TP et Cours n°3 : .net – Les concepts importants

SERVICES RÉSEAUX

Hélène CHASSAGNE
helene.chassagne@orange.fr

Frédéric CHASSAGNE
frederic.chassagne@atosorigin.com

Plan du cours

- Système de fichiers
- Sérialisation
- Cryptographie

Système de fichiers

- API du gestion des fichiers
  - System.IO
- But : Stockage physique des données
- Endroits de stockage
  - Disque dur
  - Registre
  - IsolatedStorage

Système de fichiers

- Sauvegarde sur le disque dur
  - Api utilisée : System.IO
  - Classes utiles à la manipulation de fichiers
    - Directory
    - File
    - DirectoryInfo
    - FileInfo
    - Path

Système de fichiers

- Sauvegarde sur le disque dur
  - Manipulation de flux
    - FileStream()
    - StreamReader()
    - StreamWriter()
    - MemoryStream()
    - …

Système de fichiers

- Le registre
  - Composé de 4 branches
  - Branches notables
    - CurrentUser
    - LocalMachine
- API utilisée :
  - Microsoft.Win32
- Classe : RegistryKey

Système de fichiers

Exemple

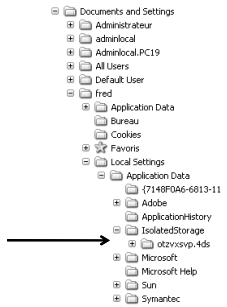
```
public static string GetClientKey(string key)
    {
       RegistryKey regKey;
       string value = "";

       if (true) regKey = Registry.LocalMachine.OpenSubKey(@"Software\DELTA MU
Conseil\OPTI MU Client", false);
       else regKey = Registry.CurrentUser.OpenSubKey(@"Software\DELTA MU Conseil\OPTI
MU Client", false);

       if (regKey != null) value = (string)regKey.GetValue(key);
        return value;
    }
```


Système de fichiers

- System.IO.IsolatedStorage
  - IsolatedStorageFile
  - IsolatedStorageFileStream



Système de fichiers

- Exemple

```
IsolatedStorageFile storageFile = null; IsolatedStorageFileStream storageFileStream = null;

storageFile = IsolatedStorageFile.GetStore(IsolatedStorageScope.Roaming |
    IsolatedStorageScope.User |    IsolatedStorageScope.Assembly |
    IsolatedStorageScope.Domain, null, null);

    foreach (string filePath in
    storageFile.GetFileNames(System.IO.Path.Combine(MY_BITMAP_FOLDER,MY_BITMAP_E
    XT)))
    {
      storageFileStream = new IsolatedStorageFileStream(filePath, System.IO.FileMode.Open,
            System.IO.FileAccess.Read, System.IO.FileShare.ReadWrite, storageFile);

... Traitement ...
    }
```

9

Système de fichiers

- Avantages de l'IsolatedStorage
  - Full accès à l'espace disque
  - Transparent pour l'utilisateur
  - Aucun path à renseigner
- Inconvénients
  - Nécessite une gestion manuelle de l'effacement
  - Ou une gestion via installation msi

10

Plan du cours

- Système de fichiers
- →Sérialisation
- Cryptographie

11

Sérialisation

- Sérialisation = Conversion d'un objet sous une forme transportable
- Désérialisation = Conversion d'un flux de données en objet
- 3 grands types de sérialisation
  - Binaire
  - SOAP
  - XML

12

Sérialisation

- Avantages sérialisation binaire
  - Complète
  - Orientée métier (format non lisible)
- Avantages sérialisation SOAP
  - Complète
  - Orientée données (format lisible)
- Avantages sérialisation XML
  - Restrictive
  - Orientée Configuration (format lisible)

13

Sérialisation

- Comparaisons

	XML	SOAP	Binaire
"Human readable"	oui	oui	non
Sérialisation de types non standards	non	oui	oui
Sérialisation des éléments privés	non	oui	oui
Sérialisation des champs	non	oui	oui
Sérialisation des propriétés	oui	non	non

14

Sérialisation

- Principe
  - Marquer la classe comme [Serializable] (convention en XML)
  - Gérer les accesseurs
    - Implémenter une méthode de Load()
    - Implémenter une méthode de Save()

15

Sérialisation

- Exemple

```
[SerializableAttribute]
  public class Homme
  {
    public string Nom;
    public string Prenom;

    public Homme(string prenom, string
nom)
    {
      Prenom = prenom;
      Nom = nom;
    }
  }
```

```
[SerializableAttribute]
  public class Hommes :
System.Collections.CollectionBase
  {
    public void Add(Homme HommeA)
    {
      this.List.Add(HommeA);
    }

    public Homme this[int Index]
    {
      get
      { return (Homme)this.List[Index]; }
    }
  }
```

16

Sérialisation

- Sérialisation binaire
  - using System.Runtime.Serialization.Formatters.Binary;
- Accesseurs

Sauvegarde

```
FileStream mFile = new
FileStream(@"c:\easyBin.net",
FileMode.Create);

BinaryFormatter mS = new BinaryFormatter();
mS.Serialize(mFile, _lesHommes);
mFile.Close();
```

Chargement

```
FileStream mFile = new
FileStream(@"c:\easyBin.net", FileMode.Open);
      BinaryFormatter mS = new
BinaryFormatter();
      _lesHommes =
(Hommes)mS.Deserialize(mFile);
      mFile.Close();
```

17

Sérialisation

- Sérialisation SOAP
  - using System.Runtime.Serialization.Formatters.Soap;
- Accesseurs

Sauvegarde

```
FileStream mFile = new
FileStream(@"c:\easySoap.net",
FileMode.Create);

SoapFormatter mS = new SoapFormatter ();
mS.Serialize(mFile, _lesHommes);
mFile.Close();
```

Chargement

```
FileStream mFile = new
FileStream(@"c:\easySoap.net",
FileMode.Open);
      SoapFormatter mS = new SoapFormatter
();
      _lesHommes =
(Hommes)mS.Deserialize(mFile);
      mFile.Close();
```

18

Sérialisation

- Sérialisation XML
  - using System.Xml.Serialization;
- Accesseurs

Sauvegarde

```
StreamWriter stream = new
StreamWriter(@"c:\easyXML.net");

XmlSerializer serializer = new
XmlSerializer(typeof(Hommes));
serializer.Serialize(stream, _lesHommes);
stream.Close();
```

Chargement

```
XmlSerializer deserializer = new
XmlSerializer(typeof(Hommes));
StreamReader stream = new
StreamReader(@"c:\easyXML.net");

_lesHommes =
(Hommes)deserializer.Deserialize(stream);
stream.Close();
```

19

Sérialisation

- Sérialisation XML
  - Marquer la classe comme [Serializable] (convention)
  - Exclure les attributs non souhaités avec [XmlIgnoreAttribute]
- Restrictions
  - Nécessite un constructeur par défaut.
  - Sérialise seulement les propriétés/champs publiques non « readonly »

20

Plan du cours

- Système de fichiers
- Sérialisation
- →Cryptographie

Cryptographie

- 2 fonctionnalités
  - Hashage des données
  - Cryptage des données
    - Symétrique
    - Asymétrique
- But du hashage = Masquer les données
- But du cryptage = Sécuriser les données

Cryptographie

- Algorithmes de Hashage
  - SHA 1, 256, 384, 512
  - MD5
- Classe de base : HashAlgorithm

- 1 bon hash = 1 hash salé
- Salage = Additionner 1 paramètre unique à la donnée a hasher pour rendre le hash unique

Cryptographie

- API utilisée pour le hashage
  - using System.Security.Cryptography;
- Exemple

```
private string TestHash(string TextToHash, HashAlgorithm hash)
{
  byte[] bytValue;
  byte[] bytHash;
 bytValue = System.Text.Encoding.UTF8.GetBytes(TextToHash);
  bytHash = hash.ComputeHash(bytValue);
  hash.Clear();
  return (Convert.ToBase64String(bytHash));
}
```

Cryptographie

- Appel depuis le code

```
TestHash("MaChaineaHasher", new SHA1CryptoServiceProvider());
```

- Intérêt
  - Indépendance entre le choix de l'algorithme et l'implémentation
  - Découplage entre Couche Métier et Présentation

25

Cryptographie

- Algorithmes de cryptage symétrique
  - D.E.S
  - Triple D.E.S
  - Classe de base
    - SymmetricAlgorithm
- Algorithmes de cryptage asymétrique
  - RSA
  - Classe de base
    - AsymmetricAlgorithm

26

Cryptographie

- API utilisée pour le cryptage
  - using System.Security.Cryptography;
- Exemple - Initialisation

```
TripleDESCryptoServiceProvider d = new TripleDESCryptoServiceProvider();
d.IV = d.GenerateIV ();
d.Key = d.GenerateKey();
```

- Stockage : System.Security.SecureString

27

Cryptographie

- Exemple – cryptage

```
public string Encrypt(string original, SymmetricAlgorithm sa)
{
   ICryptoTransform ct; MemoryStream ms; CryptoStream cs;   byte[] byt;

   ct = sa.CreateEncryptor(sa.Key, sa.IV);
   byt = Encoding.UTF8.GetBytes(original);

   ms = new MemoryStream();
   cs = new CryptoStream(ms, ct, CryptoStreamMode.Write);
   cs.Write(byt, 0, byt.Length);
   cs.FlushFinalBlock();
   cs.Close();

  return  Convert.ToBase64String(ms.ToArray());
}
```

28

Cryptographie

- Exemple - décryptage

```
public string Decrypt(string crypte, SymmetricAlgorithm sa)
{
   ICryptoTransform ct; MemoryStream ms; CryptoStream cs; byte[] byt;

   ct = sa.CreateDecryptor(sa.Key, sa.IV);
   byt = Convert.FromBase64String(crypte);

   ms = new MemoryStream();
   cs = new CryptoStream(ms, ct, CryptoStreamMode.Write);
   cs.Write(byt, 0, byt.Length);
   cs.FlushFinalBlock();
   cs.Close();

   return Encoding.UTF8.GetString(ms.ToArray());
}
```

29

Cryptographie

- Appel depuis le code

```
TripleDESCryptoServiceProvider d = new TripleDESCryptoServiceProvider();
d.GenerateIV();
d.GenerateKey();
string res =  Decrypt(Encrypt("toto", d), d);
```

- Intérêt
  - Indépendance entre le choix de l'algorithme et l'implémentation
  - Découplage entre Couche Métier et Présentation

30