Logo CADACOM

**World**

**VBA et Macros**

**Daniel DEVEAUX**

[Préambule 5](#_Toc226113936)

[La précision 5](#_Toc226113937)

[L'observation. 6](#_Toc226113938)

[La curiosité. 6](#_Toc226113939)

[Avant-propos important 8](#_Toc226113940)

[Ma première Macro simple. 9](#_Toc226113941)

[Enregistrement d'une macro 9](#_Toc226113942)

[Préambule 9](#_Toc226113943)

[But & Description de notre première macro simple 9](#_Toc226113944)

[Méthode 9](#_Toc226113945)

[Affectation à une séquence de Raccourci Clavier. 12](#_Toc226113946)

[Où stocker une macro (Ce Classeur, Autre Classeur ou encore Classeur de macros Personnelles). 13](#_Toc226113947)

[Début de l’enregistrement 14](#_Toc226113948)

[Enregistrement du code. 14](#_Toc226113949)

[Vérification du bon fonctionnement de notre première macro simple. 17](#_Toc226113950)

[Qu’avons-nous enregistré ? 17](#_Toc226113951)

[Le Module 19](#_Toc226113952)

[Le code 19](#_Toc226113953)

[Examen du Code généré 20](#_Toc226113954)

[Les Commentaires 20](#_Toc226113955)

[La correction automatique – Comment en améliorer la puissance. 22](#_Toc226113956)

[Préambule 22](#_Toc226113957)

[Le scénario proposé. 24](#_Toc226113958)

[La réalisation 24](#_Toc226113959)

[Sélection du texte 24](#_Toc226113960)

[Enregistrement de la Macro. 24](#_Toc226113961)

[Analyse et modificationdu Code 26](#_Toc226113962)

[Atteindre un signet à l’ouverture d’un document. 33](#_Toc226113963)

[Les signets 33](#_Toc226113964)

[Préambule 33](#_Toc226113965)

[Insertion d’un signet 33](#_Toc226113966)

[Atteindre un signet. 34](#_Toc226113967)

[Atteindre un signet sous forme de Macro. 34](#_Toc226113968)

[Exécuter cette macro à l’ouverture du document. 35](#_Toc226113969)

[Exécution de la Macro 36](#_Toc226113970)

[Macro faisant la liste des signets d’un document. 37](#_Toc226113971)

[Le code de base 37](#_Toc226113972)

[L’instruction Dim 37](#_Toc226113973)

[L’instruction Collapse 37](#_Toc226113974)

[Syntaxe 38](#_Toc226113975)

[Son rôle dans ce cas-ci. 38](#_Toc226113976)

[L’instruction InsertBreak 38](#_Toc226113977)

[Syntaxe 39](#_Toc226113978)

[Son rôle dans ce cas-ci. 39](#_Toc226113979)

[La boucle For Each … Next 39](#_Toc226113980)

[Syntaxe 39](#_Toc226113981)

[Son rôle dans ce cas-ci. 39](#_Toc226113982)

[Envoyer un document par Courriel. 41](#_Toc226113983)

[Le code 41](#_Toc226113984)

[Commentaires 41](#_Toc226113985)

[Ouvrir un insérer un document d’un autre format. 42](#_Toc226113986)

[La réalisation 42](#_Toc226113987)

[Le code 43](#_Toc226113988)

[Détection d’un mot particulier et surligner le paragraphe concerné. 45](#_Toc226113989)

[Processus 45](#_Toc226113990)

[Le code 45](#_Toc226113991)

[Explication du Code 45](#_Toc226113992)

[Selection.WholeStory 45](#_Toc226113993)

[Selection.HomeKey Unit:=wdStory 46](#_Toc226113994)

[With Selection.Find .Text = "Hercule 46](#_Toc226113995)

[While .Execute 46](#_Toc226113996)

[Selection.Range.HighlightColorIndex = wdYellow 46](#_Toc226113997)

[Selection.Collapse wdCollapseEnd 47](#_Toc226113998)

[Petits utilitaires pratiques 48](#_Toc226113999)

[Comment en VBA compter le nombre de lignes et/ou de mots d'un document ? 48](#_Toc226114000)

[Préambule 48](#_Toc226114001)

[Installation du code 48](#_Toc226114002)

[A noter 48](#_Toc226114003)

[Comment, en VBA, compter le nombre de pages d'un document ? 48](#_Toc226114004)

[Installation du code 48](#_Toc226114005)

[Liste de toutes les polices utilisées dans un document. 49](#_Toc226114006)

[Installation du code 49](#_Toc226114007)

[Explication du code 50](#_Toc226114008)

[Méthodologie. 50](#_Toc226114009)

[L’instruction « If » 50](#_Toc226114010)

[Selection.Style = ActiveDocument.Styles("Titre 1") 50](#_Toc226114011)

[Comment tester toutes les polices avec une phrase type que l'on aurait choisie ? 50](#_Toc226114012)

[Installation du code 50](#_Toc226114013)

[Imprimer la description des styles utilises 51](#_Toc226114014)

[Installation du code 51](#_Toc226114015)

[Remarque 52](#_Toc226114016)

[Epurer votre document des styles non utilisés. 52](#_Toc226114017)

[Installation du code 52](#_Toc226114018)

[Apport complémentaire. 53](#_Toc226114019)

[Gestion des cases à cocher dans un formulaire 54](#_Toc226114020)

[But 54](#_Toc226114021)

[Solution 54](#_Toc226114022)

[Installation du code 54](#_Toc226114023)

[Créer un nouveau document Excel au départ de Word 56](#_Toc226114024)

[Suppression des messages d’avertissement et d’erreur. 57](#_Toc226114025)

[But 57](#_Toc226114026)

[Complément et rappel 57](#_Toc226114027)

[Access. 57](#_Toc226114028)

[Excel 57](#_Toc226114029)

[Conseils 57](#_Toc226114030)

[Remerciements 58](#_Toc226114031)

[Divers 59](#_Toc226114032)

# Préambule

Je prétends qu'il faut avoir 3 qualités principales pour travailler en Windows.

1. La précision
2. L'Observation
3. La Curiosité

## La précision

Prenons Excel en exemple. Avez-vous déjà observé le changement de forme de la trace de la souris sur l’écran à l'approche de la "case courante" (A1 à l'ouverture). Un mouvement de quelques 10ème de millimètres vous provoque l'apparition de 3 traces différentes.







Qui dit changement de forme dit changement de rôle. Voyez les illustrations ci-dessus

Soyons donc précis

## L'observation.

Il y a tellement d'informations sur l'écran qu'il est parfois difficile de distinguer l'essentiel de l'accessoire. Observer permet de voir certaines choses insoupçonnables

Dans l'exemple ci-dessous vous constaterez que des cases contenant des chiffres sont sélectionnées. Observer bien la barre d'état en bas de votre écran, vous verrez le total des éléments sélectionnés.



Soyons donc observateur.

## La curiosité.

La curiosité est la qualité majeure permettant d'améliorer son savoir. L'on peut assouvir sa curiosité en généralisant l'usage intensif du bouton droit de la souris.

Tout objet, chose ou zone est presque toujours manipulable par le bouton droit de la souris.

Dans votre feuille Excel, cliquez avec le bouton droit de la souris sur la zone où la somme de vos éléments sélectionnés est indiquée. Sans cette curiosité, vous n'auriez peut-être jamais découvert que vous pouviez obtenir la moyenne de vos valeurs.



# Avant-propos important

Même si votre serviteur est quelque peu habitué à manipuler du code VBA, que ce soit en Excel, en Word voire dans une moindre mesure en Access, je ne tente jamais de me donner des qualités que je n’ai pas. Si dans certains cas j’ai été amené à étudier des livres contenant du code et des instructions de base, par la suite, et je vous invite à faire de même, je me suis inspiré, très souvent de lignes de code trouvées dans des livres et/ou sur Internet.

Le but principal de l’écriture d’une macro est de se simplifier la vie. Le but principal de ce cours n’est pas de faire de vous de grands connaisseur de VBA, mais à l’instar de votre serviteur de pouvoir comprendre des lignes de codes existantes et, le cas échéant, de pouvoir les adapter.

En espérant que ce cours répondra à vos attentes, bonne lecture et bon travail.

# Ma première Macro simple.

## Enregistrement d'une macro

### Préambule

Une macro est une suite d’instructions, simples ou complexes, permettant, soit d’exécuter des tâches répétitives de manières automatiques, soit de permettre à un utilisateur, éventuellement non expérimenté, d’accomplir des tâches complexes dont l’exécution est prise en charge, à sa place, par la Macro.

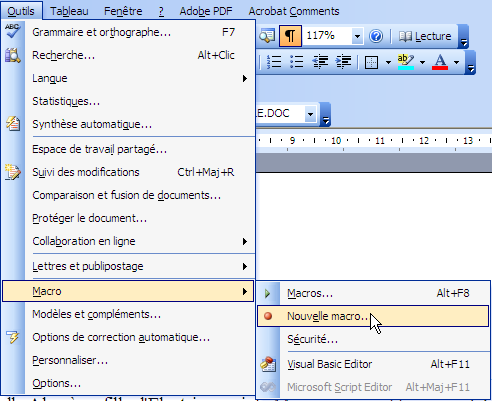
### But & Description de notre première macro simple

1. Insérer dans un document existant, par exemple « [Hercule.doc](HERCULE.DOC) » une macro permettant d’atteindre la page numéro 2.
2. Sélectionner les 3 premières lignes de celle-ci et en modifier les propriétés caractères, soit :
   1. Imposer le rouge comme étant la couleur de la police de caractère.
   2. Mettre en grasse, souligner et mettre en italique.

### Méthode

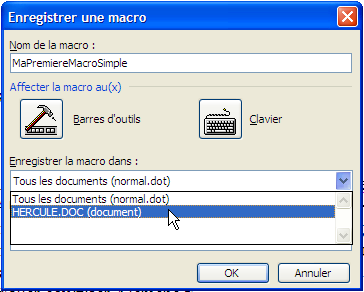
La méthode la plus pratique et conventionnelle afin d’enregistrer une macro est de profiter de l’outil d’enregistrement des macros livré avec Word.

Pour ce faire :



Enregistrement d’une nouvelle Macro

1. Utilisez le menu <**Outils**> - <**Macro**> - **Nouvelle Macro**>
2. Une boîte de dialogue apparaît vous invitant à nommer votre nouvelle Macro.



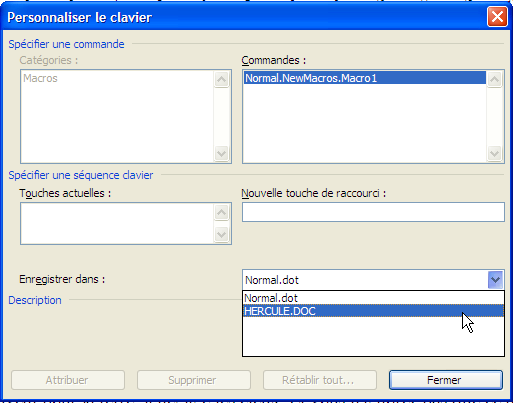
Enregistrer une Macro

1. Nous nommerons la Macro **« MaPremiereMacroSimple »**
2. Vous remarquerez l’absence d’accentué. Il s’agit d’une habitude, et non d’une contrainte. Puisque nous allons travailler avec du code écrit en Anglais, l’adoption de cette règle de travail nous permettra de faire des recherches efficaces et rigoureuses. De plus vous constaterez l’absence d’espace dans le nom de la Macro. Ils sont, à l’instar des [signets](#InsertionSignet), rigoureusement interdits dans l’appellation de celles-ci et en son temps remplacés par des caractères « soulignés bas » (Underscore) **« \_ »**. Les habitudes actuelles de travail nous ferons préférer de travailler sans ce signe, donc en « tout – attaché »

Afin d’être complet, vous pouvez utiliser des chiffres, mais votre nom doit commencer par une lettre.

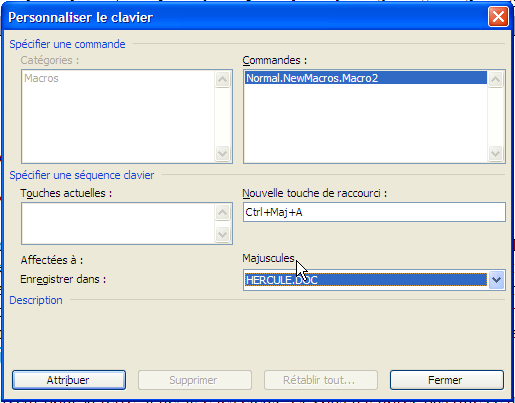
### Affectation à une séquence de Raccourci Clavier.

1. Une invite à placer un Raccourci Clavier se trouve juste au dessous de la zone de nom. Le Raccourci Clavier que nous installerons prendra toujours le pas sur le Raccourci Clavier existant dans Word. Les raccourcis claviers les plus courant se cachant derrière la séquence **« CTRL – Lettre minuscule »**, il sera sage de choisir une lettre majuscule (en l’occurrence, **« Maj + Ctrl + A »**)



Emplacement d’une macro

1. Comme nous le verrons plus bas, nous allons stocker cette macro dans le document en cours.
2. Dès que le raccourci clavier est choisi, il faut l’affecter au travers du bouton <Attribuer>

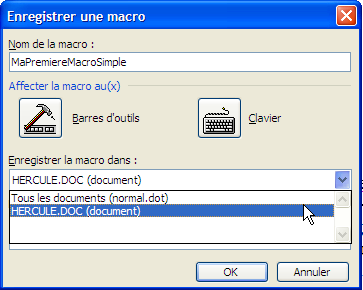


Personnalisation du clavier

1. Cette séquence de touche est déjà réservée à une fonction précise dans Word. A nous de voir si nous « l’écrasons » ou pas. Terminer cette phase en appuyant sur le bouton **<Fermer>**

### Où stocker une macro (Ce Classeur, Autre Classeur ou encore Classeur de macros Personnelles).

1. Une Macro peut s’enregistrer dans 2 emplacements différents :
   * **Un modèle de document** : Une Macro enregistrée dans ce modèle fera que la Macro sera disponible en permanence dans tous les documents de Word basés sur ce modèle. A utiliser en connaissance de cause et avec parcimonie.
   * **Le document en cours**: Cette méthode, en tant que débutant sera celle que nous privilégierons.



Où enregistrer une macro

### Début de l’enregistrement

Dès la phase préliminaire terminée, l’enregistrement de la macro commence. Ceci étant confirmé par la présence d’une cassette d’enregistrement qui accompagne le curseur, ainsi que par la présence d’une petite barre d’outils permettant l’arrêt de l’enregistrement.

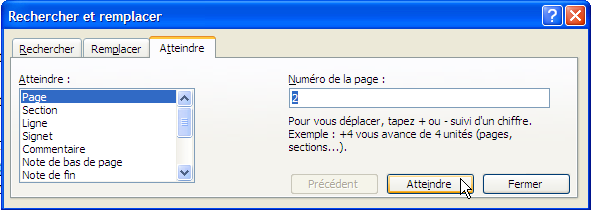


Bouton d’enregistrement

1. Toutes les actions ou opérations que vous allez entreprendre seront enregistrées par l’enregistreur de macro. Donc pas de mouvement inutile.

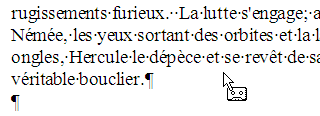
### Enregistrement du code.

1. Appuyer sur <F5> pour activer la fenêtre <Atteindre>
2. Ce raccourci fonctionne en Excel également



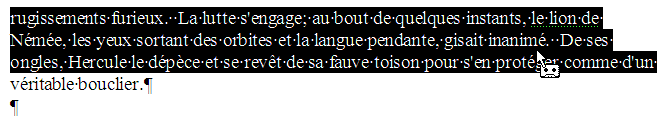
Rechercher et remplacer

1. Remarquez la collection d’objets que l’on peut atteindre. Nous nous contenterons cette fois-ci d’atteindre la page 2 parle bien de l’inscription d’un **« 2 »** dans la zone prévue à cet effet.
2. Cliquez sur le bouton <Atteindre>



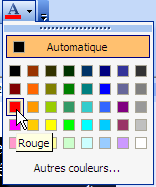
Atteindre Signet

1. Nous atteignons effectivement la page 2. Il ne nous reste plus qu’à sélectionner les 3 premières lignes.
2. A noter qu’il est impossible de sélectionner à la souris. Le texte doit être sélectionné au raccourci clavier, soit **<SHIFT>** + **<Flèche de déplacement>**



Atteindre Signet(réalisation)

1. L’action suivante va être de modifier les caractéristiques caractères de notre texte, soit :
   1. Mettre la police en rouge



Changement de couleur

* 1. Mettre en grasse



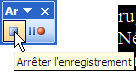
Mettre en grasse

* 1. Mettre en Italique



Mettre en italique

Aux fins de terminer l’opération, il nous faut cliquer sur le boutons d’arrêt de l’enregistrement de la macro.



Arrêt de l’enregistrement

1. Une erreur communément commise est de cliquer sur la croix noire qui permet de fermer une barre d’outils ou de la faire disparaître. Ne tombez pas dans ce piège.



Erreur à ne pas commettre

## Vérification du bon fonctionnement de notre première macro simple.

Pour ce faire quelques étapes :

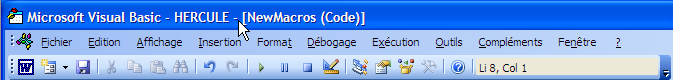
1. Sélectionner à nouveau le texte modifié.
2. Le mettre en style Normal (**<Shift> - <CTRL> - <N>**)
3. Retourner au début du texte
4. Lancer la macro (**<Shift> - <CTRL> - <A>**)

## Qu’avons-nous enregistré ?

Notre macro a été enregistrée sous forme de code exprimé en un langage de programmation, du VBA (pour Visual Basic for Application) dans un espace qui lui est spécifiquement réservé, un ***Module.***

Nous pouvons atteindre ce module en utilisant le raccourci clavier (**<Alt> - <F11>**)

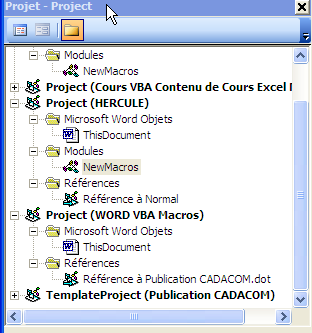
La fenêtre qui se présente à nous est une fenêtre d’application Windows.



Editeur Visual Basic

Cette application est une application qui tourne en parallèle avec les documents Word ouverts et qui se compose, outre des classiques lignes de titres, de Menus et d’outils de :

D'un explorateur de projet.



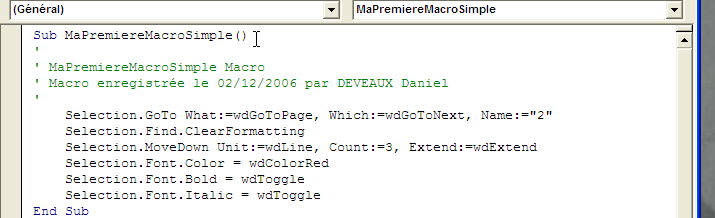
Explorateur de projet

D'une fenêtre de propriétés.



Fenêtre Propriétés

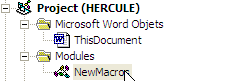
D'une fenêtre de code.



Premier Code

### Le Module

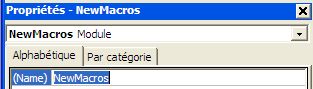
Le code s’est enregistré dans un module dont la présence est visible dans la fenêtre de propriétés



Module

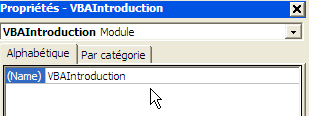
Le nom standard et provisoire qui lui est attribué est **«** **NewMacros »,** écrit en « **casse chameau** »

Ce nom peut-être changé grâce à la fenêtre de propriétés. Un double clic sur le nom provisoire :



Fenêtre des propriétés

Tapez le nouveau Nom que vous souhaitez et confirmez par <**ENTER**>



Renommer un module

Cela peut vous paraître insignifiant, mais lorsque vous aborderez des problèmes plus complexes, il vous sera plus agréable de partager votre travail en plusieurs « sous travaux » que vous stockerez dans des modules séparés afin d’en amélioré la lisibilité.

### Le code

Sub MaPremiereMacroSimple()

'

' MaPremiereMacroSimple Macro

' Macro enregistrée le 02/12/2006 par DEVEAUX Daniel

'

Selection.GoTo What:=wdGoToPage, Which:=wdGoToNext,

Name:="2"

Selection.Find.ClearFormatting

Selection.MoveDown Unit:=wdLine, Count:=3,

Extend:=wdExtend

Selection.Font.Color = wdColorRed

Selection.Font.Bold = wdToggle

Selection.Font.Italic = wdToggle

End Sub

Première Macro Simple

### Examen du Code généré

Soit de Manière plus générale :

**Début de Routine**

**‘** Commentaires

Action

Action

Action

Action

**Fin de Routine**

Notion de Routine – Sous-routine

L’on doit déclarer, automatiquement ou manuellement un début de boucle de Macro et une fin de celle-ci via les instructions <Sub> et <End Sub>.

### Les Commentaires

1. **N.B. :** Les commentaires sont toujours précédés par une **« ‘ »**, réminiscence de l’instruction **« REM »** du passé. Ces lignes de commentaires ne sont pas considérées comme du code, mais bien comme du simple texte explicatif.  
     
   L’avantage des remarques est que l’on peut les installer en fin de ligne de code, précédées, bien sûr par une **« ‘ »** (voir ci-dessous)

Selection.GoTo What:=wdGoToPage, Which:=wdGoToNext,

Name:="2" ‘Cette instruction me fait aller à la page 2

Commentaire et sélection de ligne

1. **N.B. :** Pour des raisons d’encombrement les commentaires n’apparaîtront plus dans les reproductions de code.

# La correction automatique – Comment en améliorer la puissance.

## Préambule

En Word l’outil **« Correction automatique »** est méconnu, vaut la peine d’être vu et d’être amélioré.

Parmi les corrections automatiques standards fournies avec Word, signalons que lorsque vous tapez, en français :

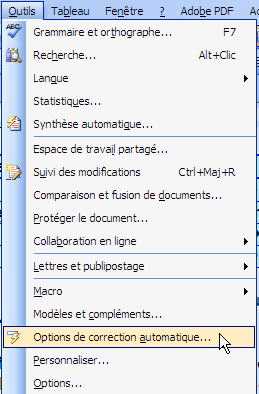
|  |  |
| --- | --- |
| Ce que vous saisissez | Ce que vous obtenez |
| a t on | a-t-on |
| amorticement | Amortissement |
| ==> | 🡺 |
| :-) | ☺ |

Cet outil, je le répète, sous utilisé, peut-être complété de telle manière à créer votre propre sténographie.

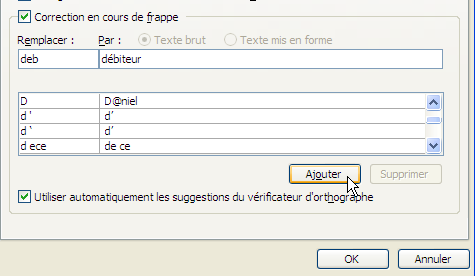
Sur les différents PC que j’utilise, lorsque par exemple je tape DD, j’obtiens Daniel DEVEAUX. Quelques exemples, ci-dessous.

|  |  |
| --- | --- |
| Ce que vous saisissez | Ce que vous obtenez |
| DD | Daniel DEVEAUX |
| ^m | même |
| cad | c’est-à-dire |
| EP | Parlement Européen |

Pour alimenter cette liste de correction automatique, il faut aller dans le menu **<Outils> - <Options de correction automatique>** et remplir manuellement les cases idoines soit :



Option de correction automatique



Ajout à la Correction automatique

Puis cliquer sur **<Ajouter>**, puis sur **<OK>**

Cela fonctionne mais est une tâche fastidieuse, répétitive et donc pouvant être prise en partie en charge au moyen d’une macro qui va nous permettre d’améliorer sensiblement la saisie de corrections automatiques

1. Toutes les corrections automatiques que vous saisissez sont stockées dans un fichier au format texte nommé MSO1036.ACL (1036 étant le code de la langue que vous utilisé, en l’occurrence le français, la liste de ces codes étant disponible sur le site [www.servezvous.be](http://www.servezvous.be)) et fait partie de votre profil. Cette correction vous suit, peu importe la machine sur laquelle vous vous logez. Dans le cas d’un ordinateur familial, donc « stand alone », le chemin typique de ce fichier est :

C:\Documents and Settings\Daniel\Application Data\Microsoft\Office\MSO1036.ACL

Ce commentaire ayant pour but de vous inciter à faire une copie de sauvegarde de ce fichier afin de ne pas perdre le fruit de votre travail lors d’un « crash » éventuel.

## Le scénario proposé.

Au sein de notre texte, sélectionner un mot « Hercule » par exemple, par un jeu de raccourcis clavier faire apparaître une boîte de dialogue vous invitant à introduire le code, « her » par exemple afin de provoquer l’apparition automatique du mot « Hercule »

## La réalisation

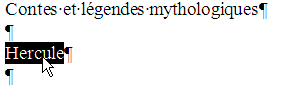
### Sélection du texte

Au sein du document sélectionnez le mot Hercule en prenant bien soin de ne ***pas*** sélectionner une marque de fin de paragraphe, ce qui ferait conserver dans la correction automatique la mise en forme de cette sélection. Par exemple si le texte était en rouge, souligné et mis en grasse, si vous faites appel à cette correction automatique, vous verriez apparaître les mêmes attributs de caractères.

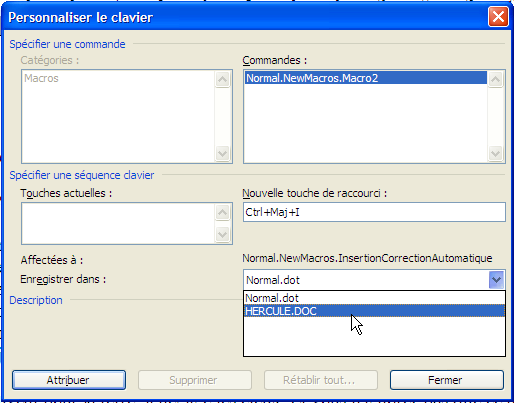
De plus, en ne sélectionnant pas la marque de fin de paragraphe, vous bénéficierez de votre correction automatique dans les autres applications Office telles qu’Excel, PowerPoint, Access et Publisher.

### Enregistrement de la Macro.

1. Le mot « Hercule » étant sélectionné, lancer l’enregistreur de Macro, saisissez le raccourci clavier **<Shift> - <CTRL> - <I>** au travers de l’attribution d’un raccourci à votre clavier.

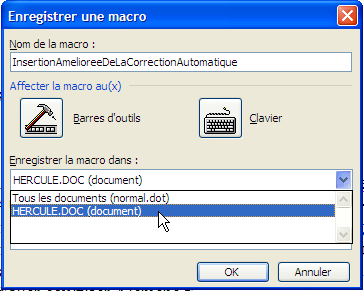


Utilisation de la Correction Automatique



Eviter les modèles

1. Par défaut, l’enregistreur de Macros Word va tenter d’enregistrer votre macro dans un modèle. Pour cette session, nous ne travaillerons que pour le document en cours.



Enregistrement Macro

1. Saisissez le nom de votre Macro. N’hésitez pas à utiliser des appellations longues et explicites.

En l’occurrence, je suggère <InsertionAmelioreeDeLaCorrectionAutomatique>.

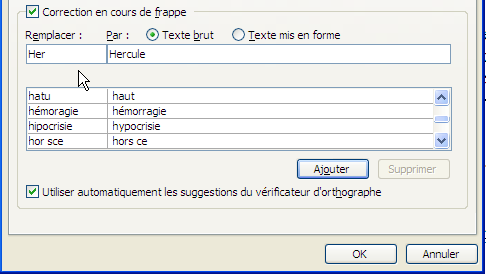
Cliquez sur <OK>

1. Exécutez votre « manœuvre »sous le couvert de l’enregistrement de votre Macro, soit <Outils> - <Option de correction automatique>

Constatez que votre texte sélectionné se trouve déjà dans la boîte de dialogue.

Ayant pris la précaution de ne pas sélectionner une marque de fin de paragraphe vous constatez que votre correction sera en texte brut, donc non mise en forme.

1. Saisissez **« Her »** dans la zone prévue à cet effet
2. Cliquez sur **<Ajouter>** puis **<OK>**



Ajout à la Correction Automatique

1. Arrêtez l’enregistrement de votre Macro

### Analyse et modificationdu Code

Basculez vers l’éditeur de code grâce à **<Alt> - <F11>**

Vous devriez découvrir le code suivant :

Sub InsertionAmelioreeDeLaCorrectionAutomatique()

'

' InsertionAmelioreeDeLaCorrectionAutomatique Macro

' Macro enregistrée le 02/12/2006 par DEVEAUX Daniel

'

AutoCorrect.Entries.Add Name:="Her", Value:="Hercule"

With AutoCorrect

.CorrectInitialCaps = True

.CorrectSentenceCaps = True

.CorrectDays = True

.CorrectCapsLock = True

.ReplaceText = True

.ReplaceTextFromSpellingChecker = True

.CorrectKeyboardSetting = False

.DisplayAutoCorrectOptions = True

.CorrectTableCells = True

End With

End Sub

Comme signalé plus haut enlevons les commentaires, soit

Sub InsertionAmelioreeDeLaCorrectionAutomatique()

AutoCorrect.Entries.Add Name:="Her", Value:="Hercule"

With AutoCorrect

.CorrectInitialCaps = True

.CorrectSentenceCaps = True

.CorrectDays = True

Retrait

.CorrectCapsLock = True

.ReplaceText = True

.ReplaceTextFromSpellingChecker = True

.CorrectKeyboardSetting = False

.DisplayAutoCorrectOptions = True

.CorrectTableCells = True

End With

End Sub

#### Les boucles

Le principe de début de boucle et de fin de celle-ci a été décrit plus haut.

Constatez que la boucle SUB se termine par un End Sub.

Dès que l’on commence à travailler dans une boucle, on fait précéder l’ensemble de son code par une marque de tabulation de façon à travailler en retrait à l’intérieur de chaque déclaration de boucle.

Cette démarche n’est absolument pas obligatoire, mais augmente de manière sensible la lisibilité de longs codes.

A titre d’exemple voyez le code entre With et End With.

#### Les propriétés

Au sein de la boucle

Sub InsertionAmelioreeDeLaCorrectionAutomatique()

AutoCorrect.Entries.Add Name:="Her", Value:="Hercule"

With AutoCorrect

.CorrectInitialCaps = True

.CorrectSentenceCaps = True

.CorrectDays = True

.CorrectCapsLock = True

.ReplaceText = True

.ReplaceTextFromSpellingChecker = True

.CorrectKeyboardSetting = False

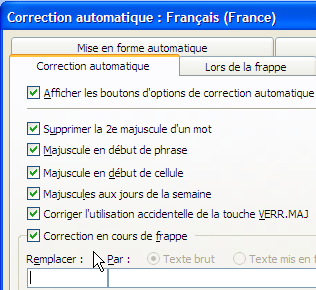
.DisplayAutoCorrectOptions = True

.CorrectTableCells = True

End With

End Sub

Il est à remarquer que dans la boucle With et End With, l’on retrouve, sous forme de propriétés toutes les options de la correction automatique, soit :



Les propriétés

Ceci sans que ma liste soit exhaustive…

Toutes ces propriétés ne sont qu’une nouvelle déclaration de ce qu’aurait appliqué Word, par défaut. Il s’agit donc de ce que l’on appelle communément du code inutile.

Puis-ce qu’il est inutile, nous allons le supprimer afin de rendre notre code plus simple.

Soit :

Sub InsertionAmelioreeDeLaCorrectionAutomatique()

AutoCorrect.Entries.Add Name:="Her", Value:="Hercule"

With AutoCorrect

.CorrectInitialCaps = True

.CorrectSentenceCaps = True

.CorrectDays = True

.CorrectCapsLock = True

.ReplaceText = True

.ReplaceTextFromSpellingChecker = True

.CorrectKeyboardSetting = False

.DisplayAutoCorrectOptions = True

.CorrectTableCells = True

End With

End Sub

Ce qui nous laisse :

Sub InsertionAmelioreeDeLaCorrectionAutomatique()

AutoCorrect.Entries.Add Name:="Her", Value:="Hercule"

End Sub

A la lecture de ce code finalement très simple, l’on peut comprendre que cette ligne de code unique ne sera, en l’état, dédicacée à la seule déclaration confirmée à chaque exécution de celui-ci que le texte « Her » sera remplacé par « Hercule ». Ceci n’est pas notre souhait.

Notre souhait est d’optimiser l’alimentation de la correction automatique.

Nous devons partir de n’importe quel texte sélectionné (sans marque de fin de paragraphe) et le faire remplacer par le code que nous souhaitons.

Il faut donc que notre macro nous interpelle, nous interrompe afin de nous demander quel code l’on souhaite insérer dans la correction automatique.

#### L’instruction **« InputBox »**

L’instruction **« InputBox »** permet de faire apparaître à l’écran une boîte de dialogue qui comportera une question (« Prompt »), un titre et éventuellement une valeur par défaut.

Placez-vous de telle manière à saisir le texte de façon à ce que votre code soit identique au texte ci dessous

Sub InsertionAmelioreeDeLaCorrectionAutomatique()

Dim MaSaisie

MaSaisie = InputBox(“Votre code s.v.p. ?”, “Alimentation de la correction autoamtique”)

AutoCorrect.Entries.Add Name:="Her", Value:="Hercule"

End Sub

L’Instruction InputBox 4

En ajoutant cette instruction, vous obtenez, à l’écran, une boîte de dialogue qui vous demande l’introduction d’une valeur.

Le système VBA doit savoir où stocker l’information. Cette information ira se placer dans un registre qu’on lui aura désigné. En l’occurrence **« MonContenu »** dont on aura, au préalable déclaré l’existence par l’introduction d’une instruction **« Dim »**.

Cette instruction (non obligatoire) possède le triple avantage de confirmer l’existence de la variable, d’éventuellement lui confier un rôle comme nous verrons dans une autre partie du cours et de remettre le contenu de cette variable à **« Vide »** ou **« Empty »**

Le code VBA dans ses versions modernes présente des aides instantanées non négligeables. Dès que vous avez écrit le contenu suivant **« MonContenu=InputBox(»** une fenêtre interactive apparaît :



Aide Interactive

La proposition en caractère gras est la proposition courante (celle en cours).

Celle(s) entre crochet **« [xxx] »** est (sont) facultative(s). Si les différents arguments de l’instruction sont des textes, ceux-ci doivent être écrits entre guillemets. Si par contre les Arguments sont le contenu de variables, en ce cas les guillemets sont à proscrire.

A noter que nous sommes du côté « Américain » de l’application, et que dès lors le séparateur d’argument d’une instruction telle que **« InputBox »** est la virgule **« , » et non pas le point – virgule « ; »**

Exécutons la Macro. On peut s’apercevoir que la boîte de dialogue apparaît comme souhaité.

Mis à part que pendant l’exécution de l’instruction, le contenu de notre réponse sera stocké dans la variable « MaSaisie », la macro est toujours dédicacée au même but soit remplacer « Her » par « Hercule ».

Pour parvenir à nos fins il nous faudra donc modifier le code de telle manière à ce que le texte sélectionné, quel qu’il soit, soit remplacé par le contenu de notre réponse à l’invite de notre boîte de dialogue (« InPutBox »)

Il faudra donc changer le code de telle manière à ce qu’il soit identique au suivant :

Sub InsertionAmelioreeDeLaCorrectionAutomatique()

Dim MaSaisie

MaSaisie = InputBox(“Votre code s.v.p. ?”, “Alimentation de la correction automatique”)

AutoCorrect.Entries.Add Name:= MaSaisie, Value:=

Selection

End Sub

L’Instruction InputBox 5

Faisons l’essai de notre macro en sélectionnant le mot « Amphitryon », utilisons notre séquence de raccourci clavier, c’est-à-dire (<Shift> - <CTRL> - <I>), la boîte de dialogue apparaît, répondons « Amp », cliquons sur <OK>.

Allons dans n’importe quelle partie de notre texte, dactylographions le texte « Amp »suivit d’un espace, et Amphitryon apparaît.

A vous d’imaginer la vie qui va avec ceci…

#### Remarque importante

Si l’on saisit le texte « amp » (le « A » n’étant pas en majuscule) la correction ne se fait pas. Si dans le code à remplacer l’on a saisi une ou des majuscules, il faut respecter ce choix lors de son utilisation, sous peine de ne pas voir notre macro fonctionner

# Atteindre un signet à l’ouverture d’un document.

## Les signets

### Préambule

Les signets sont à Word ce que sont les zones nommées à Excel.

S’il nous était possible de travailler en bureau sans papier (paperless desktop) comme j’arrive à le faire dans mon activité professionnelle, la notion de numérotation de page est quelque chose de purement subjectif.

Le fait que nous ajoutions des pages ou que nous en retirions au dessus du dernier travail d’Hercule, ferait bouger celui-ci de numéro de page. Bien entendu en hiérarchisant nos titres en titre de niveau 1,2,…, 9, nous pourrions organiser une table des matières et, le cas échéant la remettre à jour si changement, il y a.

Si à l’ouverture de mon document je souhaite atteindre directement le dernier travail d’Hercule, « le Chien Cerbère », il me faudrait savoir à quelle page il se trouve.

Si j’équipe ce sujet d’un signet, celui-ci étant dynamique, je pourrais me permettre de l’ignorer.

### Insertion d’un signet

Placez-vous devant le « L » du paragraphe « Le Chien Cerbère », cliquez sur **<insertion>** - <Signet…>



Insertion d’un signet

Inscrivez le nom que vous souhaitez, par exemple « MonSignet » et cliquez sur <OK>

Les règles de baptême de signets sont les mêmes que celles pour les noms de Macro ([Voir](#VersSignet))

### Atteindre un signet.

Exécutez <CTRL> - <Home>) afin d’atteindre le début du document.

Appuyer sur <F5> pour avoir accès à <Atteindre>.

Sur côté gauche de cette fenêtre distinguez l’instruction <Signet>. Cliquez dessus. Le nom de votre seul et unique signet apparaît, sélectionnez-le et cliquez sur atteindre.

Refaite l’expérience en intercalant autant de pages que vous le souhaitez, avant « Le Chien Cerbère », vous atteindrez cet endroit sans soucis. Quand je vous disais que la notion de numéro de page était subjective…

## Atteindre un signet sous forme de Macro.

Un simple enregistrement, sans affectation clavier suffit.

Suivez les instructions décrites plus [haut](#EnregistrementMacro) pour enregistrer votre macro que vous baptiserez « AtteindreMonSignet », en casse chameau, bien sûr.

1. Veillez à enregistrer votre macro dans le document en cours et non dans un modèle

Après suppression des remarques et du code inutile, l’inspection du code vous donnera le texte suivant :

Sub AtteindreMonSignet()

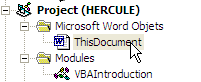
Selection.GoTo What:=wdGoToBookmark, Name:="MonSignet"

End Sub

## Exécuter cette macro à l’ouverture du document.

Afin d’atteindre ce but il faut faire appel à la routine (macro) que nous venons d’enregistrer dans un endroit tout à fait particulier.

Dans l’éditeur de code VBA repérez l’endroit suivant :



Emplacement de code au démarrage

et faites-y un double clic. La page de code apparaissant devrait être vide.

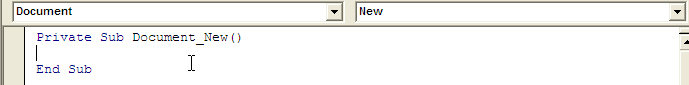
Cette page de code va recenser les actions à entreprendre lors de divers événements du document, comme par exemple son ouverture.

Repérez l’endroit suivant :



Déclaration dans le document

Dès que vous aurez cliqué sur <Document>, vous serez en présence de cette déclaration :



Événement par défaut

Celle-ci signifiant : « Que va-t-on faire lors de la création d’un nouveau document ? »

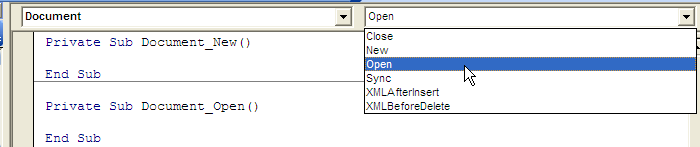
Ceci ne nous intéresse pas. L’événement qui nous intéresse est l’ouverture de ce document.

Cliquez à droite, sur la liste déroulante des événements disponibles et choisissez l’événement Open.

Aux fin de se débarrasser d’informations inutiles, nous allons supprimer la boucle inutile, soit :

Private Sub Document\_New()

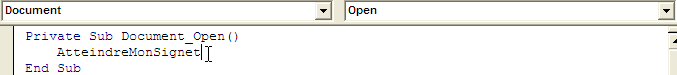
End Sub



Changer d’événement

Dans la boucle faisant référence à l’événement <Open> l’on va rajouter l’instruction <AtteindreMonSignet>.

1. Soyons paresseux, le PC cette utile mais stupide machine est notre esclave qui, par définition doit travailler pour nous et non pas le contraire, nous pour la machine.  
     
   L’éditeur VBA est déjà « au courant » que <AtteindreMonSignet> existe. Il suffit de saisir quelques premiers caractères significatifs ce cette instruction et de faire suivre par un <CTRL> - <Espace>, le restant de la saisie se fera pour nous. Il en ira de même lorsque l’on aura la déclaration de l’existence d’un registre au travers de l’instruction <Dim>. Il en ira également de même pour les commandes internes du VBA. Pour l’instruction InputBox, par exemple, la saisie de <Inp> suivi d’un <CTRL> - <Espace> est suffisante. Soyons donc paresseux, cela nous fera travailler plus rapidement et avec un risque amoindri de saisie d’erreur.



Déclaration

1. Remarquez la présence de l’instruction <Private> avant la sub. Si vous faites appel à une macro qui n’est pas présente dans votre document courant, VBA va chercher si elle n’existe pas dans d’autres documents ouverts dans la session. Si VBA trouve la macro appelée, elle sera exécutée, sauf s’il y a la présence de l’instruction <Private> qui fera que cette macro ne s’exécutera que pour le document en cours.

## Exécution de la Macro

Basculez vers le document, sauvez-le, puis ouvrez-le à nouveau.

Le curseur, ou point d’insertion se déplace automatiquement vers l’endroit souhaité.

# Macro faisant la liste des signets d’un document.

L’on peut très bien être en présence d’un document dont on est l’auteur ou d’un document que l’on reçoit d’une source extérieure.

La liste des signets présents dans le document est consultable via le menu <Insertion> - <Signet>, mais pourquoi ne pas les faire imprimer à la fin d’un document et d’y apporter des commentaires aux fins de se souvenir de leur rôle respectif.

## Le code de base

Sub ListeSignets()

Dim MonSignet As Bookmark

With ActiveDocument.Content

.Collapse Direction:=wdCollapseEnd

.InsertBreak Type:=wdPageBreak

For Each MonSignet In ActiveDocument.Bookmarks

.InsertAfter Signet.Name & vbCr

Next MonSignet

End With

End Sub

Comme vous pouvez le constater, ce code est loin d’être long à écrire, par contre il fait appel à de nouvelles notions.

## L’instruction Dim

L’instruction Dim ne sert pas uniquement à déclarer l’existence d’un registre, à lui conférer la propriété <Empty>, mais également à lui assigner un rôle bien particulier. Dans ce cas-ci, le registre « MonSignet » ne pourra contenir que des noms de signets.

## L’instruction Collapse

Réduit une plage ou désactive une sélection en son emplacement de début ou de fin. Après la réduction d'une plage ou la désactivation d'une sélection, les emplacements de début et de fin forment un seul emplacement.

Cette instruction peut vous paraître quelque peu nébuleuse, elle n’en est pas moins importante, voire primordiale. On pourrait la comparer à une cellule d’origine en Excel. Nous en reparlerons [plus avant](#InstructionCollapse) dans ce cours

### Syntaxe

Expression.Collapse(Direction)

La notion d’expression est obligatoire. Cette expression fait-elle appel à une ligne, un paragraphe, à tout le document, etc.

La Direction est un argument facultatif. Direction dans laquelle sera réduite la plage ou désactivée la sélection. Il peut s'agir de l'une des constantes WdCollapseDirection suivantes : wdCollapseEnd ou wdCollapseStart.

WdCollapseStart est la valeur par défaut.

### Son rôle dans ce cas-ci.

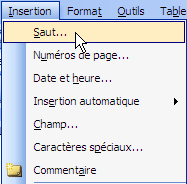
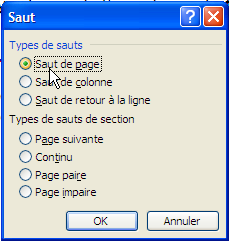
L’instruction complète serait :

ActiveDocument.Content.Collapse Direction:=wdCollapseEnd

Cette instruction va envoyer le point d’insertion à la fin du document.

## L’instruction InsertBreak

Plus simple que la précédente : insère un saut tel qu’on peut le faire manuellement au travers du menu

Les sauts les sauts (suite)

### Syntaxe

Expression.InsertBreak(Type)

La notion d’expression est obligatoire. Cette expression fait-elle appel à une ligne, un paragraphe, à tout le document, etc.

Le Type détermine le type de saut qui doit être inséré et est facultatif(voir liste dans la figure ci-dessus).

Les types sont wdPageBreak (saut de page), wdColumnBreak (saut de colonne), wdSectionBreakNextPage (saut de page associé à un saut de section : typique lorsque l’on veut travailler avec des en-têtes et/ou des pieds de pages différents dans le courant d’un document), wdSectionBreakContinuous (Saut de section continu), wdSectionBreakEvenPage (Saut de section associé à un saut de page paire), wdSectionBreakOddPage (Saut de section associé à un saut de page impaire), wdLineBreak (Saut de ligne sans création d’un nouveau paragraphe)

La valeur par défaut étant wdPageBreak, on aurait pu l’omettre en ce cas-ci.

### Son rôle dans ce cas-ci.

L’instruction complète serait :

ActiveDocument.Content.InsertBreak Type:=wdPageBreak

Cette instruction insère un saut de page.

## La boucle For Each … Next

Cette boucle détecte dans la sélection, en l’occurrence l’entièreté du document chaque signet

### Syntaxe

For Each Element In group Next [element]

Inspection de chaque élément dans une collection (Group) entraînant nécessairement une action. Next fait appel à l’élément suivant de la collection.

### Son rôle dans ce cas-ci.

L’instruction complète est :

For Each MonSignet In ActiveDocument.Bookmarks

.InsertAfter Signet.Name & vbCr

Next MonSignet

La variable « MonSignet » est affublé au fur et à mesure de l’inspection de l’entièreté du document du nom du signet éventuellement rencontré.

La deuxième ligne inscrit le nom du signet en question et le chaîne (caractère &) avec un caractère CR, soit « retour chariot » (en anglais « Carriage Return ») faisant référence à une vieille expression informatique qui elle même faisait appel à l’image du retour chariot d’une machine à écrire mécanique ou électrique. La constante « vbCr » représente un saut de paragraphe qui implicitement inclus l’insertion d’une ligne complémentaire (Ligne Feed)

La dernière instruction complète la boucle en faisant appel à l’élément suivant de la collection.

# Envoyer un document par Courriel.

Tout en sachant que l’on peut envoyer un message en pièce attachée à un courriel par sa messagerie déclarée par défaut au travers du menu <Fichier> - <Envoyer vers…> - <Destinataire du message (en tant que pièce jointe)…>, il est peut-être bon de savoir que l’on peut inclure cette action dans du code VBA

## Le code

Sub SetMailAttachOption()

Application.Options.SendMailAttach = True 'Or False

ActiveDocument.SendMail

End Sub

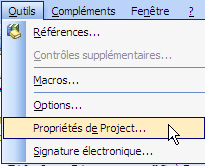
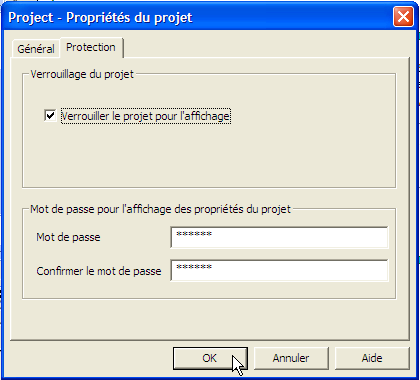
## Commentaires

Ils se limitent à pas grand-chose, l’instruction ActiveDocument.SendMail parlant d’elle même, je pense.

Il est quand même bon de savoir que l’on peut empêcher d’envoyer le document par courriel grâce à la ligne d’instructions précédente.

Cette information ne serait pas complète si je n’ajoutais pas le commentaire suivant :

Il est toujours possible de protéger son code au travers des possibilités de l’éditeur VBA.

Propriétés de projet Le mot de passe

Faites l’essai, fermez le document, puis ouvrez-le à nouveau et tenter de modifier le code.

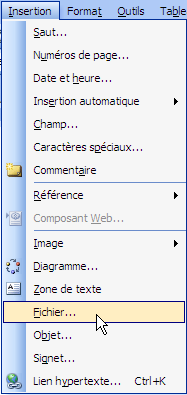
# Ouvrir un insérer un document d’un autre format.

A supposer que nous soyons au sein d’un document tel que le présent document et que je souhaite y insérer un autre fichier, texte par exemple, qui se trouverait sur mon « Bureau »[[1]](#footnote-1)

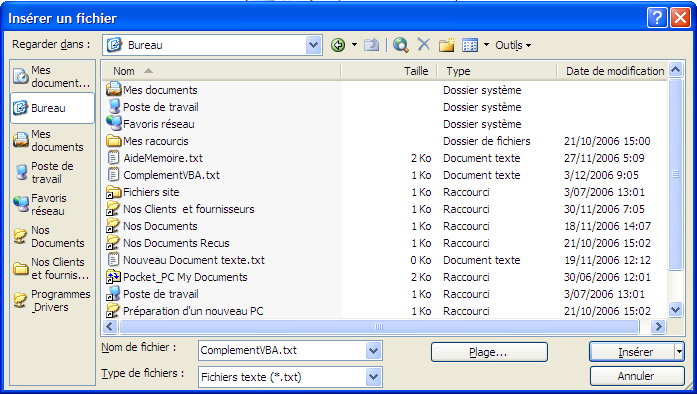
## La réalisation

Lancez l’enregistreur de macro et nommez votre macro « InsertionFichierExterne », puis sous le couvert de l’enregistrement, cliquez sur <Insertion> - <Fichier…>.

L'insertion du contenu d'un document texte, ou autre n'est pas tellement compliquée.

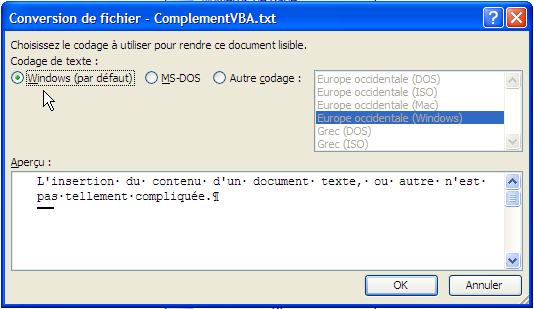


Insertion Fichier



Insertion de quel type

Modifiez le <Type de fichier>, cliquez sur le bouton <Bureau>, sélectionnez le fichier et cliquez sur <Insérer>



Conversion de fichier

Choisissez la conversion <Windows (par défaut) et cliquez sur <OK>

Le résultat pourrait ressembler au contenu ci-dessous :

L'insertion du contenu d'un document texte, ou autre n'est pas tellement compliquée.

N’oubliez pas d’arrêter l’enregistrement de la macro.

## Le code

Sub InsertionFichierExterne()

ChangeFileOpenDirectory "C:\Documents and Settings\Daniel\Bureau\"

Selection.InsertFile FileName:="ComplementVBA.txt", Range:="", \_

ConfirmConversions:=False, Link:=False, Attachment:=False

End Sub

A noter :

l’instruction < ChangeFileOpenDirectory> suivi de son argument

L’on pourrait intervenir dans le code pour demander à l’intervenant le chemin du document en question suivi d’une demande de nom de fichier. Ceci pourrait donner :

Sub InsertionFichierExterne()

Dim MonChemin, MonFichier

MonChemin = InputBox("Le chemin de votre fichier, svp ?", "Saisie du Chemin")

MonFichier = InputBox("Le nom du Fichier (avec son extension), svp ?", "Saisie du nom de fichier")

ChangeFileOpenDirectory MonChemin

Selection.InsertFile FileName:=MonFichier, Range:="", \_

ConfirmConversions:=False, Link:=False, Attachment:=False

End Sub

l’instruction < Selection.InsertFile FileName>

l’instruction < ConfirmConversions>

# Détection d’un mot particulier et surligner le paragraphe concerné.

## Processus

Ouvrez le document [Hercule.doc](HERCULE.DOC), déjà utilisé plus haut

Installez le code suivant dans le module existant.

Cette macro va chercher dans la sélection, soit tout le document le mot « Hercule » et va surligner chaque paragraphe trouvé en jaune.

## Le code

Sub DetectionHercule()

Selection.WholeStory

Selection.HomeKey Unit:=wdStory

Selection.Find.ClearFormatting

With Selection.Find

.Text = "Hercule"

While .Execute

Selection.Paragraphs(1).Range.Select

Selection.Range.HighlightColorIndex = wdYellow

Selection.Collapse wdCollapseEnd

Wend

End With

Exit Sub

End Sub

## Explication du Code

### Selection.WholeStory

Cette instruction sélectionne l’entièreté du document

### Selection.HomeKey Unit:=wdStory

Cette méthode <HomeKey Unit> ramène le curseur à une position qui en l’occurrence est le début du document, le « système VBA » retenant le fait que l’on a démarré de la sélection totale du document.

Annule la mise en forme de texte et de paragraphe d'une sélection ou celle spécifiée dans le cadre d'une opération de recherche ou de remplacement.

#### Selection.Find.ClearFormatting

ClearFormatting

Annule la mise en forme de texte et de paragraphe d'une sélection ou celle spécifiée dans le cadre d'une opération de recherche ou de remplacement.

Pour vous assurer que les attributs de mise en forme ne figurent pas parmi les critères de recherche ou de remplacement, exécutez cette méthode avant d'effectuer une recherche ou un remplacement.

### With Selection.Find .Text = "Hercule

Permet de trouver le mot Hercule. A noter qu’aucune des options de recherche n’est activée

### While .Execute

Sous-entendu « Tant que » (While) tu l’as trouvé, tu exécutes.

Une boucle « Tant Que » doit se terminer par une boucle « Fin de Tant Que ».

A noter que jusqu’à la version 6 de Word on pouvait écrire le VBA dans la langue de l’utilisateur. Les problèmes techniques s’accumulant, Microsoft a du faire marche arrière pour revenir à une rédaction en Anglais du code.

La boucle est donc maintenant : While … Wend

#### Selection.Paragraphs(1).Range.Select

Sélectionne le paragraphe en cours.

### Selection.Range.HighlightColorIndex = wdYellow

Surligne le paragraphe avec la couleur choisie, le jaune en l’occurrence.

Dans la collection des couleurs, une instruction permet de supprimer la surbrillance : wdNoHighlight

A titre d’exemple, voici une liste exhaustive des couleurs de la propriété WdColorindex.



Les couleurs de surlignage

### Selection.Collapse wdCollapseEnd

Tout en sachant que nous en avons parlé [au début de ce fascicule](#DefinitioInstructionCollapse), cette instruction est fondamentale dans le déroulement de code VBA sous Word.

Nous partons d’une sélection en l’occurrence un paragraphe. Encore faut-il exprimer notre volonté de continuer l’exécution du code après exécution de l’instruction précédente, et à partir de la fin de la sélection affectée par cette instruction.

# Petits utilitaires pratiques

## Comment en VBA compter le nombre de lignes et/ou de mots d'un document ?

### Préambule

Des Firmes et/ou des Institutions comme le Parlement Européen pourraient avoir recours à des traducteurs payés à la ligne, voire au mot.

Cette macro permettrait d’inclure le montant dans une facture ou un devis.

### Installation du code

Sub nblignes()

Dim NombreLignes, NombreMots

NombreLignes =

ActiveDocument.BuiltInDocumentProperties(wdPropertyLines)

NombreMots =

ActiveDocument.BuiltInDocumentProperties(wdPropertyWords)

MsgBox "Ce document contient " & NombreLignes & " lignes."

MsgBox "Ce document contient " & NombreMots & " mots."

End Sub

### A noter

Une nouvelle instruction <MsgBox> qui permet de faire afficher une boîte à message à l’écran, l’utilisateur devant cliquer sur le bouton <OK>

## Comment, en VBA, compter le nombre de pages d'un document ?

### Installation du code

Public Sub test()

Dim NbPages As Integer

ActiveDocument.Bookmarks("endofdoc").Select

NbPages = Selection.Information(wdActiveEndPageNumber)

End Sub

## Liste de toutes les polices utilisées dans un document.

### Installation du code

Dans un module de ce document, installer le code suivant :

Sub ListePolice()

Dim MaListe, MaFonteDeCaractere

For Each MaFonteDeCaractere In FontNames

Selection.WholeStory

Selection.HomeKey Unit:=wdStory

Selection.Find.ClearFormatting

With Selection.Find

.Font.Name = MaFonteDeCaractere

End With

If Selection.Find.Execute = True Then

MaListe = MaListe & Chr(13) &

MaFonteDeCaractere

End If

Next MaFonteDeCaractere

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.Style = ActiveDocument.Styles("Titre 1")

Selection.TypeText Text:="Liste des Polices de

caractères"

Selection.TypeParagraph

Selection.TypeText Text:=MaListe

End Sub

## Explication du code

### Méthodologie.

A la lecture de ce code vous constaterez que l’on vérifie la présence dans ce document l’utilisation ou non de chacune des polices mises à ma disposition. Dès qu’une police utilisée est trouvée, son nom va alimenter une liste de noms de fontes de caractères qui sera ajoutée à la fin du document.

### L’instruction « If »

La syntaxe du code VBA de la Fonction Conditionnelle à une condition est donc :

If...Then...Else...End If

Soit de manière plus générale

**If** Condition **Then**

Action à réaliser si condition réalisée

**Else**

Action à réaliser si condition non vérifiée

**End If**

If…Then…Else - Généralités 6

Dans cet exemple la possibilité « Else » n’est pas utilisée.

### Selection.Style = ActiveDocument.Styles("Titre 1")

Permet de déclarer dans quel style sera inséré le paragraphe suivant. A noter que dans le présent document le style du paragraphe qui suivra le style « Titre 1 » est le style « Normal »

## Comment tester toutes les polices avec une phrase type que l'on aurait choisie ?

### Installation du code

Sub TestDesPolicesInstallees()

Dim MaFonteDeCaractere

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.Style = ActiveDocument.Styles("Titre 1")

Selection.TypeText Text:="Liste des Polices de caractères"

Selection.TypeParagraph

For Each MaFonteDeCaractere In FontNames

With Selection

.InsertAfter vbTab & MaFonteDeCaractere & vbCrLf

.InsertAfter "Servez un whisky au vieux juge blond qui fume la pipe"

.Font.Name = MaFonteDeCaractere

.Font.Size = 12

.InsertAfter vbCrLf & vbCrLf

End With

Selection.EndKey Unit:=wdStory

Next MaFonteDeCaractere

End Sub

## Imprimer la description des styles utilises

### Installation du code

Sub DescriptionDesStylesUtilises()

Dim MonStyle As Style

Selection.EndKey Unit:=wdStory

Selection.TypeParagraph

Selection.Style = ActiveDocument.Styles("Titre 1")

Selection.TypeText Text:="Description des styles personnels"

Selection.TypeParagraph

For Each MonStyle In ActiveDocument.Styles

If MonStyle.BuiltIn = False Then

ActiveDocument.Content.InsertAfter vbCr & "Style utilisateur " & MonStyle & " : " & MonStyle.Description & vbCr

End If

Next MonStyle

End Sub

### Remarque

L’instruction « MonStyle.BuiltIn = False » détecte la présence de styles personnalisés. Un « non test » de cette propriété vous donnera la liste de tous les styles présents dans le document.

## Epurer votre document des styles non utilisés.

### Installation du code

Sub NettoyerStylesPersonalisesEtNonUtilises()

Dim MonStyle, MonCompteur

For Each MonStyle In ActiveDocument.Styles

If Not MonStyle.BuiltIn Then

With ActiveDocument.Content

.Find.ClearFormatting

.Find.Style = ActiveDocument.Styles(MonStyle)

If Not .Find.Execute() Then

MonStyle.Delete: MonCompteur = MonCompteur + 1

End If

End With

End If

Next MonStyle

MsgBox Str(MonCompteur) & "styles personalisés

inutilisés supprimés !"

End Sub

### Apport complémentaire.

L’instruction « MonCompteur = MonCompteur + 1 » permet de fabriquer un compteur comme le nom du registre nous le laisse supposer

# Gestion des cases à cocher dans un formulaire

## But

Installer une macro pour que dans un formulaire avec des cases à cocher, style oui ou non, seule la case nécessaire sera cochée et jamais l'autre.

Effectivement sans Macro on peut cocher les deux, ce qui parfois n’est pas souhaitable.

## Solution

Il faut écrire une « macro de sortie » par champ case à cocher.

Imaginons et réalisons un formulaire équipé de 2 cases à cocher où l’on ne pourrait accepter qu’une seule coche. Soit un formulaire avec deux cases à cocher CaseACocher1 et CaseACocher2 l'une excluant l'autre.

Comme exemple pratique, posons dans un nouveau formulaire la question « Viendrez-vous à la réunion – déjeuner de demain : Oui

Non

Pour que la démonstration soit probante, nous ne pouvons pas le faire dans ce document que nous ne pouvons pas verrouiller. Il faudra donc le faire dans un nouveau document que nous nommerons « Question.doc »

La macro suivante placée en sortie de CaseACocherOui force CaseACocherNon à prendre la valeur inverse

## Installation du code

Sub UneSeuleCase()

With ActiveDocument.FormFields

.Item("CaseACocherNon").CheckBox.Value = Not

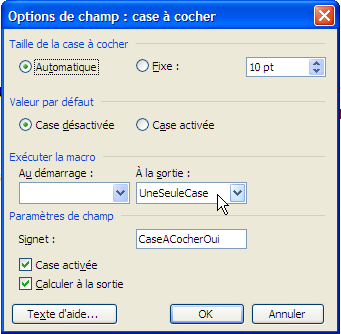
.Item("CaseACocherOui").CheckBox.Value = True

End With

End Sub

L’étape suivante se passe dans l’édition du formulaire.

Il faut déclarer l’exécution de la macro à la **sortie** de chacune des cases à cocher.



Option de Champ

# Créer un nouveau document Excel au départ de Word

Dim objWD as Word.Application

Set objWD = CreateObject("Word.Application")

Dim objWdDoc As Word.Document

Set objWdDoc = GetObject("c:\my documents\doc1.doc")

# Suppression des messages d’avertissement et d’erreur.

## But

Lors de l’exécution d’un code, l’application en cours en l’occurrence Word peut parfaitement s’arrêter pour signaler un avertissement (« Attention, êtes‑vous sûr de vouloir faire cela ? »), ou pour signaler une erreur d’exécution suite à un problème rencontré auquel vous n’aviez pas pensé.

## Complément et rappel

Dans les autres applications l’expression est quelque peu différente

### Access.

Docmd.Setwarnings False

### Excel

Application.DisplayAlerts = False

Sub GestionDesMessagesAlertesEtErreurs()

Application.DisplayAlerts = wdAlertsNone

Application.DisplayAlerts = wdAlertsAll

On Error Resume Next

End Sub

### Conseils

L’instruction qui permet de ne pas être interrompu par un message d’alerte peut-être très pratique, néanmoins il reste conseillé de rétablir les avertissements dès qu’il n’est plus souhaitable de les inhiber.

L’instruction que permet de passer outre de (presque) toutes les erreurs est à placer en tout dernière étape de la réalisation de votre projet, et ce en début de code après la déclaration des registres.

# Remerciements

Merci aux auteurs de ce site : <http://faqword.free.fr/articles.php?lng=fr&pg=61>

# Divers

Ce produit est entièrement gratuit. Si vous l'avez apprécié, soyez assez sympa de verser une contribution que vous estimerez à l'œuvre caritative de votre choix.

1. Je me suis permis cette petite fantaisie de sauvegarde d’un document sur le bureau, bien que ceci soit à proscrire. Le bureau fait partie de votre profil, donc si vous devez travailler sur une machine qui n’est pas votre machine habituelle, vous allez devoir attendre que votre profil entier se charge, ce et y compris les éventuels énormes documents que vous auriez chargés sur votre bureau. [↑](#footnote-ref-1)