**Лекція 12 Технологія Remoting. Об’єкт що активізується клієнтом**

Типова архітектура триланкової системи з окремо створюваним Remoting-об'єктом у вигляді бібліотеки, Remoting сервером, що надає об'єкт клієнтам, Remoting клієнтом, які отримують об’єкт і сервером бд

Сервер БД

Реализация запросов

1.Создание канала

2.Регистрация канала

3.Регистрация объекта

Remoting-сервер

Remoting-клиент

1. Регистрация объекта
2. Создание объекта
3. Вызов методов

Remoting-объект

С методами и реализацией

# Создание объекта

Об'єкт можна створити як звичайну dll -бібліотеку класів (Class Library), в якій створюється клас, що містить реалізацію методів призначених для виклику клієнтами. Ім'я класу - ім'я remoting - об'єкта  
Залежно від способу надання об'єкта клієнтам є два варіанти оголошення класу:  
1 варіант: Передача об'єкта за посиланням  
public class RemOb1: System.MarshalByRefObject  
     {  
         public RemOb1 ()  
         {  
           // MessageBox.Show ("Constructor RemOb1 called");  
             
         }  
          
         public DataTable GetADOTable (string name)  
         {String connstr = "Provider = Microsoft.Jet.OLEDB.4.0; Data Source = D: \\ ASOER\_CB \\ ASOER\_DB \\ ASOER\_ML.mdb";  
              DataTable restab;  
              OleDbCommand Ocmd; OleDbConnection Conn;  
           // DataSet XMLDataSet = new DataSet (); // DataRow dr;  
              Conn = new OleDbConnection (connstr); Ocmd = Conn.CreateCommand ();  
              Ocmd.CommandText = name; Ocmd.CommandType = CommandType.Text;  
              Conn.Open ();  
              OleDbDataAdapter da = new OleDbDataAdapter (Ocmd.CommandText, Conn);  
              restab = new DataTable ();  
              da.Fill (restab);  
            // MessageBox.Show ("Metod GetADOTable called");  
             return (restab);  
}  
}  
  
2варіант: Передача об'єкта за значенням (шляхом серіалізациі)  
[Serializable]  
public class RemOb1  
     {  
         public RemOb1 ()  
         {// MessageBox.Show ("Constructor RemOb1 called"); }  
          
         public DataTable GetADOTable (string sqlname)  
          {// Аналогічна реалізація методу}  
    }  
   
**Серверний додаток**Об'єкти, що активізуються клієнтом (Client-activated objects, CAO)  
Об'єкти, що активуються клієнтом, - це об'єкти, це об'єкти на серверній стороні, керуючі по запиту клієнта час існування яких не управляється сервером. Спосіб активізації цих серверних об'єктів багато в чому аналогічний активізації соклассов у класичній моделі COM. Коли клієнт за допомогою оператора new відправляє запит до серверного об'єкту, віддаленого додатком надсилається повідомлення із запитом на активізацію. Потім сервер створює екземпляр запитуваного класу і повертає ObjRef зухвалому клієнтського додатку. Далі на клієнтській стороні за отриманим ObjRef створюється проксі, через який клієнт буде викликати методи віддаленого об'єкта. Об'єкти, що активізуються клієнтом, можуть зберігати інформацію про стан між викликами методів для конкретного клієнта, але не для різних клієнтських об'єктів. Кожен виклик new повертає проксі для незалежного примірника типу, існуючого на сервері.  
На сервері спочатку необхідно створити і реєструвати канал доступу до об'єкта, наприклад  
// Створюємо Tcp або Http канал channel  
     channel = new TcpChannel (888); // Варіант HttpChannel (888)  
   // Реєструємо канал channel;  
             ChannelServices.RegisterChannel (channel, false);  
   
  Потім, для створення об'єкта активируемого сервером його необхідно зареєструвати, а саме:  
// Реєструємо з боку сервера RemOb1 об'єкт як об'єкт типу Activated (керований клієнтом, без URI)  
RemotingConfiguration.RegisterActivatedServiceType (typeof (RemOb1));  
   
Повний опис методу  
Метод RemotingConfiguration.RegisterActivatedServiceType (Type)  
Реєструє вказаний тип об'єктів, зареєстрованих з боку служби, в якості типу, який може бути активований по запиту клієнта.  
список перевантаження

|  |  |
| --- | --- |
| ім'я | Опис |
| RegisterActivatedServiceType (ActivatedServiceTypeEntry) | Реєструє тип об'єкта наданий параметром ActivatedServiceTypeEntry в якості типу, який може активуватися по запиту клієнта. |
| RegisterActivatedServiceType (Type) | Реєструє вказаний тип об'єктів, зареєстрованих з боку служби, в якості типу, який може бути активований по запиту клієнта. |

Параметри type Тип: System.Type тип об'єкта для реєстрації.  
Винятки: SecurityException Принаймні в одного з викликають операторів, що знаходяться у верхній частині стека викликів, відсутній дозвіл на налаштування типів і каналів віддаленого взаємодії.  
    Використання конфігураційного файлу дозволяє адміністратору налаштовувати поведінку програми без необхідності перекомпілювати його. Але найголовніше, можна замінити весь попередній код одним рядком:  
  
{  
RemotingConfiguration.Configuref @ "... \ JobServer.exe.config");  
  }  
  
     ПРИМІТКА За угодою ім'я конфігураційного файлу утворюється шляхом додавання до імені виконуваного файлу програми рядки «.config».  
Нижче показаний конфігураційний файл JobServer.exe.config:  
  
<configuration>  
      <system.runtime.remoting>  
           Application name = "JobServer">  
                <service>  
                       type = "JobServerLib. JobServerlmpl, JobServerLib"  
                </ service>  
                <channels>  
                     <channel ref = "http"  
                         port = "4000" />  
                </ channels>  
           </ application>  
      </system.runtime.remoting>  
</ configuration>  
  
Зверніть увагу на те, як співвідноситься вміст конфігураційного файлу і код віддаленого взаємодії, доданий раніше. Елемент <channel /> містить ту ж саму інформацію для налаштування каналу, яка була використана в первинному варіанті програмної конфігурації. Інформація для регістра ції об'єктів з серверної активізацією укладена в елементі <service>. І елемент <service>, і елемент <channels> можуть здобути кілька вкладених елементів. Як видно з цього прикладу, можливості конфігураційних файлів просто вражаючі  
клієнтську програму  
Щоб створити екземпляр типу, активируемого сервером, клієнтські програми повинні бути налаштовані програмним чином або за допомогою файлу конфігурації. У разі налаштування програми програмним чином незалежно від способу активації є два способи отримання об'єкта  
1 спосіб - з попередньою реєстрацією Activated - об'єкта з боку клієнта  
Спочатку проводиться реєстрація методом  
RemotingConfiguration.RegisterActivatedClientType(typeof(имя\_объекта),"протокол://адрес:порт");  
                
             Object obj = null;  
              obj = new ім'я об'єкта ();  
приклад:  
   
RemotingConfiguration.RegisterActivatedClientType (typeof (RemOb1), "TCP: // localhost: 888 /");  
Створення об'єкта для виклику методів:  
             RemOb1obj = null;  
              obj = new RemOb1 ();  
Використання об'єкта для виклику методів  
             DataTable tb = new DataTable ();  
             String SqlSt;  
             SqlSt = this.textBox1.Text;  
             tb = obj.GetADOTable (SqlSt);  
Повний опис методу реєстрації  
Метод RemotingConfiguration.RegisterActivatedClientType (ActivatedClientTypeEntry)  
Реєструє об'єкт Type з боку клієнта в якості типу, який може активуватися сервером.  
список перевантаження

|  |  |
| --- | --- |
| ім'я | Опис |
| RegisterActivatedClientType  (ActivatedClientTypeEntry) | Реєструє об'єкт Type, наданий ActivatedClientTypeEntry на стороні клієнта в якості типу, який може бути активований сервером по запиту клієнта |
| RegisterActivatedClientType (Type, String) | Реєструє об'єкт Type з боку клієнта в якості типу, який може активуватися сервером по зопросу клієнта, використовуючи дані параметри Type і String |

Реєструє об'єкт Type, записаний в наданий ActivatedClientTypeEntry на стороні клієнта в якості типу, який може бути активований сервером.  
Параметри entry Тип: System.Runtime.Remoting.ActivatedClientTypeEntry  
Параметри конфігурації для активованого клієнтом типу.  
Винятки: SecurityException Принаймні в одного з викликають операторів, що знаходяться у верхній частині стека викликів, відсутній дозвіл на налаштування типів і каналів віддаленого взаємодії.  
  
2 спосіб - без попередньої реєстрації  
У цьому випадку для створення на клієнті примірника об'єкта, який активується сервером, використовується метод Activator.CreateInstance  
У наступному прикладі коду показаний виклик методу Activator.CreateInstance  
             // Отримуємо Activated віддалений об'єкт сервера без реєстрації  
             obj = null;  
// Варіант1  
  this.obj = (RemOb1) Activator.CreateInstance <RemOb1> ();  
      
   // варіант2  
       this.obj = (RemOb1) Activator.CreateInstance (typeof (RemOb1), "TCP: // localhost: 888 /");  
  
             if (obj == null)  
             {  
                 this.label7.Text = "Сервер не визначається, об'єкт не створено";  
                 return;  
             }  
             else  
             {  
                 this.label7.Text = "Сервер визначається, Activated об'єкт створений успішно, без оегістраціі";  
  
виклик методів  
Після отримання об'єкта використовуючи точковий синтаксис можна викликати його методи, наприклад  
DataTabe dt = new DataTable ();  
String sqls = @ "select \* from Personal"  
dt = obj.GetADOTable (sqls)  
Використання конфігураційного файлу для клієнтського застосування  
Другий метод, що надається .NET Remoting для налаштування параметрів віддаленого взаємодії, використовує конфігураційні файли. Раніше було розглянуто використанні конфігураційних файлів в розділі «Реалізація програми JobServer». При налаштування клієнтського додатка застосовуються інші теги. Ось так виглядає XML-код конфігураційного файлу програми JobClient:  
<configuration>  
     <system.runtime.remoting>  
        Opplication name = "JobCHent">  
            <client>  
                <wellknown  
                   type = "JobserverLib.JobServerlmpl, JobServerLib"  
                   url = "http: // localhost: 4000 /" />  
            </ client>  
            <channels>  
                <channel ref = "http" port = "0" />  
            </ channels>  
        </ application>  
     </system.runtime.remoting>  
</ configuration>  
Використання файлу JobClient.exe.config для настройки віддаленого взаємодії шляхом виклику RemotingConfiguration.Configure:  
private IJobServe r GetJobServerQ  
{  
      RemotingConfiguration.Configure (@ ".. \ .. \ JobClient.exe.config"  
      return (IJobServer) new JobServerImpl;  
};