

Date: Vendredi 28 septembre 2012 à 22:30:09
Sujet: Programmation C# .NET

C# Copier un objet sans implémenter ICloneable

Pour cloner un objet en C#, il est recommandé d'implémenter l'interface `ICloneable`. On peut ensuite appeler la méthode `Clone()` sur l'objet implementant la classe `ICloneable`.

Cependant il n'est pas toujours possible d'implémenter l'interface `ICloneable` si la classe de l'objet à copier ne nous appartient pas. La plupart des bibliothèques externes proposent des méthodes `Clone` quand cela est nécessaire mais ce n'est pas toujours le cas.

Lorsque vous n'avez pas accès à la classe de l'objet à copier, il reste une solution, copier l'objet en utilisant la serialization binaire.

Ce type de clonage fait une copie bit par bit de l'objet cible. Les références pointeront donc vers les mêmes objets que l'objet initial et seuls les types valeurs seront également clonés.

Ce type de copie ne peut donc pas être utilisée dans tous les cas et il faut bien comprendre les impacts avant de l'utiliser.

Voici le code d'un serializer permettant d'effectuer une copie binaire d'un objet même s'il ne possède pas de fonction `Clone`.

Pour l'utiliser, il vous suffit d'appeler `ObjectCloner.Clone(objetACopier)`.

Exemple:

```
MyClass copie = ObjectCloner.Clone(original);
```

```
using System.IO;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;

public static class ObjectCloner
{
    /// <summary>
    /// Perform a deep Copy of the object.
    /// </summary>
    /// <param name="T">The type of object being copied.</param>
    /// <param name="source">The object instance to copy.</param>
    /// <returns>The copied object.</returns>
    public static T Clone<T>(T source)
    {
        if (!typeof(T).IsSerializable)
        {
```

```
      &nbsp; throw new ArgumentException("The type must be serializable.", "source");  
       }  
  
      &nbsp; // Don't serialize a null object, simply return  
the default for that object  
&nbsp; &nbsp; &nbsp; if (Object.ReferenceEquals(source, null))  
&nbsp; &nbsp; &nbsp; {  
&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; return default(T);  
&nbsp; &nbsp; &nbsp; }  
  
&nbsp; &nbsp; &nbsp; IFormatter formatter = new BinaryFormatter();  
&nbsp; &nbsp; &nbsp; Stream stream = new MemoryStream();  
&nbsp; &nbsp; &nbsp; using (stream)  
&nbsp; &nbsp; &nbsp; {  
&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; formatter.Serialize(stream,  
source);  
&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; stream.Seek(0, SeekOrigin.  
Begin);  
&nbsp; &nbsp; &nbsp; &nbsp; &nbsp; &nbsp; return (T)formatter.Deserialize  
(stream);  
&nbsp; &nbsp; &nbsp; }  
&nbsp; &nbsp; }  

```

Publication de Tout sur l'informatique - Programmation C#,
Sécurité, Divx, P2P:
<http://www.zmaster.fr>

URL de cette publication

<http://www.zmaster.fr/modules.php?name=News&file=article&sid=243>