variable saisie par l'utilisateur sera enregistrée dans
Titre	Le nom de la zone de saisie
Rapide	Texte à l'intérieur de la zone de saisie
CACHER	Affiche la saisie de l'utilisateur sous forme d'astérisques pour masquer l'entrée - tapez HIDE pour activer
Largeur	La largeur de la zone de saisie
la taille	La hauteur de la zone de saisie
X	La quantité de pixels à partir du bord gauche de l'écran que le coin supérieur gauche de la zone de saisie sera
Y	La quantité de pixels à partir du bord supérieur de l'écran que le coin supérieur gauche de la zone de saisie sera
Temps libre	Ferme automatiquement la zone de saisie et enregistre la saisie de l'utilisateur après cette heure en millisecondes
Défaut	Le texte qui apparaîtra dans le champ modifiable de la zone de saisie lorsqu'il est ouvert

Remarques

Une zone de saisie est un élément de l'interface graphique, il sera donc traité comme un élément de l'interface graphique.

Une liste de niveaux d'erreur pour cette commande:

Niveau de l'erreur	Ce que cela veut dire
0	L'utilisateur a appuyé sur le bouton 'OK'
1	L'utilisateur a appuyé sur le bouton 'Annuler'
2	La zone de saisie a expiré

Vous pouvez trouver la page pour cette commande dans la documentation AutoHotkey ici: https://autohotkey.com/docs/commands/InputBox.htm

Examples

Exemple d'utilisation de base

```
InputBox, userinput
```

Cela stockera ce que l'utilisateur tape dans la zone de saisie de la variable nommée *userinput*

Mots de passe

```
InputBox, password, Enter your Password,, HIDE,, 100

Loop, {
  if (errorlevel = 1)
return

  if (password = "password") {
MsgBox, The password is correct.
    return
  } else if (password != "password") {
MsgBox, The password is incorrect.
InputBox, password, Enter your Password,, HIDE,, 100
  }
}
```

Cela vérifiera si l'utilisateur a tapé "password" dans la zone de saisie. Si l'utilisateur tape la valeur correcte, il dira "Le mot de passe est correct". et fermez la zone de saisie. Si l'utilisateur tape la valeur incorrecte, il dira "Le mot de passe est incorrect". et rouvrez la zone de saisie. Si le niveau d'erreur est 1 (l'utilisateur a appuyé sur Annuler), le script sera terminé.

Lire Champ de saisie en ligne: https://riptutorial.com/fr/autohotkey/topic/9917/champ-de-saisie

Chapitre 4: Ouvrir un fichier dans un script

Introduction

Différentes manières d'ouvrir un fichier avec lequel travailler dans un script.

Examples

Ouvrir un fichier via l'Explorateur Windows

Dans le script, utilisez la première ligne pour stocker la toute première variable (dans cet exemple, `%1%`) avec un nom à traiter. Exemple: `OpenWithFile = %1%`

Une fois que vous ouvrez un fichier avec ce script via Windows (cliquez avec le bouton droit sur n'importe quel fichier sur MS Windows et choisissez «Ouvrir avec…», puis sélectionnez la version **compilée** du script telle que script.exe), le nom du fichier choisi sera stocké dans cette variable et, ainsi, le script sera capable de travailler avec elle. Exemple:

```
OpenWithFile = %1%
if OpenWithFile !=
{
FileRead, content, %OpenWithFile%
msgbox %content%
return
}
```

Ouvrir une boîte de dialogue Fichier via SelectFile

L'exemple suivant crée une interface graphique avec un seul bouton qui amène la boîte de dialogue SelectFile.

```
Gui, Loader: New
Gui, Loader: Add, Button, Default Center w220 vLOAD, LOAD
Gui, Loader: Show, AutoSize Center, Loader

return

LoaderButtonLOAD:
FileSelectFile, LoadedFile, , , ,

if ErrorLevel=1
{
return
}

else
{
    FileRead, content, %LoadedFile%
    msgbox %content%
}
```

```
return
```

Ouvrir un fichier via Windows Drag n 'Drop

Cet exemple crée un nouveau Gui sensible à l'événement Drag n 'Drop:

```
Gui, Dropper: New
Gui, Dropper: Font, s10 w700
Gui, Dropper: Add, Text, y80 vText1, Drag the files here
Gui, Dropper: Show, w200 h200 Center, Dropper

return

DropperGuiDropFiles:
DroppedFile:=A_GuiEvent

    FileRead, content, %DroppedFile%
    msgbox %content%

return
```

Lire Ouvrir un fichier dans un script en ligne: https://riptutorial.com/fr/autohotkey/topic/9070/ouvrir-un-fichier-dans-un-script

Chapitre 5: Scripts Hotkey

Syntaxe

- raccourcis clavier::
- ::abréviation::
- Revenir

Paramètres

Raccourcis clavier	Détails
^	Touche Ctrl
!	touche Alt
+	Touche Majuscule
#	Clé Windows
{entrer}	envoyer la clé d'entrée
{languette}	clé de tabulation
*	joker, n'importe quelle touche peut être enfoncée
~	la fonction native de la clé ne sera pas bloquée
<symbole	spécifie la touche gauche (<+ est à gauche)
> le symbole	spécifie la bonne clé

Examples

Raccourci clavier

Pour créer un raccourci clavier qui envoie la séquence de touches 'Hello World' en appuyant sur `Ctrl` + `J` dans la fenêtre active (peut être démontré dans le Bloc-notes, par exemple)

```
^j::
    Send, Hello World
Return
```

Hotstring

Pour créer un script pour remplacer une phrase, utilisez la syntaxe `::abbreviation::` hotstring. Cela remplacera `btw` par `by the way` chaque fois que vous entrez `btw` puis la touche espace.

```
::btw::by the way
```

Si vous voulez créer un script de connexion pour vous connecter plus rapidement, vous pouvez créer un script comme celui-ci (le fichier n'est pas chiffré, donc toute information de votre script sera visible par toute personne ayant accès au fichier).

```
::lmi::user{tab}password{enter}
```

Touches multiples

Pour exécuter un script lorsque plusieurs touches sont enfoncées, utilisez les touches `&` entre les touches.

```
Numpad0 & Numpad1::
    MsgBox You pressed 0 and 1
return
```

Raccourcis contextuels et Hotstrings

Pour créer un raccourci clavier ou une chaîne de raccourci qui ne se déclenche que lorsque certaines fenêtres sont actives ou existantes, vous pouvez placer une ou plusieurs des *directives* suivantes avant la définition du raccourci clavier:

```
#IfWinActive [, WinTitle, WinText]
#IfWinExist [, WinTitle, WinText]
#IfWinNotActive [, WinTitle, WinText]
#IfWinNotExist [, WinTitle, WinText]
```

Exemple: Vous voulez `stackoverflow.com` à envoyer chaque fois que vous tapez `so` (et un espace après que) dans Google Chrome, mais ignorer la combinaison de chaîne dans une autre fenêtre.

```
#IfWinActive, ahk_class Chrome_WidgetWin_1
::so::stackoverflow.com
```

En utilisant `#If [, Expression ]` , vous ne pouvez déclencher un raccourci que lorsqu'une expression arbitraire est vraie, par exemple:

```
#If A_Hour < 9
F1::
    MsgBox, It is too early to ask for help!
return
```

Touches de remappage

L'exemple suivant remappe la clé `Z` sur `Y` et inversement, par exemple si vous souhaitez utiliser la

disposition QWERTY sur un clavier QWERTZ.

```
z::y
y::z
```

Touches d'activation commutables

Le script suivant entre des chaînes prédéfinies sur les touches de raccourci si le verrouillage de défilement est actif. Cela peut être utile si vous collez souvent un certain nombre de chaînes répétées. Raccourcis inclus pour l'actualisation du script (par exemple, si vous devez modifier des chaînes pouvant être collées).

```
; refresh script hotkey
Numpad9::
    GetKeyState, state, ScrollLock, T
    if ( state = "D" )
        Reload
Return

Numpad1::
    GetKeyState, state, ScrollLock, T
    if ( state = "D" )
        Send,         Hello
Return

Numpad2::
    GetKeyState, state, ScrollLock, T
    if ( state = "D" )
        Send,         World
Return
;...
```

Lire Scripts Hotkey en ligne: https://riptutorial.com/fr/autohotkey/topic/3853/scripts-hotkey

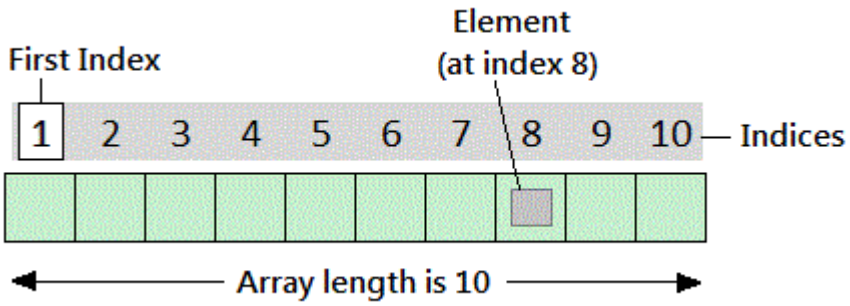
Chapitre 6: Tableaux

Examples

Création et initialisation de tableaux simples

Introduction

Un *tableau* est un objet conteneur contenant plusieurs valeurs. Dans l'image suivante, vous pouvez voir un tableau de taille 10, le premier élément indexé 1 et le dernier élément 10.



Autohotkey offre quelques manières de définir et de créer des tableaux.

- Tableau: = []
- Tableau: = Array ()

Création et initialisation de tableaux avec un nombre N d'éléments

```
Array := [Item1, Item2, ..., ItemN]
Array := Array(Item1, Item2, ..., ItemN)
```

Dans Autohotkey, il est possible d'avoir des tableaux sans élément:

```
Array := [] ; works fine.
```

Et des éléments peuvent ensuite lui être assignés:

```
Array[0] := 1
```

La taille du tableau peut être déterminée en utilisant une méthode appelée `length` :

```
msgbox % array.length()  ; shows 1 in this case.
```

Si le tableau n'est pas vide, **MinIndex** et **MaxIndex / Length** renvoient l'index le plus bas et le plus élevé actuellement utilisé dans le tableau. Comme l'index le plus bas est presque toujours 1, MaxIndex renvoie généralement le nombre d'éléments. Cependant, s'il n'y a pas de clés entières, MaxIndex renvoie une chaîne vide alors que Longueur renvoie 0.

Création et initialisation de tableaux multidimensionnels

Vous pouvez créer un tableau multidimensionnel comme suit:

```
Array[1, 2] := 3
```

Vous pouvez créer et initialiser en même temps, et il n'est pas nécessaire que les tableaux internes aient la même longueur.

```
Array := [[4,5,6],7,8]
```

Les tableaux comme celui-ci sont également appelés tableaux de tableaux.

Remplir un tableau

```
; Assign an item:
Array[Index] := Value

; Insert one or more items at a given index:
Array.InsertAt(Index, Value, Value2, ...)

; Append one or more items:
Array.Push(Value, Value2, ...)
```

La valeur d'un index pour un élément de tableau peut également être un entier négatif (-1, 0, 1, 2, 3, 4, ...)

Suppression d'éléments d'un tableau

```
; Remove an item:
RemovedValue := Array.RemoveAt(Index)

; Remove the last item:
RemovedValue := Array.Pop()
```

Ajout de méthodes personnalisées en

remplaçant la fonction Array ()

AutoHotkey est un langage de programmation basé sur un prototype , ce qui signifie que vous pouvez remplacer n'importe quelle fonction / objet intégré à tout moment. Cet exemple montre comment remplacer la fonction Array () afin d'ajouter des méthodes dans un objet de classe personnalisé.

```
; Overrides Array()
Array(Args*) {
    Args.Base := _Array
    Return Args
}

; Custom Class Object with Methods
Class _Array {

    ; Reverses the order of the array.
    Reverse() {
        Reversed := []
        Loop % This.MaxIndex()
            Reversed.Push(This.Pop())
        Return Reversed
    }

    ; Sums all Integers in Array
    Sum(Sum=0) {
        For Each, Value In This
            Sum += Value Is Integer ? Value : 0
        Return Sum
    }
}

; Arr == ["Hello, World!", 4, 3, 2, 1]
Arr := [1, 2, 3, 4, "Hello, World!"].Reverse()

; SumOfArray == 10
SumOfArray := Arr.Sum()
```

Lire Tableaux en ligne: https://riptutorial.com/fr/autohotkey/topic/4468/tableaux

Chapitre 7: Utiliser des fonctions au lieu d'étiquettes

Remarques

AutoHotkey utilisé pour s'appuyer fortement sur les étiquettes jusqu'à la version 1.1.20. Sa dépendance aux étiquettes avait de très sérieux inconvénients. La principale étant que les étiquettes s'exécutent généralement dans la portée globale, ce qui signifie que toute variable définie dans une étiquette sera disponible globalement. Cela semble bien jusqu'à ce que vous réalisiez que, par exemple, vous ne pouvez pas simplement utiliser les bibliothèques d'autres peuples sans vous assurer que leurs variables n'interfèrent pas avec les vôtres.
Travailler dans un contexte mondial quand ce n'est pas nécessaire est simplement une mauvaise pratique.

C'est donc là que les fonctions entrent en jeu. Depuis la version 1.1.20, chaque commande AutoHotkey qui accepte un nom-étiquette en tant que paramètre accepte désormais alternativement un nom-fonction.

Examples

Exemple très basique démontrant l'utilisation de la fonction sur SetTimer.

```
;Sends the keystroke for the letter "a" every 3 seconds.
#Persistent
SetTimer, SendLetterA, 3000
return

SendLetterA() {
    Send, a
}
```

Utilisation avancée de SetTimer: appel de la même fonction avec des paramètres différents

Ceci est un exemple de quelque chose qui aurait été impossible avec les étiquettes. Si vous exécutez le même libellé plusieurs fois en même temps et que celles-ci reposent sur des variables définies, elles risquent d'interférer et de provoquer un comportement inattendu.

Voici comment procéder avec les fonctions:

```
; This script will switch between showing "Hello 1" and "Hello 2"

#Persistent
DisplayMessage_Hello1 := Func("DisplayMessage").bind("Hello 1")
SetTimer, %DisplayMessage_Hello1%, 2000
```

```
Sleep, 1000

DisplayMessage_Hello2 := Func("DisplayMessage").bind("Hello 2")
SetTimer, %DisplayMessage_Hello2%, 2000

DisplayMessage(messageToDisplay) {
    TrayTip ; remove other traytips
    TrayTip, Message to display:, %messageToDisplay%
}
```

Voici **comment ne pas le faire** (avec des étiquettes):

```
;This script will never display the message "Hello 1". It will always show "Hello 2".

#Persistent
messageToDisplay := "Hello 1"
SetTimer, DisplayMessage, 2000

Sleep, 1000

messageToDisplay := "Hello 2"
SetTimer, DisplayMessage, 2000

DisplayMessage:
    TrayTip ; remove other traytips
    TrayTip, Message to display:, %messageToDisplay%
Return
```

Gui avec fonctions au lieu d'étiquettes

Exemple montrant comment créer Guis en utilisant des fonctions au lieu d'étiquettes.

```
Gui, Add, Button, gCtrlEvent vButton1, Button 1
Gui, Add, Button, gCtrlEvent vButton2, Button 2
Gui, Add, Button, gGoButton, Go Button
Gui, Add, Edit, vEditField, Example text
Gui, Show,, Functions instead of labels

CtrlEvent(CtrlHwnd:=0, GuiEvent:="", EventInfo:="", ErrLvl:="") {
    GuiControlGet, controlName, Name, %CtrlHwnd%
    MsgBox, %controlName% has been clicked!
}
GoButton(CtrlHwnd:=0, GuiEvent:="", EventInfo:="", ErrLvl:="") {
    GuiControlGet, EditField
    MsgBox, Go has been clicked! The content of the edit field is "%EditField%"!
}

GuiClose(hWnd) {
    WinGetTitle, windowTitle, ahk_id %hWnd%
    MsgBox, The Gui with title "%windowTitle%" has been closed!
    ExitApp
}
```

Hotkeys avec des fonctions au lieu d'étiquettes

Exemples d'utilisation de fonctions avec des raccourcis clavier:

```
Hotkey, a, MyFunction ; Calls MyFunction() when a is pressed

MyFunction() {
    MsgBox You pressed %A_ThisHotkey%.
}
```

Ou:

```
a::MyFunction()

MyFunction() {
    MsgBox You pressed %A_ThisHotkey%.
}
```

Actions du menu Plateau avec fonctions

```
#Persistent

Menu, Tray, NoStandard ; remove default tray menu entries
Menu, Tray, Add, MyDefaultAction, OnDefaultTrayAction ; add a new tray menu entry
Menu, Tray, Add, Exit, Exit ; add another tray menu entry
Menu, Tray, Default, MyDefaultAction ;When doubleclicking the tray icon, run the tray menu
entry called "MyDefaultAction".


OnDefaultTrayAction() {
    MsgBox, You double clicked the tray icon of this script or you clicked the MyDefaultAction
entry!
}

Exit() {
    MsgBox, You clicked the Exit entry! The script will close itself now.
    ExitApp
}
```

Exemple général de fonctions vs étiquettes

Je vais démontrer l'utilisation de base de l'utilisation des fonctions par rapport aux étiquettes + gosub.
Dans cet exemple, nous allons implémenter des fonctionnalités simples pour ajouter deux nombres et les stocker dans une variable.

Avec des fonctions:

```
c := Add(3, 2) ; function call
MsgBox, Result: %c%

Add(a, b) { ; This is a function. Put it wherever you want, it doesn't matter.
    ; the a and b inside of this function are set by the function call above
    Return a+b ; the function will return the result of the expression "a+b"
}
```

Avec des étiquettes (s'il vous plaît ne faites pas cela):

```
a := 3
b := 2
GoSub, Add ; execute the label "Add" then jump back to the next line here
MsgBox, Result: %c%
Return ; without this, the label would be executed again for no reason.

Add: ; This is a label. Please put them at the bottom of your script and use "Return" in a
line above.
    c := a+b
Return
```

OnClipboardChange avec fucntion / label

Le code suivant est extrait de la documentation officielle d'AutoHotkey:

Implémentation de la fonction:

```
#Persistent
OnClipboardChange("ClipChanged")
return

ClipChanged(Type) {
    ToolTip Clipboard data type: %Type%
    Sleep 1000
    ToolTip  ; Turn off the tip.
}
```

Mise en place du label:

```
#Persistent
return

OnClipboardChange:
ToolTip Clipboard data type: %A_EventInfo%
Sleep 1000
ToolTip  ; Turn off the tip.
return
```

Exemple de Gui plus compliqué avec plusieurs vues de liste utilisant la même fonction de rappel d'événement

Ce script montre comment recevoir des événements d'interface graphique compliqués à partir de différents contrôles dans la même fonction de rappel d'événement. Nous utiliserons deux contrôles ListView pour cela.
Désormais, chaque fois qu'une action est détectée sur l'un de ces contrôles ListView, nous voulons une description précise de ce qui s'est passé et que celle-ci soit connectée à un contrôle d'édition dans la même interface graphique.

```
Gui, Add, ListView, gListCtrlEvent vMyFirstListView AltSubmit -ReadOnly R10 w310,
ColumnTitle1|ColumnTitle2|ColumnTitle3
Gui, Add, ListView, gListCtrlEvent vMySecondListView AltSubmit -ReadOnly R10 w310,
ColumnTitle1|ColumnTitle2|ColumnTitle3
Gui, Add, Text, w310, Action Log
```

```
Gui, Add, Edit, vLog R7 w310,
Gui, Show,, Functions instead of labels

; Create example entries for the first ListView
Gui, ListView, MyFirstListView
Loop, 10 {
    LV_Add("", "Column-1 | Row-" A_Index ,  "Column-2 | Row-" A_Index,  "Column-3 | Row-"
A_Index)
}
LV_ModifyCol()

; Create example entries for the second ListView
Gui, ListView, MySecondListView
Loop, 10 {
    LV_Add("", "Column-1 | Row-" A_Index ,  "Column-2 | Row-" A_Index,  "Column-3 | Row-"
A_Index)
}
LV_ModifyCol()


ListCtrlEvent(ctrlHwnd:=0, guiEvent:="", eventInfo:="", errLvl:="") {
    GuiControlGet, ctrlName, Name, %CtrlHwnd%
    whatHappened := "Action detected!`n"
    whatHappened .= "Control handle: " ctrlHwnd "`n"
    whatHappened .= "Control name: " ctrlName "`n"

    If (guiEvent = "DoubleClick") {
        whatHappened .= "`nThe user has double-clicked within the control."
        whatHappened .= "`n> Focused row number: " eventInfo
    } Else If (guiEvent = "R") {
        whatHappened .= "`nThe user has double-right-clicked within the control."
        whatHappened .= "`n> Focused row number: " eventInfo
    } Else If (guiEvent = "ColClick") {
        whatHappened .= "`nThe user has clicked a column header."
        whatHappened .= "`n> Column number: " eventInfo
    } Else If (guiEvent = "D") {
        whatHappened .= "`nThe user has attempted to start dragging a row or icon."
        whatHappened .= "`n> Focused row number: " eventInfo
    } Else If (guiEvent = "d") {
        whatHappened .= "`nThe user has attempted to start right-click-dragging a row or
icon."
        whatHappened .= "`n> Focused row number: " eventInfo
    } Else If (guiEvent = "e") {
        whatHappened .= "`nThe user has finished editing the first field of a row."
        whatHappened .= "`n> Row number: " eventInfo
    } Else If (guiEvent = "Normal") {
        whatHappened .= "`nThe user has left-clicked a row."
        whatHappened .= "`n> Focused row number: " eventInfo
    } Else If (guiEvent = "RightClick") {
        whatHappened .= "`nThe user has right-clicked a row."
        whatHappened .= "`n> Focused row number: " eventInfo
    } Else If (guiEvent = "A") {
        whatHappened .= "`nA row has been activated."
        whatHappened .= "`n> Row number: " eventInfo
    } Else If (guiEvent = "C") {
        whatHappened .= "`nThe ListView has released mouse capture."
    } Else If (guiEvent = "E") {
        whatHappened .= "`nThe user has begun editing the first field of a row."
        whatHappened .= "`n> Row number: " eventInfo
    } Else If (guiEvent = "F") {
        whatHappened .= "`nThe ListView has received keyboard focus."
```

```
    } Else If (guiEvent = "f") {
        whatHappened .= "`nThe ListView has lost keyboard focus."
    } Else If (guiEvent = "I") {
        whatHappened .= "`nItem changed. (A row has changed by becoming selected/deselected,
checked/unchecked, etc.)"
        whatHappened .= "`n> Row number: " eventInfo
    } Else If (guiEvent = "K") {
        whatHappened .= "`nThe user has pressed a key while the ListView has focus."
        whatHappened .= "`n> Key pressed: " GetKeyName(Format("vk{:x}", eventInfo))
    } Else If (guiEvent = "M") {
        whatHappened .= "`nItem changed. (A row has changed by becoming selected/deselected,
checked/unchecked, etc.)"
        whatHappened .= "`n> Row number: " eventInfo
    } Else If (guiEvent = "S") {
        whatHappened .= "`nMarquee. The user has started to drag a selection-rectangle around
a group of rows or icons."
    } Else If (guiEvent = "s") {
        whatHappened .= "`nThe user has finished scrolling the ListView."
    }
    GuiControlGet, Log
    GuiControl,, Log, % whatHappened "`n----------------------`n" Log
}

GuiClose(hWnd) {
    WinGetTitle, windowTitle, ahk_id %hWnd%
    MsgBox, The Gui with title "%windowTitle%" is going to be closed! This script will exit
afterwards!
    ExitApp
}
```

Lire Utiliser des fonctions au lieu d'étiquettes en ligne:
https://riptutorial.com/fr/autohotkey/topic/4062/utiliser-des-fonctions-au-lieu-d-etiquettes

Chapitre 8: Variables et fonctions intégrées

Remarques

AutoHotkey est livré avec de nombreuses fonctions et variables intégrées qui peuvent être utilisées n'importe où dans un script.
Pour une liste complète comprenant des explications, voir:

- Liste des variables intégrées
- Liste des fonctions intégrées

Examples

Détermination du temps d'inactivité de l'utilisateur

```
if(A_TimeIdlePhysical > 60000) { ; 60,000 milliseconds
    WinClose, ahk_class Chrome_WidgetWin_1
    MsgBox, Google Chrome was closed due to user inactivity.
}
```

Cette vérification peut être effectuée périodiquement, par exemple en utilisant `SetTimer` .

Insérer automatiquement le nom du jour de la semaine en cours

Cet exemple insère / envoie le jour complet du nom complet de la semaine (par exemple *dimanche*) chaque fois que vous `appuyez sur Ctrl` + `Alt` + `D` :

```
^!d::Send, %A_DDDD%
```

Extraire des parties de chaîne à l'aide de RegEx

```
myDebt := 9000
index  := RegExMatch("You owe me $42", "\$(\d+)", dollars)
if(index > 0) { ; indices are usually 1-based in AHK
    myDebt += dollars1
    MsgBox, Current debt: %myDebt%
}
```

Résultat:

> Dette actuelle: 9042

Couper une ficelle

```
myString := "  hello, Trim()! "
trimmed  := Trim(myString)
```

```
FileAppend, % trimmed "`n", TrimmedStrings.txt
```

Notez que `Trim()` ne manipulera pas la chaîne d'origine, mais renverra une nouvelle qui doit être stockée ou sortie quelque part.

Lire Variables et fonctions intégrées en ligne:
https://riptutorial.com/fr/autohotkey/topic/4469/variables-et-fonctions-integrees

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec AutoHotkey	blackholyman, Community, depperm, Forivin, Joe DF, Shyam Sundar Shankar, tlm, Vijay
2	Bonjour le monde	errorseven, Forivin, Goerman, mikew, vasili111
3	Champ de saisie	Meh
4	Ouvrir un fichier dans un script	freestock.tk
5	Scripts Hotkey	depperm, MCL, user5226582
6	Tableaux	blackholyman, errorseven
7	Utiliser des fonctions au lieu d'étiquettes	Forivin
8	Variables et fonctions intégrées	MCL