TD Objets distribués : un service de gestion de scores en .NET Remoting avec persistance des données

---

Ce TD devra aboutir sur une extension de votre réalisation 'Jeu'. Le produit final devra être rendu en fin de séance par mail.

On se propose d'écrire une application distribuée en .NET Remoting réalisant un service de gestion de scores. Vous avez déjà implémenté ce service (TopTen de votre application 'Jeu') sans se soucier de la persistance des données. Pour des raisons de simplicité, nous ne traiterons pas les aspects 'authentification' et 'suivi de connexion'.
Votre travail se décomposera en deux parties :

- Réalisation du service
- Intégration du service dans votre application 'Jeu'.

Le service permettra de gérer des scores (AjouterScore, SuppressionScore, etc.) et de les rendre accessibles (ConsulterScore). On vous demande de réaliser le client et le serveur. Ce document vous donne quelques morceaux de code avec des fonctionnalités restreintes (gestion minimale des exceptions, etc.)

1. La couche physique des données (CPD)

1.1. ACCESS

En exploitant votre SGBD ACCESS, créer une base de données LesScores avec une seule table Score(Numero,Nom,Prénom,Score).
Alimenter votre table Score de quelques lignes.

1.2. MySQL

1.3. SQL-SERVER

2. La couche d'accès aux données (CAD)

Cette couche devra permettre d'accéder aux données persistantes sans connaître la technologie de stockage des données (fichiers, base de données, etc.).
Fichier ScoresCAD.cs :

```
using System;
using System.Data;
using System.Data.OleDb;
namespace TD_Scores
{
    public class ScoresCAD
    {
        public ScoresCAD() { }
        void AjouterScoreSQL(string n, string p, int v)
                              //n:Nom, p:Prenom, v:Valeur
```

```
        {
            System.Data.OleDb.OleDbConnection Connection;
            Connection = new System.Data.OleDb.OleDbConnection();
            Connection.ConnectionString =
"Provider=microsoft.jet.oledb.4.0;Data Source=../../../Scores.mdb;";
            Connection.Open();
            // Requête d'insertion et son exécution
            //A FAIRE…
            // Fermeture de la connexion
            Connection.Close();
        }
        DataSet ExtraireScoreSQL(string nom)
        {
           System.Data.OleDb.OleDbConnection Connection;
            Connection = new System.Data.OleDb.OleDbConnection();
            Connection.ConnectionString =
"Provider=microsoft.jet.oledb.4.0;Data Source=../../../Scores.mdb;";
            Connection.Open();
            String Requete = "SELECT Nom,Prenom,Valeur
                              FROM Score WHERE Nom = '" + nom + "'";
            OleDbDataAdapter da = new OleDbDataAdapter(Requete,Connection);
            // Préparation du DataSet ds
            DataSet ds = new DataSet();
            da.Fill(ds, "ListeScores");
            // Fermeture de la connexion
            Connection.Close();
            return ds;
        }
        /** Recherche d'un score dans la BD **/
        public Score ExtraireScore(string nomRecherche)
        {
            Score score = new Score();
            System.Data.DataSet requete = ExtraireScoreSQL(nomRecherche);
            score.Nom =
                   (string)requete.Tables["ListeScores"].Rows[0]["Nom"];
            score.Valeur =
                   (int)requete.Tables["ListeScores"].Rows[0]["Valeur"];
            try
            {
                score.Prenom =
                   (string)requete.Tables["ListeScores"].Rows[0]["Prenom"];
            }
            catch (Exception e)
            {
                score.Prenom = "";
            }
            return score;
        }

        public void AjouterScore(Score score)
        {
            AjouterScoreSQL(score.Nom, score.Prenom, score.Valeur);
        }
    }
}
```

3. La couche métier

On doit spécifier à ce niveau les services, ce qui se traduit naturellement par la spécification des interfaces du service.

Fichier IScoreService.cs :

```
using System;
namespace TD_Scores
{

    public interface IScoresService
    {
        Score Rechercher(string nom);
        void Ajouter(Score score);
    }
}
```

L'implémentation de ces services utilisera la couche d'accès aux données (CAD).

Fichier ScoresService.cs :

```
using System;
namespace TD_Scores
{

    public class ScoresAdminService : IScoresService
    {
        public ScoresAdminService() { }
        public Score Rechercher(string nom)
        {
            ScoresCAD cad = new ScoresCAD();
            Score S = cad.ExtraireScore(nom);
            return S;
        }
        public void Ajouter(Score score)
        {
            ScoresCAD cad = new ScoresCAD();
            Score Doublon = cad.ExtraireScore(score.Nom);
            if (Doublon == null)
                cad.AjouterScore(score);
            else
                Console.WriteLine(" Ce nom a déjà un score : il faut le
remplacer !");// AFAIRE !
        }
    }
}
```

4. L'objet Score

Fichier Score.cs :

```
using System;

namespace TD_Scores
{
    [Serializable]
    public class Score
    {
        public string Nom;
        public string Prenom;
        public int Valeur;
    }
}
```

5. Accès des services via .Net Remoting

De manière similaire aux précédentes séances, nous distinguerons l'objet publié (ScoreServiceRemoting.cs) du code qui publiera cet objet (Serveur.cs)

ScoreServiceRemoting.cs :

```
using System;
using System.Runtime.Remoting;
using System.Runtime.Remoting.Channels;
using System.Runtime.Remoting.Channels.Tcp;
using TD_Scores;

namespace MesScoresRemoting
{
    public class ScoresServiceRemoting : MarshalByRefObject,
    IScoresService
    {
        IScoresService les_scores = new ScoresAdminService();
        public Score Rechercher(string nom)
        {
            Score S = les_scores.Rechercher(nom);
            return S;
        }
        public void Ajouter(Score score)
        {
            les_scores.Ajouter(score);
        }
    }
}
```

Serveur.cs :

```
using System ;
using System.Runtime.Remoting ;
using System.Runtime.Remoting.Channels ;
using System.Runtime.Remoting.Channels.Tcp;
using MesScoresRemoting ;
public class Server {
[STAThread]
static void Main ( string [] args )
    {
        IChannel chan = new TcpChannel (5555);
        ChannelServices.RegisterChannel(chan,true);
        RemotingConfiguration.RegisterWellKnownServiceType (typeof
(ScoresServiceRemoting),
                            "TcpService",
                            WellKnownObjectMode.Singleton );

        Console.WriteLine ("Appuyer une touche pour arrêter le service");
        Console.ReadLine ();
   }
}
```

6. Client .Net Remoting

Similaire aux précédentes séances.

Client.cs :

```
using TD_Scores ;
using System;
using System.Runtime.Remoting;
using System.Runtime.Remoting.Channels;
using System.Runtime.Remoting.Channels.Tcp;

class ClientRemoting
{
static void Main ( string [] args ) {
```

```
TcpChannel channel = new TcpChannel ();
ChannelServices . RegisterChannel ( channel,true );
IScoresService les_scores
=(IScoresService)Activator.GetObject(typeof(IScoresService),"tcp://127.0.0.
1:5555/TcpService");
Score S = les_scores.Rechercher("HARRISON");
Console.WriteLine("Le nom de HARRISON est "+S.Nom);
Console.WriteLine("Le prénom de HARRISON est "+S.Prenom);
Console.WriteLine("Le score de HARRISON est "+S.Valeur.ToString());

Console.Read();
}
}
```

7. Intégration du service dans votre application ‘Jeu’

Dans votre application ‘Jeu’, remplacer votre service TopTen sans persistance des données par ce service LesScores.

8. Extensions

8.1. Utiliser un autre SGBD pour le stockage des données

MySQL par exemple.

8.2. Utiliser un service web pour accéder au service

8.3. Utiliser un page ASP.NET pour accéder aux données

8.4. Utiliser localement l’objet Scores