**Приклад реалізації системи REMOTING**

Створення REMOTING –Об’єкта

using System;

using System.Collections.Generic;

using System.Windows.Forms;

using System.Data;

using System.ComponentModel;

using System.Data.OleDb;

using System.Xml;

using System.Text;

namespace WAppRemOb1

{

public class RemOb1 : System.MarshalByRefObject

{

public RemOb1()

{

// MessageBox.Show("Constructor RemOb1 called");

}

~RemOb1()

{

// MessageBox.Show("Destructor RemOb1 called");

}

private static XmlDataDocument xdd = null;

// свойство для хранения ссылки на экземпляр XmlDataDocument

private static XmlDataDocument DataDocument

{

get

{

if (xdd == null)

{ // объявить и создать рабочий документ

XmlDataDocument doc = new XmlDataDocument();

// прочитать схему документа

doc.DataSet.ReadXmlSchema("dataschema.xsd");

// загрузить документ

doc.Load("data.xml");

// обновить документ

doc.DataSet.AcceptChanges();

// сформировать xdd

xdd = doc;

}

return xdd;

}

}

public DataSet GetPreparedDataSet()

{ // метод возвращения клиенту по запросу набора данных

// MessageBox.Show(" Metod GetPreparedDataSet called");

return DataDocument.DataSet;

}

public void Save(DataSet dts)

{ // метод слияния исходного и полученного от клиента набора данных и записи итогового результата

// MessageBox.Show(" Metod Save called");

xdd.DataSet.Merge(dts);

xdd.Save("data.xml");

}

public string GetNewName(string name)

{

// MessageBox.Show(" Metod GetTable called");

return "NewName for " + name+ " is agent"+name;

}

public DataTable GetADOTable(string name)

{ String connstr = " Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\\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);

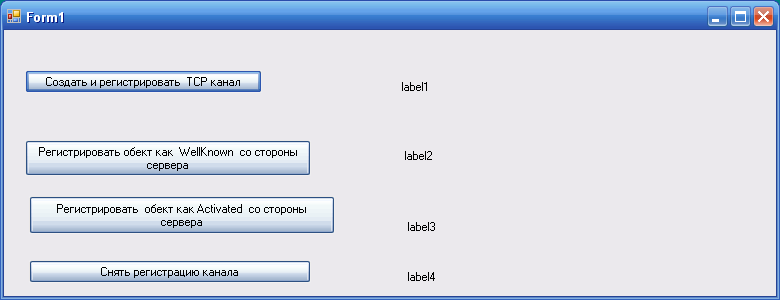
// MessageBox.Show(" Metod GetADOTable called");

return (restab);

}

}

}



СОЗДАНИЕ REMOTING -СЕРВЕРА

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Runtime.Remoting;

using System.Runtime.Remoting.Channels;

//using System.Runtime.Remoting.Channels;

using System.Drawing;

using WAppRemOb1;

using System.Text;

using System.Windows.Forms;

using System.Runtime.Remoting.Channels.Http;

using System.Runtime.Remoting.Channels.Tcp;

namespace WAppRemOb1Server

{

public partial class Form1 : Form

{

TcpChannel channel;

public Form1()

{

InitializeComponent();

}

private void btCrRegChan\_Click(object sender, EventArgs e)

{

// Создаем Tcp канал channel

channel = new TcpChannel(888);

// Регистрируем канал channel;

ChannelServices.RegisterChannel(channel, false);

this.label1.Text = "Канал создан и зарегистрирован";

}

private void button1\_Click(object sender, EventArgs e)

{

// регистрируем со стороны сервера RemOb1 объект как объект типа WellKnown ( управляемый сервером, c заданием URI) - обработчика однопотоковых сообщений

// для каждого вызова создается новый екземпляр обекта

RemotingConfiguration.RegisterWellKnownServiceType(typeof(RemOb1), "Rserv", WellKnownObjectMode.SingleCall);

this.label2.Text = "объект зарегистрирован как WellKnown со стороны сервера";

}

private void button2\_Click(object sender, EventArgs e)

{

// регистрируем со стороны сервера RemOb1 объект как объект типа Activated( управляемый клиентом, без URI)

RemotingConfiguration.RegisterActivatedServiceType(typeof(RemOb1));

this.label3.Text = " регистрация обекта Activated со стороны сервера прошла";

}

private void button3\_Click(object sender, EventArgs e)

{

// разрегистрируем канал channel;

// ChannelServices.UnregisterChannel(channel);

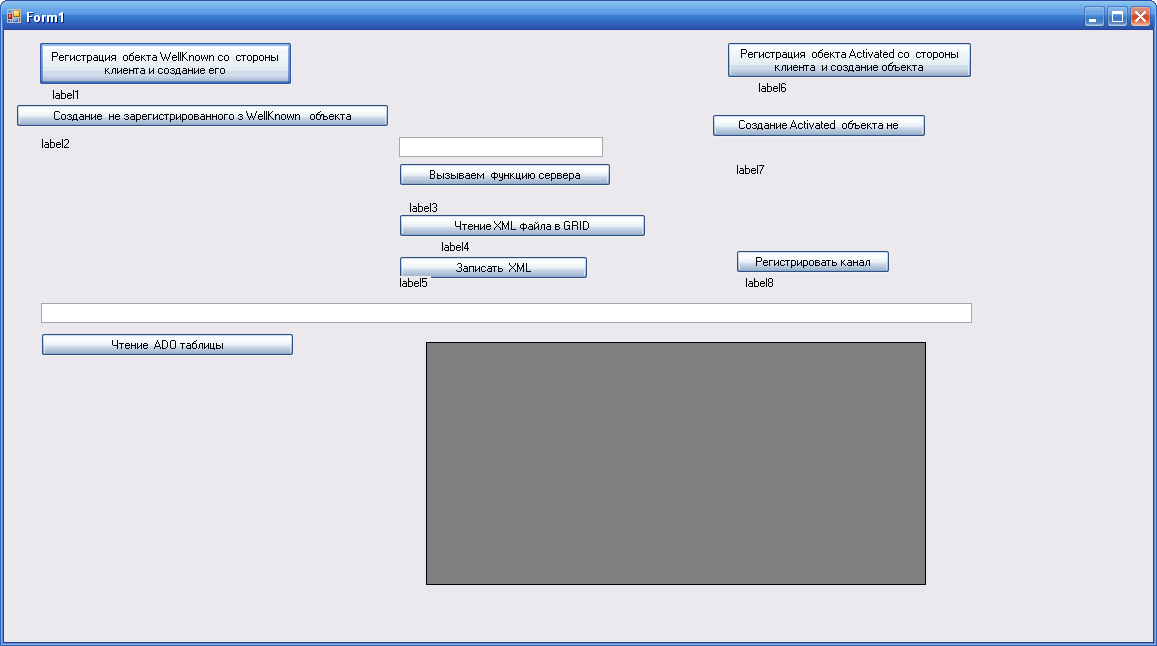
// this.label4.Text = "Канал отрегистрирован";

}

}

}

СОЗДАНИЕ REMOTING - КЛИЕНТА



using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Runtime.Remoting;

using System.Runtime.Remoting.Channels;

using System.Runtime.Remoting.Channels.Http;

using System.Runtime.Remoting.Services;

using WAppRemOb1;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace WAppRemOb1Klient

{

public partial class Form1 : Form

{

RemOb1 obj;

private DataGrid dg = new DataGrid();

public Form1()

{

// RemOb1 obj;

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

// регистрация WellKnown обекта со стороны клиента

RemotingConfiguration.RegisterWellKnownClientType(typeof(RemOb1),"TCP://localhost:888/Rserv");

this.label1.Text = " регистрация обекта со стороны клиента прошла";

obj = null;

obj = new RemOb1();

if (obj == null)

{

this.label1.Text = "Сервер не определяется, WellKnown обект не создан";

return;

}

else

{

this.label1.Text = "Сервер определяется успешно, WellKnown обект создан !";

}

}

private void button2\_Click(object sender, EventArgs e)

{ // получаем WellKnown удаленный объект сервера ( зарегистрированний на сервере) без предварительной регистрации на клиенте

this.obj = (RemOb1)Activator.GetObject(typeof(RemOb1), "TCP://localhost:888/Rserv");

if (obj == null)

{

this.label2.Text = "Сервер не определяется, WellKnown обект не создан";

return;

}

else

{

this.label2.Text = "Сервер определяется успешно, WellKnown обект создан !";

}

}

private void button3\_Click(object sender, EventArgs e)

{

// Вызов метода GetNewName() удаленного Remoting объекта obj типа RemOb1

if (obj != null)

{

string name = this.textBox2.Text;

this.label3.Text = "Результат вызова функции - " + this.obj.GetNewName(name);

}

else

{

this.label3.Text = " объект не создан, функции - не доступны ";

}

}

private void button4\_Click(object sender, EventArgs e)

{

// Вызов метода GetPreparedDataSet() удаленного Remoting объекта obj типа RemOb1

this.dataGridView1.DataSource = obj.GetPreparedDataSet();

this.dataGridView1.DataMember = "record";

this.label4.Text =" XML файл получен";

// InitControls();

}

private void InitControls()

{

}

private void SaveClick(object sender, EventArgs ea)

{

// SaveChanges();

}

private void SaveChanges()

{

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void button5\_Click(object sender, EventArgs e)

{

// Вызов метода Save() удаленного Remoting объекта obj типа RemOb1

DataSet cds = new DataSet();

//((DataSet)this.dataGridView1.DataSource).GetChanges(DataRowState.Added);

((DataSet)this.dataGridView1.DataSource).AcceptChanges() ;

cds = ((DataSet)this.dataGridView1.DataSource);

obj.Save(cds);

this.label5.Text = " XML файл записан";

}

private void button6\_Click(object sender, EventArgs e)

{

// Вызов метода GetADOTable() удаленного Remoting объекта obj типа RemOb1

DataTable tb = new DataTable();

String SqlSt;

SqlSt = this.textBox1.Text;

tb = obj.GetADOTable(SqlSt);

BindingSource binsu = new BindingSource();

binsu.DataSource = tb;

this.dataGridView1.DataSource = binsu;

// this.dataGridView1.DataMember = "record";

}

private void button7\_Click(object sender, EventArgs e)

{

// регистрация обекта Activated со стороны клиента и активизация его

RemotingConfiguration.RegisterActivatedClientType(typeof(RemOb1), "TCP://localhost:888/");

// this.label6.Text = " регистрация обекта Activated со стороны клиента прошла";

obj = null;

obj = new RemOb1();

if (obj == null)

{

this.label6.Text = "Сервер не определяется, Activated обект не создан";

return;

}

else

{

this.label6.Text = "Сервер определяется успешно, Activated обект создан !";

}

}

private void button8\_Click(object sender, EventArgs e)

{

// получаем Activated удаленный объект сервера без регистрации

obj = null;

this.obj = (RemOb1)Activator.CreateInstance<RemOb1>();

if (obj == null)

{

this.label7.Text = "Сервер не определяется, объект не создан";

return;

}

else

{

this.label7.Text = "Сервер определяется, Activated объект создан успешно , без оегистрации";

}

}

private void button9\_Click(object sender, EventArgs e)

{

// регистрируем клиентский канал

ChannelServices.RegisterChannel(new HttpClientChannel(),false);

this.label8.Text = " регистрация канала со стороны клиента прошла";

}

}

}