



eIGdiplus

Tutoriel : Données Exif

Module VBA de gestion d'image

par Thierry GASPHERMENT (arkham46.developpez.com/)

Date de publication : 24/04/08

Dernière mise à jour : 24/04/08

Gérez les données Exif de vos photos.

I - Introduction.....	3
II - Création du formulaire.....	3
III - Création de la classe.....	4
IV - Déclaration de la classe.....	4
V - Lecture des données EXIF.....	5
VI - Modification des données EXIF.....	8
VII - Sauvegarde du fichier.....	8
VIII - Le code complet.....	9
IX - Conclusion.....	11
X - Téléchargements.....	11

I - Introduction

L'objectif est de proposer un exemple simple de lecture/écriture des données Exif d'une image.


Les données Exif sont des informations stockées dans un fichier Jpeg (ou TIFF).

Elles sont très largement utilisées par les appareils photos numériques pour stocker des informations telles que :

- La date et l'heure du cliché
- Une miniature de l'image pour pré-visualisation rapide
- La vitesse ISO
- Le modèle de l'appareil
- Le fabricant de l'appareil
- Le temps d'exposition
- L'utilisation du flash lors de la prise de la photo

-

Pour lire ces données on utilisera une classe spécifique **cIGdiPlus** qui s'appuie sur la librairie Gdi+ de Microsoft.

 **Gid+** renvoie les valeurs brutes selon le type de données mentionné dans la norme.

Par exemple :

- Le temps d'exposition est de type **RATIO** :
--> On reçoit comme valeur de la propriété un tableau de deux valeurs composant la fraction.
- Le type d'appareil est de type **ASCII** :
--> On reçoit comme valeur de la propriété une chaîne de caractères.



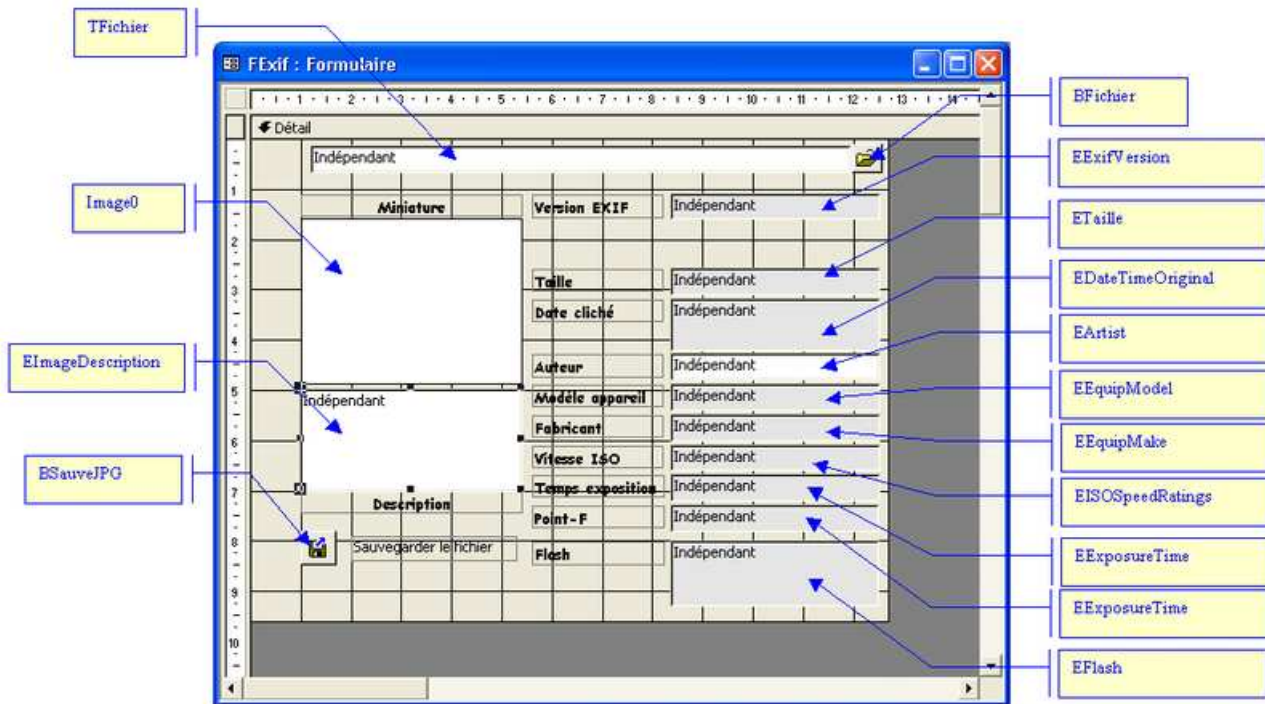
Voici le résultat que l'on obtiendra à la fin du tutoriel

 **Consultez la documentation des fonctions et des propriétés**

II - Création du formulaire

On crée un formulaire indépendant (sans source de données) dans lequel on place :

- Une zone de texte TFichier pour le nom du fichier image
- Un bouton BFichier pour l'affichage de la boîte de dialogue "Ouvrir un fichier"
- Un bouton BSauveJPG pour la sauvegarde de l'image au format Jpeg
- Un contrôle image Image0 pour l'affichage de la miniature
- Autant de zones de texte qu'il est nécessaire pour afficher les tags Exif



Formulaire en mode création

Seuls les champs EArtist et EImageDescription seront modifiables.

On modifie les propriétés des autres zones de texte :

Dans l'onglet "Données":

- Activé : Non
- Verrouillé : Oui

Dans l'onglet "Format":

- Couleur de fond : 15066597

III - Création de la classe

Téléchargez la classe au format texte ([HTTP](http://))

Dans l'éditeur VBA :

- 1 - Créez un nouveau module de classe.
- 2 - Collez-y le contenu du fichier texte.
- 3 - Sauvegardez le module avec le nom **clGdiPlus**.
- 4 - Compiler (**Débogage** --> **Compiler**)

Attention : le nom sous lequel vous sauvegardez le module est important.

IV - Déclaration de la classe

On va écrire notre code dans le module du formulaire. Cliquez sur **Affichage** --> **Code** pour ouvrir ce module.

Vérifiez que vous avez l'instruction **Option Explicit** en haut du module.

Sinon rajoutez cette ligne, elle impose la déclaration de toutes les variables et évite ainsi les erreurs de saisie dans leur nom.

En-tête de module

```
Option Compare Database
Option Explicit
```

Pour pouvoir utiliser la classe il est nécessaire de la **déclarer** et l'**initialiser**.

La classe est un objet dont le type est le nom sous lequel on a sauvegardé notre module de classe.

Elle se déclare comme n'importe quelle autre variable.
 Ensuite on initialise la classe dans l'événement **Sur Ouverture** du formulaire.

Déclaration de la classe

```
Private clGdip As New clGdiPlus
```

Initialisation de la classe

```
Private Sub Form_Open(Cancel As Integer)
    ' Création de la classe à l'ouverture du formulaire
    Set clGdip = New clGdiPlus
End Sub
```

Première chose à faire : pensez à libérer la classe à la fermeture du formulaire.

Dans les propriétés du formulaire, définissez [**Procédure événementielle**] dans l'événement **Sur Fermeture**. Cliquez sur les trois petits points [...] pour générer l'événement dans le code.

A l'intérieur de la procédure **Form_Close** on va libérer la classe : il suffit de lui donner la valeur **Nothing** si elle n'a pas déjà cette valeur.

Libération de la classe

```
Private Sub Form_Close()
    ' Libération de la classe à la fermeture du formulaire
    If Not clGdip is Nothing Then Set clGdip = Nothing
End Sub
```

V - Lecture des données EXIF

Pour afficher une boîte de dialogue de sélection de fichier on utilise une fonction spécifique. Utilisez votre propre fonction si vous en avez déjà une, sinon vous pouvez utiliser la fonction **GetFileName** dont le code est dans la section [Le code Complet](#)

Sélection du fichier via une boîte de dialogue

```
Private Sub BFichier_Click()
    ' Boîte de dialogue
    TFichier = GetFileName(0, "Fichier JPEG", "JPEG", "JPG", ApplicationPath)
    ' Lecture des données Exif après mise à jour du nom de fichier
    TFichier_AfterUpdate
End Sub
```

On va lire les données Exif après mise à jour du nom de fichier dans la zone de texte TFichier.

On va placer notre code de lecture sur l'événement **Après Mise à jour** du contrôle TFichier.

- Ouverture du nouveau fichier

- Lecture des données EXIF

On laisse le fichier ouvert tant qu'on ne change pas de fichier.

On peut ainsi modifier une donnée Exif et sauvegarder l'image en cours.

Il faut par contre bien penser à fermer le fichier quand on ne l'utilise plus.

Remarque : La libération de la classe ferme automatiquement un éventuel fichier ouvert.

Il n'est donc pas nécessaire de fermer le fichier à la fermeture du formulaire.

Idem lors de l'ouverture d'un fichier, le précédent est automatiquement fermé.

On déclare d'abord une variable de type Variant pour stocker la valeur brute des données Exif (dans le cas de données complexes).

Variable pour données brutes

```
' Variable pour donnée brute
Dim lData As Variant
```

On ajoute ensuite une petite gestion d'erreur rapide...

Gestion d'erreurs

```
' Gestion d'erreurs rapide
  On Error Resume Next
```

Puis on ouvre le fichier.

Ouverture du fichier

```
' Ouverture du nouveau fichier
  clGdip.OpenFile TFichier
```

Après l'appel à la fonction **OpenFile**, on peut lire les données Exifs et remplir nos zones de texte. Pour lire une donnée il suffit d'appeler la fonction **GetExifData** en passant en paramètre le code du tag Exif. La liste exhaustive des tags est disponible dans la norme Exif. Les codes des tags les plus courants sont définis dans la classe sous forme de propriété. Par exemple : **TagEquipModel** est le code du tag pour lire le type d'appareil photo. Si un tag n'est pas défini dans l'image la fonction **GetExifData** renvoie **Null**. On va commencer par lire les données simples.

Lecture des données simples

```
' Taille de l'image
  ETaille.Value = clGdip.GetExifData(TagImageWidth) & " x " & clGdip.GetExifData(TagImageHeight)
' Vitesse ISO
  EISOSpeedRatings.Value = Format(clGdip.GetExifData(TagISOSpeedRatings), "\I\S\O 000")
' Modèle appareil
  EEquipModel.Value = clGdip.GetExifData(TagEquipModel)
' Fabricant
  EEquipMake.Value = clGdip.GetExifData(TagEquipMake)
' Version EXIF
  EExifVersion = clGdip.GetExifData(TagExifVersion)
' Description
  EImageDescription = clGdip.GetExifData(TagImageDescription)
' Auteur
  EArtist = clGdip.GetExifData(TagArtist)
```

Ensuite on lit les données plus complexes.

Date et heure du cliché :

Le tag **DateTimeOriginal** renvoie la date et l'heure à laquelle a été prise la photo.

La donnée est un champ de type **Date** auquel on peut appliquer la fonction **Format** pour mettre en forme la date.

Date et heure du cliché

```
' Date du cliché
  EDateTimeOriginal.Value =
  Format(clGdip.GetExifData(TagDateTimeOriginal), "d mmmm yyyy" & vbCrLf & "hh:nn:ss")
```

Miniature :

La miniature intégrée dans le fichier est très pratique.

Elle permet une prévisualisation de l'image sans ouvrir le fichier complet.

On affecte directement la valeur lue par la fonction à la propriété **PictureData** du contrôle image.

Miniature

```
' Miniature
  Me.Image0.Picture = ""
  Me.Image0.PictureData = clGdip.GetExifData(TagThumbnailData)
```

Temps d'exposition :

Le temps d'exposition est un rationnel, c'est à dire une fraction.

On récupère en fait deux valeurs (sous forme d'un tableau) qui correspondent au numérateur et au dénominateur de la fraction.

Par exemple :

Pour un temps d'exposition de 1/60^e de seconde

- 1^{ère} valeur = 10
- 2^{ème} valeur = 600
- Temps d'exposition = $10/600 = 1/60$

Pour un temps d'exposition de 8 secondes

- 1^{ère} valeur = 800
- 2^{ème} valeur = 100
- Temps d'exposition = $800/100 = 8$

Temps d'exposition

```
' Temps exposition
' On stock d'abord le résultat dans lData
lData = clGdip.GetExifData(TagExposureTime)
' On obtient un tableau de 2 valeurs
If Not IsNull(lData) Then
    If lData(1) > lData(0) Then
        ' Temps inférieur à 1 secondes
        EExposureTime.Value = "1/" & Int(lData(1) / lData(0)) & " secondes"
    Else
        ' Temps supérieur ou égal à 1 secondes
        EExposureTime.Value = Int(lData(0) / lData(1)) & " secondes"
    End If
Else
    EExposureTime.Value = Null
End If
```

Point F:

Cette donnée est également un rationnel.

On calcule la valeur à afficher en divisant le numérateur et le dénominateur.

Point F

```
' Point -F
lData = clGdip.GetExifData(TagFNumber)
If Not IsNull(lData) Then
    EFNumer.Value = Format(lData(0) / lData(1), "F0.0")
Else
    EFNumer.Value = Null
End If
```

Le flash :

Le flash n'est pas une donnée facile à lire.

La norme précise que la donnée est stockée sous forme d'un nombre qu'il faut transformer en binaire pour ensuite analyser la valeur de chaque bit.

La classe renvoie directement la valeur binaire dans une chaîne de 8 caractères de long.

le 8^{ème} caractère détermine si le flash s'est déclenché.

les 4^{ème} et 5^{ème} caractères déterminent le mode de flash défini lors du clic.

le 2^{ème} caractère détermine si l'anti-yeux rouges était activé.

(Pour plus d'infos voir la norme Exif).

Flash

```
' Flash
lData = clGdip.GetExifData(TagFlash)
If Not IsNull(lData) Then
    EFlash.Value = IIf(Mid(lData, 8, 1) = "1", "Flash déclenché", "Flash non déclenché")
```

Flash

```

EFlash.Value = EFlash.Value & vbCrLf & Switch(Mid(lData, 4, 2) = "00", "Mode inconnu", _
                                                Mid(lData, 4, 2) = "01", "Flash forcé", _
                                                Mid(lData, 4, 2) = "10", "Flash désactivé", _
                                                Mid(lData, 4, 2) = "11", "Flash auto")
EFlash.Value = EFlash.Value & vbCrLf & Switch(Mid(lData, 2, 1) = "0", "Anti-Yeux rouges désactivé",
                                                Mid(lData, 2, 1) = "1", "Anti-Yeux rouges activé")
Else
    EFlash.Value = Null
End If
    
```

VI - Modification des données EXIF

On va modifier les données Auteur (Artist) et Description (ImageDescription).

Lorsqu'on modifie une donnée Exif, celle-ci n'est pas directement modifiée dans le fichier.

Il faut sauvegarder le fichier pour écrire les modifications.

Les zones **EArtist** et **ElmageDescription** sont modifiables.

On va modifier la donnée Exif correspondante lors de la mise à jour de ces zones.

La fonction à utiliser est **SetExifData**, là encore en passant le code du tag en paramètre.

Il faut également donner à la fonction la nouvelle valeur à écrire

Utilisez la propriété Value de la zone de texte pour bien donner la valeur et non l'objet contrôlé.

Mise à jour des données Exif

```

Private Sub EArtist_AfterUpdate()
    ' Mise à jour de l'auteur
    clGdip.SetExifData TagArtist, EArtist.Value
End Sub

Private Sub EImageDescription_AfterUpdate()
    ' Mise à jour de la description
    clGdip.SetExifData TagImageDescription, EImageDescription.Value
End Sub
    
```

Les modifications ne sont pas apportées sur le fichier directement.

On a avec ce code modifié les données uniquement en mémoire.

VII - Sauvegarde du fichier

On va sauvegarder le fichier avec les données Exif modifiées.

Pour éviter de recompresser l'image lors de la sauvegarde, on utilise la fonction **SaveJpegLossLess**.

L'écrasement du fichier courant n'est pas possible, il faut sauvegarder sous un autre nom de fichier.

Eventuellement vous pouvez ensuite copier le fichier sous son nom initial puis supprimer la copie.

J'utilise là encore la boîte de dialogue de sélection de fichier.

Le code est bien sûr à placer sur l'événement **Sur Click** du Bouton **BSauveJPG**.

Sauvegarde du fichier sans recompression

```

Private Sub BSauveJPG_Click()
    Dim lFichier As String
    On Error GoTo Gestion_Erreurs
    ' Récupère un nom de fichier
    lFichier = GetFileName(0, "Fichier JPEG", "JPEG", "JPG", ApplicationPath)
    If lFichier <> "" Then
        ' Sauvegarde le fichier
        clGdip.SaveJpegLossLess lFichier
    End If
Gestion_Erreurs:
    If Err.Number <> 0 Then MsgBox Err.Description
End Sub
    
```

VIII - Le code complet

Voici le code complet de ce tutoriel :

Code complet

```

Option Compare Database
Option Explicit

' Déclaration de la classe
Dim clGdip As New clGdiPlus

Private Sub BFichier_Click()
    ' Boîte de dialogue
    TFichier = GetFileName(0, "Fichier JPEG", "JPEG", "JPG", ApplicationPath)
    ' Lecture des données Exif après mise à jour du nom de fichier
    TFichier_AfterUpdate
End Sub

Private Sub BSauveJPG_Click()
    Dim lFichier As String
    On Error GoTo Gestion_Erreurs
    ' Récupère un nom de fichier
    lFichier = GetFileName(0, "Fichier JPEG", "JPEG", "JPG", ApplicationPath)
    If lFichier <> "" Then
        ' Sauvegarde le fichier
        clGdip.SaveFile lFichier
    End If
Gestion_Erreurs:
    If Err.Number <> 0 Then MsgBox Err.Description
End Sub

Private Sub EArtist_AfterUpdate()
    ' Mise à jour de l'auteur
    clGdip.SetExifData TagArtist, EArtist.Value
End Sub

Private Sub EImageDescription_AfterUpdate()
    ' Mise à jour de la description
    clGdip.SetExifData TagImageDescription, EImageDescription.Value
End Sub

Private Sub Form_Close()
    If Not clGdip Is Nothing Then Set clGdip = Nothing
End Sub

Private Sub TFichier_AfterUpdate()
    ' Variable pour donnée brute
    Dim lData As Variant
    ' Gestion d'erreurs rapide
    On Error Resume Next
    ' Ouverture du nouveau fichier
    clGdip.OpenFile TFichier
    ' Taille de l'image
    ETaille.Value = clGdip.GetExifData(TagImageWidth) & " x " & clGdip.GetExifData(TagImageHeight)
    ' Vitesse ISO
    EISOSpeedRatings.Value = Format(clGdip.GetExifData(TagISOSpeedRatings), "\I\S\O 000")
    ' Modèle appareil
    EEquipModel.Value = clGdip.GetExifData(TagEquipModel)
    ' Fabricant
    EEquipMake.Value = clGdip.GetExifData(TagEquipMake)
    ' Version EXIF
    EExifVersion = clGdip.GetExifData(TagExifVersion)
    ' Description
    EImageDescription = clGdip.GetExifData(TagImageDescription)
    ' Auteur
    EArtist = clGdip.GetExifData(TagArtist)
    ' Date du cliché

```

Code complet

```

    EDateTimeOriginal.Value =
Format(clGdip.GetExifData(TagDateTimeOriginal), "d mmmm yyyy" & vbCrLf & "hh:nn:ss")
' Miniature
    Me.Image0.Picture = ""
    Me.Image0.PictureData = clGdip.GetExifData(TagThumbnailData)
' Temps exposition
    ' On stock d'abord le résultat dans lData
    lData = clGdip.GetExifData(TagExposureTime)
    ' On obtient un tableau de 2 valeurs
    If Not IsNull(lData) Then
        If lData(1) > lData(0) Then
            ' Temps inférieur à 1 seconde
            EExposureTime.Value = "1/" & Int(lData(1) / lData(0)) & " seconde"
        Else
            ' Temps supérieur ou égal à 1 seconde
            EExposureTime.Value = Int(lData(0) / lData(1)) & " secondes"
        End If
    Else
        EExposureTime.Value = Null
    End If
' Point -F
    lData = clGdip.GetExifData(TagFNumber)
    If Not IsNull(lData) Then
        EFNumer.Value = Format(lData(0) / lData(1), "F0.0")
    Else
        EFNumer.Value = Null
    End If
' Flash
    lData = clGdip.GetExifData(TagFlash)
    If Not IsNull(lData) Then
        EFlash.Value = IIf(Mid(lData, 8, 1) = "1", "Flash déclenché", "Flash non déclenché")
        EFlash.Value = EFlash.Value & vbCrLf & Switch(Mid(lData, 4, 2) = "00", "Mode inconnu", _
            Mid(lData, 4, 2) = "01", "Flash forcé", _
            Mid(lData, 4, 2) = "10", "Flash désactivé", _
            Mid(lData, 4, 2) = "11", "Flash auto")
        EFlash.Value = EFlash.Value & vbCrLf & Switch(Mid(lData, 2, 1) = "0", "Anti-Yeux rouges désactivé",
            Mid(lData, 2, 1) = "1", "Anti-Yeux rouges activé")
    Else
        EFlash.Value = Null
    End If
End Sub

```

Pour l'ouverture de la boîte de dialogue de sélection d'un fichier, coller ce code dans un module.

Boîte dialogue

```

Option Compare Database
Option Explicit

'Déclaration de l'API
Private Declare Function GetOpenFileName Lib "comdlg32.dll" Alias "GetOpenFileNameA" (pOpenfilename As
OPENFILENAME) As Long
'Structure du fichier
Private Type OPENFILENAME
    lStructSize As Long
    hwndOwner As Long
    hInstance As Long
    lpstrFilter As String
    lpstrCustomFilter As String
    nMaxCustFilter As Long
    nFilterIndex As Long
    lpstrFile As String
    nMaxFile As Long
    lpstrFileName As String
    nMaxFileName As Long
    lpstrInitialDir As String
    lpstrTitle As String

```

Boîte dialogue

```

flags As Long
nFileOffset As Integer
nFileExtension As Integer
lpstrDefExt As String
lCustData As Long
lpfnHook As Long
lpTemplateName As String
End Type
Public Function GetFileName(handle As Long, Titre As String, Optional TitreFiltre As String, _
    Optional TypeFichier As String, Optional RepParDefaut As String) As String
Dim StructFile As OPENFILENAME
Dim sFiltre As String
'Construction du filtre en fonction des arguments spécifiés
If Len(TitreFiltre) > 0 And Len(TypeFichier) > 0 Then
    sFiltre = TitreFiltre & " (" & TypeFichier & ")" & Chr$(0) & "." & TypeFichier & Chr$(0)
End If
sFiltre = sFiltre & "Tous (*.*)" & Chr$(0) & ".*" & Chr$(0)
'Configuration de la boîte de dialogue
With StructFile
    .lStructSize = Len(StructFile)
    .hwndOwner = handle
    .lpstrFilter = sFiltre
    .lpstrFile = String$(254, vbNullChar)
    .nMaxFile = 254
    .lpstrFileTitle = String$(254, vbNullChar)
    .nMaxFileTitle = 254
    .lpstrTitle = Titre
    .flags = 0
    .lpstrInitialDir = RepParDefaut
End With
If (GetOpenFileName(StructFile)) Then
    GetFileName = Trim$(Left(StructFile.lpstrFile, InStr(1, StructFile.lpstrFile, vbNullChar) - 1))
End If
End Function

'-----
' Chemin de l'application
'-----
' Utile pour access97 : currentproject.path n'existe pas
'-----
Public Function ApplicationPath() As String
    ApplicationPath = Left(CurrentDb.Name, Len(CurrentDb.Name) - Len(Dir(CurrentDb.Name)))
End Function
    
```

IX - Conclusion

N'hésitez pas à lire la norme pour plus de détails ou si vous avez besoin de lire des tags particuliers. Merci à l'équipe Office de developpez.com pour ses relectures, commentaires et encouragements!

X - Téléchargements

Téléchargez la classe au format texte ([HTTP](#))

Télécharger la base Access de ce tutoriel au format ACCESS 2000 ([HTTP](#))