

C# Windows Forms

DialogResult, MessageBox, openFileDialog, folderBrowserDialog

Sommaire

Historique du document :	2
Liens utiles	2
Initialisation des objets utilisés dans ce document	2
MessageBox	3
MessageBox : Affichage d'un message à l'utilisateur	3
MessageBox : Personnalisation de la boîte de message	4
DialogResult	5
OpenFileDialog	6
OpenFileDialog : Sélection d'un fichier unique	6
OpenFileDialog : Sélection de plusieurs fichiers	7
OpenFileDialog : Filtres de sélection	8
FolderBrowserDialog	9
FolderBrowserDialog : Sélection d'un répertoire	9
Exercice	10
Sélection de fichiers	10
Sélection de répertoire	10
Confirmation de sortie d'application	10

Historique du document :

Date	Auteur	Observations
12/10/2017	Mickaël DEVOLDÈRE	Création du document

Liens utiles

MessageBox MSDN	Documentation officielle pour la classe MessageBox
DialogResult MSDN	Documentation officielle pour l'énumération DialogResult
OpenFileDialog MSDN	Documentation officielle pour la classe OpenFileDialog
FolderBrowserDialog MSDN	Documentation officielle pour la classe FolderBrowserDialog

Initialisation des objets utilisés dans ce document

MessageBox	Méthodes statiques, pas de constructeur
DialogResult	Énumération
OpenFileDialog	OpenFileDialog openFile = new OpenFileDialog() ;
FolderBrowserDialog	FolderBrowserDialog openFolder = new FolderBrowserDialog() ;

MessageBox

La classe `System.Windows.Forms.MessageBox` et plus particulièrement la méthode statique `MessageBox.Show()` permet d'afficher des messages d'informations à l'utilisateur.

MessageBox : Affichage d'un message à l'utilisateur

Pour afficher un simple message à l'utilisateur :

```
MessageBox.Show("Bonjour !");
```



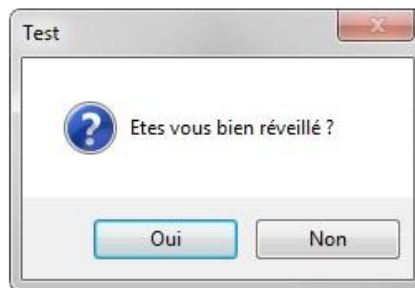
MessageBox : Personnalisation de la boîte de message

Il est possible de personnaliser certaines propriétés de la boîte de message :

- Le titre
- Les boutons utilisables
- L'icône affichée

Le code suivant personnalise la boîte de message avec un titre, les boutons oui/non et l'icône « question »:

```
MessageBox.Show("Etes vous bien réveillé ?", "Test", MessageBoxButtons.YesNo, MessageBoxIcon.Question);
```



DialogResult

L'intérêt de pouvoir personnaliser les boutons réside dans le fait qu'il est possible de savoir sur quel bouton l'utilisateur a cliqué et ainsi réaliser les actions adéquates.

Pour cela, la méthode statique `MessageBox.Show()` renvoie une des valeurs de l'énumération `Dialogresult` (valeur dépendant des boutons choisis via `MessageBoxButtons` [3^{ème} paramètre de la méthode `MessageBox.Show()`]).

Le code suivant démontre comment récupérer cette valeur pour pouvoir l'exploiter.

```
DialogResult resultat = MessageBox.Show("Etes vous bien réveillé ?", "Test", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if(resultat == DialogResult.Yes)
{
    MessageBox.Show("Vous avez cliqué sur Oui");
}
else
{
    MessageBox.Show("Vous avez cliqué sur Non");
}
```

Dans l'exemple suivant, `MessageBox` et `DialogResult` sont utilisés dans une méthode qui gère l'évènement `FormClosing` (évènement déclenché lorsqu'un formulaire est sur le point d'être fermé).

```
/// <summary>
/// Confirmation de l'utilisateur avant de fermer l'application
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    // Affichage de la boîte de message + récupération du bouton cliqué
    DialogResult result = MessageBox.Show(
        "Quitter l'application ?",
        "Confirmation",
        MessageBoxButtons.YesNo,
        MessageBoxIcon.Question
    );

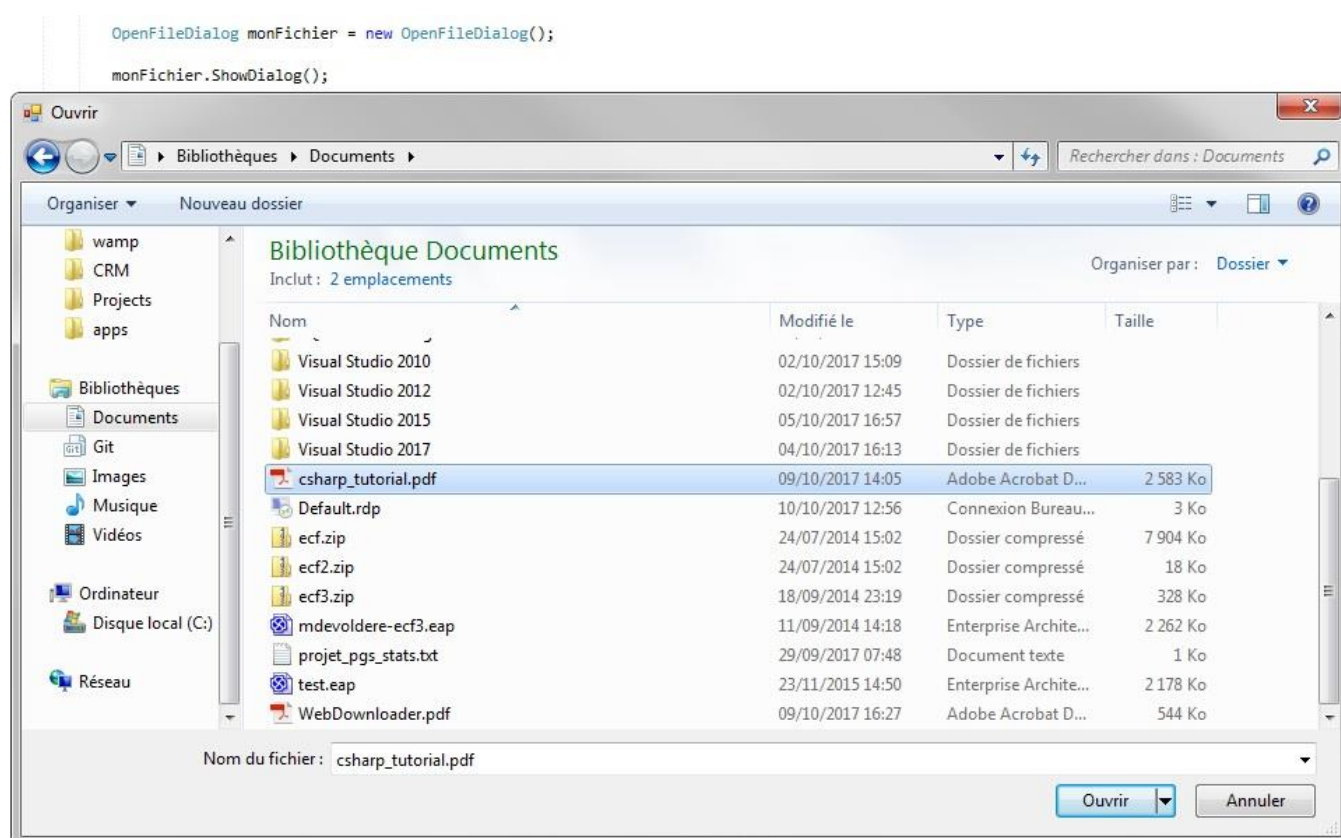
    // Si l'utilisateur a cliqué sur "Non"
    if (result == DialogResult.No)
    {
        e.Cancel = true; // e.Cancel à true annule l'évènement en cours
    }
}
```

OpenFileDialog

La classe `System.Windows.Forms.OpenFileDialog` permet de sélectionner un ou plusieurs fichier(s) stocké(s) sur l'ordinateur.

En appelant la méthode `OpenFileDialog.ShowDialog()`, une fenêtre de sélection de fichiers apparaît.

OpenFileDialog : Sélection d'un fichier unique



Lorsque l'utilisateur a sélectionné un fichier, son chemin est stocké dans la propriété `FileName` de l'instance courante de la classe `OpenFileDialog`.

```
OpenFileDialog monFichier = new OpenFileDialog();
DialogResult resultat = monFichier.ShowDialog();
string cheminDuFichier = monFichier.FileName;
```

Note : Si l'utilisateur clique sur le bouton « Annuler » ou ferme la fenêtre en utilisant la croix dans le coin supérieur droit, la propriété `FileName` sera vide.

L'exemple suivant démontre, au travers d'un évènement Click, comment récupérer le chemin du fichier sélectionné par l'utilisateur uniquement si celui a validé sa sélection en cliquant sur le bouton « ouvrir ».

```
/// <summary>
/// Sélectionner un fichier sur le disque
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btn_file_Click(object sender, EventArgs e)
{
    if (openFile.ShowDialog() == DialogResult.OK) // si l'utilisateur a cliqué sur le bouton OK (estampillé "Ouvrir")
    {
        FileInfo fInfo = new FileInfo(openFile.FileName); // récupération des informations du fichier

        txt_fileinfo.Text = fInfo.Attributes.ToString(); // Affichage des attributs du fichier

        txt_file.Text = openFile.FileName; // Affichage du chemin du fichier
    }
}
```

OpenFileDialog : Sélection de plusieurs fichiers

Il est également possible pour l'utilisateur de sélectionner plusieurs fichiers à partir de la même fenêtre de sélection en définissant la propriété *MultiSelect* à *true*. Dans ce cas, la liste des fichiers sélectionnés par l'utilisateur sera stockée dans la propriété *FileNames*.

L'exemple suivant démontre la récupération de la liste des fichiers sélectionnés par l'utilisateur.

```
/// <summary>
/// Sélectionner un ou plusieurs fichiers sur le disque
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btn_file_Multiple_Click(object sender, EventArgs e)
{
    openFile.Multiselect = true;

    if (openFile.ShowDialog() == DialogResult.OK) // si l'utilisateur a cliqué sur le bouton OK (estampillé "Ouvrir")
    {
        foreach (string unFichier in openFile.FileNames) // parcours de la liste des fichiers sélectionnés
        {
            MessageBox.Show(unFichier); // affichage du fichier courant
        }
    }
}
```


OpenFileDialog : Filtres de sélection

La propriété *Filter* de la classe OpenFileDialog permet de limiter les types de fichiers que l'on souhaite pouvoir ouvrir.

`openFileDialog.Filter = "Affichage filtre|extensions " + "|affichage autrefiltre|extensions"`

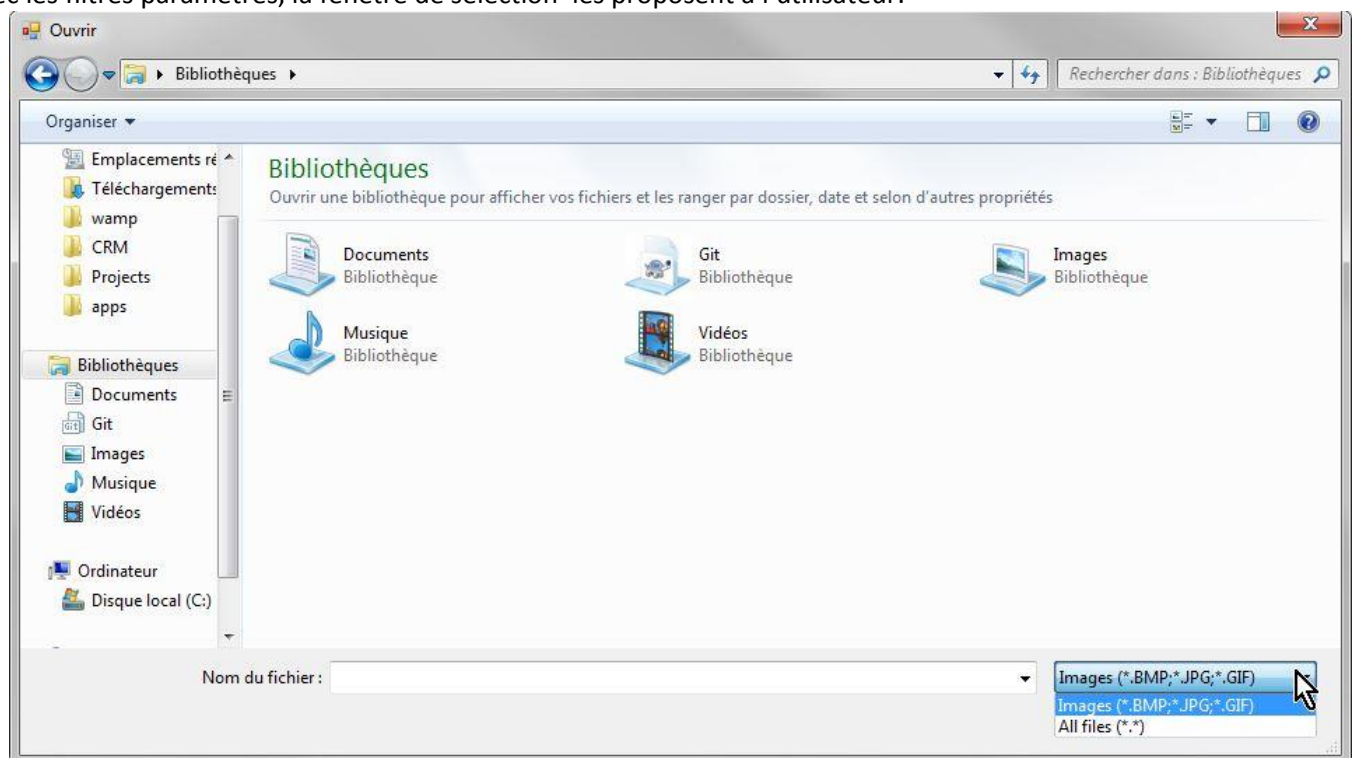
L'exemple suivant démontre comment créer des filtres de types de fichiers :

```
/// <summary>
/// Sélectionner un ou plusieurs fichiers sur le disque
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btn_file_Multiple_Click(object sender, EventArgs e)
{
    openFile.Multiselect = true;

    openFile.Filter = "Images (*.BMP;*.JPG;*.GIF)|*.BMP;*.JPG;*.GIF|" + "All files (*.*)|*.*";

    if (openFile.ShowDialog() == DialogResult.OK) // si l'utilisateur a cliqué sur le bouton OK (estampillé "Ouvrir")
    {
        foreach (string unFichier in openFile.FileNames) // parcours de la liste des fichiers sélectionnés
        {
            MessageBox.Show(unFichier); // affichage du fichier courant
        }
    }
}
```

Avec les filtres paramétrés, la fenêtre de sélection les proposent à l'utilisateur:



FolderBrowserDialog

FolderBrowserDialog : Sélection d'un répertoire

La classe `System.Windows.Forms.FolderBrowserDialog` fonctionne « *globalement* » de la même manière que la classe `System.Windows.Forms.OpenFileDialog` à la différence que l'on va sélectionner un répertoire au lieu d'un fichier.

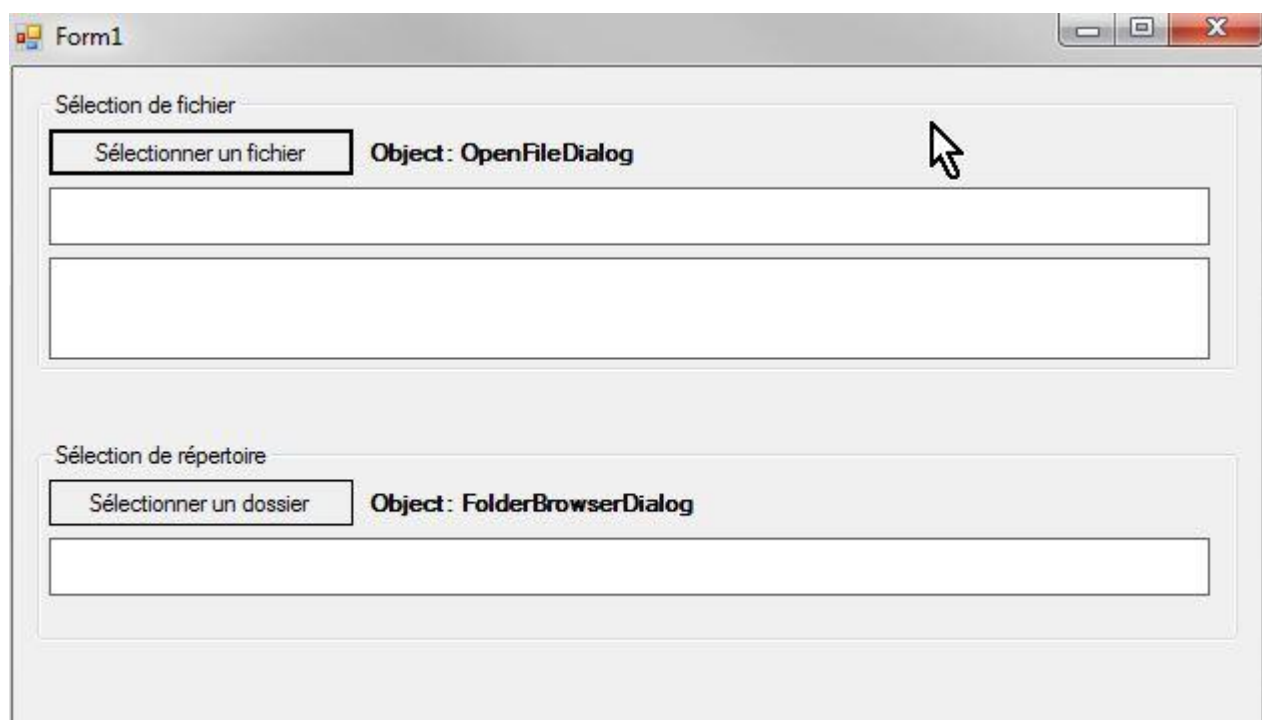
L'exemple suivant démontre, au travers d'un évènement Click, comment récupérer le chemin du répertoire sélectionné par l'utilisateur uniquement si celui-ci a validé sa sélection par le bouton OK.

```
/// <summary>
/// Sélectionner un répertoire sur le disque
/// </summary>
/// <param name="sender"></param>
/// <param name="e"></param>
private void btn_folder_Click(object sender, EventArgs e)
{
    if (openFolder.ShowDialog() == DialogResult.OK) // si l'utilisateur a cliqué sur le bouton OK
    {
        txt_folder.Text = openFolder.SelectedPath; // Affichage du chemin dans un TextBox
    }
}
```

Lorsque l'utilisateur a sélectionné un répertoire, son chemin est stocké dans la propriété `SelectedPath` de l'instance courante de la classe `FolderBrowserDialog`.

Exercice

En vous aidant de ce document, réalisez une application de sélection de fichiers/répertoires ressemblant à la capture d'écran suivante :



Sélection de fichiers

Le bouton ouvre la boîte de dialogue de sélection d'un fichier unique. Le chemin du fichier sélectionné sera affiché dans le 1^{er} textbox.

Le 2nd textbox affichera les attributs du fichier sélectionné.

Sélection de répertoire

Le bouton ouvre la boîte de dialogue de sélection de répertoire. Le chemin du répertoire sélectionné sera affiché dans le TextBox placé juste en dessous.

Confirmation de sortie d'application

Implémenter une boîte de dialogue de confirmation lorsque l'utilisateur souhaite fermer l'application.