**Part 10 - Message Contract in WCF**

Con i Data Contracts abbiamo un controllo molto limitato sui messaggi di richiesta e risposta XML SOAP generati. Utilizza Message Contracts se vuoi avere il controllo completo sui messaggi XML SOAP generati.

Alcuni esempi di quando i contratti di messaggio possono essere utili

1. Includere alcuni dati personalizzati nell'intestazione SOAP. In generale, le intestazioni SOAP vengono utilizzate per passare credenziali utente, chiavi di licenza, chiavi di sessione ecc.

2. Modificare il nome dell'elemento wrapper nel messaggio SOAP o rimuoverlo del tutto.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

Immagine che contiene testo, schermata, Carattere, numero

Il contenuto generato dall'IA potrebbe non essere corretto.

In questa demo, parleremo di come usare MessageContracts 1. Per includere LicenseKey nell'intestazione SOAP 2. Cambiare il nome dell'elemento wrapper nel corpo SOAP

Messaggio di risposta SOAP generato utilizzando i contratti di messaggio

Immagine che contiene testo, schermata, Carattere, software

Il contenuto generato dall'IA potrebbe non essere corretto.

Decora una classe con l'attributo MessageContract, quindi usa quella classe come parametro di contratto operativo o tipo di ritorno. L'attributo MessageContract ha i seguenti parametri.

1. IsWrapped

2. WrapperName

3. WrapperNamespace

4. ProtectionLevel

L'attributo MessageHeader viene applicato a una proprietà della classe che si desidera includere nell'intestazione SOAP. L'attributo MessageHeader ha i seguenti parametri.

1. Name  
   2. Namespace  
   3. ProtectionLevel  
   4. Actor  
   5. MustUnderstand  
   6. Relay

L'attributo MessageBodyMember viene applicato a una proprietà della classe che vuoi includere nella sezione del corpo SOAP. L'attributo MessageBodyMember ha i seguenti parametri.

1. Nome 2.
2. Namespace 3.
3. Ordine
4. 4. ProtectionLevel

In generale, utilizzare MessageContract solo se c'è un motivo per modificare la struttura del messaggio XML SOAP.

**Employee.cs**  
using System;  
using System.Runtime.Serialization;  
using System.ServiceModel;  
namespace EmployeeService  
{  
    [MessageContract(IsWrapped=true,   
                     WrapperName="EmployeeRequestObject",   
                     WrapperNamespace="http://MyCompany.com/Employee")]  
    public class EmployeeRequest  
    {  
        [MessageBodyMember(Namespace = "http://MyCompany.com/Employee")]  
        public int EmployeeId { get; set; }  
  
        [MessageHeader(Namespace = "http://MyCompany.com/Employee")]  
        public string LicenseKey { get; set; }  
    }  
  
    [MessageContract(IsWrapped=true,   
                     WrapperName="EmployeeInfoObject",   
                     WrapperNamespace="http://MyCompany.com/Employee")]  
    public class EmployeeInfo  
    {  
        public EmployeeInfo()  
        {  
        }  
        public EmployeeInfo(Employee employee)  
        {  
            this.ID = employee.Id;  
            this.Name = employee.Name;  
            this.Gender = employee.Gender;  
            this.DOB = employee.DateOfBirth;  
            this.Type = employee.Type;  
            if (this.Type == EmployeeType.FullTimeEmployee)  
            {  
                this.AnnualSalary = ((FullTimeEmployee)employee).AnnualSalary;  
            }  
            else  
            {  
                this.HourlyPay = ((PartTimeEmployee)employee).HourlyPay;  
                this.HoursWorked = ((PartTimeEmployee)employee).HoursWorked;  
            }  
        }  
  
        [MessageBodyMember(Order = 1,Namespace = "http://MyCompany.com/Employee")]  
        public int ID { get; set; }  
        [MessageBodyMember(Order = 2,Namespace = "http://MyCompany.com/Employee")]  
        public string Name { get; set; }  
        [MessageBodyMember(Order = 3,Namespace = "http://MyCompany.com/Employee")]  
        public string Gender { get; set; }  
        [MessageBodyMember(Order = 4,Namespace = "http://MyCompany.com/Employee")]  
        public DateTime DOB { get; set; }  
        [MessageBodyMember(Order = 5,Namespace = "http://MyCompany.com/Employee")]  
        public EmployeeType Type { get; set; }  
        [MessageBodyMember(Order = 6,Namespace = "http://MyCompany.com/Employee")]  
        public int AnnualSalary { get; set; }  
        [MessageBodyMember(Order = 7,Namespace = "http://MyCompany.com/Employee")]  
        public int HourlyPay { get; set; }  
        [MessageBodyMember(Order = 8,Namespace = "http://MyCompany.com/Employee")]  
        public int HoursWorked { get; set; }  
    }  
      
    [DataContract(Namespace = "http://MyCompany.com/Employee")]  
    public class Employee  
    {  
        [DataMember(Order = 1)]  
        public int Id { get; set; }  
  
        [DataMember(Order = 2)]  
        public string Name { get; set; }  
  
        [DataMember(Order = 3)]  
        public string Gender { get; set; }  
  
        [DataMember(Order = 4)]  
        public DateTime DateOfBirth { get; set; }  
  
        [DataMember(Order = 5)]  
        public EmployeeType Type { get; set; }  
    }  
  
    public enum EmployeeType  
    {  
        FullTimeEmployee = 1,  
        PartTimeEmployee = 2  
    }  
}  
  
**IEmployeeService.cs**  
using System.ServiceModel;  
namespace EmployeeService  
{  
    [ServiceContract]  
    public interface IEmployeeService  
    {  
        [OperationContract]  
        EmployeeInfo GetEmployee(EmployeeRequest employeeRequest);  
  
        [OperationContract]  
        void SaveEmployee(EmployeeInfo Employee);  
    }  
}  
  
**EmployeeService.cs**  
using System;  
using System.Data;  
using System.Data.SqlClient;  
using System.Configuration;  
namespace EmployeeService  
{  
    public class EmployeeService : IEmployeeService  
    {  
        public EmployeeInfo GetEmployee(EmployeeRequest employeeRequest)  
        {  
            Console.WriteLine("License Key = " + employeeRequest.LicenseKey);  
            Employee employee = null;  
            string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;  
            using (SqlConnection con = new SqlConnection(cs))  
            {  
                SqlCommand cmd = new SqlCommand("spGetEmployee", con);  
                cmd.CommandType = CommandType.StoredProcedure;  
                SqlParameter parameterId = new SqlParameter();  
                parameterId.ParameterName = "@Id";  
                parameterId.Value = employeeRequest.EmployeeId;  
                cmd.Parameters.Add(parameterId);  
                con.Open();  
                SqlDataReader reader = cmd.ExecuteReader();  
                while (reader.Read())  
                {  
                    if ((EmployeeType)reader["EmployeeType"] == EmployeeType.FullTimeEmployee)  
                    {  
                        employee = new FullTimeEmployee  
                        {  
                            Id = Convert.ToInt32(reader["Id"]),  
                            Name = reader["Name"].ToString(),  
                            Gender = reader["Gender"].ToString(),  
                            DateOfBirth = Convert.ToDateTime(reader["DateOfBirth"]),  
                            Type = EmployeeType.FullTimeEmployee,  
                            AnnualSalary = Convert.ToInt32(reader["AnnualSalary"])  
                        };  
                    }  
                    else  
                    {  
                        employee = new PartTimeEmployee  
                        {  
                            Id = Convert.ToInt32(reader["Id"]),  
                            Name = reader["Name"].ToString(),  
                            Gender = reader["Gender"].ToString(),  
                            DateOfBirth = Convert.ToDateTime(reader["DateOfBirth"]),  
                            Type = EmployeeType.PartTimeEmployee,  
                            HourlyPay = Convert.ToInt32(reader["HourlyPay"]),  
                            HoursWorked = Convert.ToInt32(reader["HoursWorked"]),  
                        };  
                    }  
                }  
            }  
            return new EmployeeInfo(employee);  
        }  
  
        public void SaveEmployee(EmployeeInfo employee)  
        {  
            string cs = ConfigurationManager.ConnectionStrings["DBCS"].ConnectionString;  
            using (SqlConnection con = new SqlConnection(cs))  
            {  
                SqlCommand cmd = new SqlCommand("spSaveEmployee", con);  
                cmd.CommandType = CommandType.StoredProcedure;  
                SqlParameter parameterId = new SqlParameter  
                {  
                    ParameterName = "@Id",  
                    Value = employee.ID  
                };  
                cmd.Parameters.Add(parameterId);  
  
                SqlParameter parameterName = new SqlParameter   
                {  
                    ParameterName = "@Name",  
                    Value = employee.Name  
                };  
                cmd.Parameters.Add(parameterName);  
  
                SqlParameter parameterGender = new SqlParameter   
                {  
                    ParameterName = "@Gender",  
                    Value = employee.Gender  
                };  
                cmd.Parameters.Add(parameterGender);  
  
                SqlParameter parameterDateOfBirth = new SqlParameter   
                {  
                    ParameterName = "@DateOfBirth",  
                    Value = employee.DOB  
                };  
                cmd.Parameters.Add(parameterDateOfBirth);  
  
                SqlParameter parameterEmployeeType = new SqlParameter   
                {  
                    ParameterName = "@EmployeeType",  
                    Value = employee.Type  
                };  
                cmd.Parameters.Add(parameterEmployeeType);  
  
                if (employee.Type == EmployeeType.FullTimeEmployee)  
                {  
                    SqlParameter parameterAnnualSalary = new SqlParameter   
                    {  
                        ParameterName = "@AnnualSalary",  
                        Value = employee.AnnualSalary  
                    };  
                    cmd.Parameters.Add(parameterAnnualSalary);  
                }  
                else  
                {  
                    SqlParameter parameterHourlyPay = new SqlParameter   
                    {  
                        ParameterName = "@HourlyPay",  
                        Value = employee.HourlyPay,  
                    };  
                    cmd.Parameters.Add(parameterHourlyPay);  
  
                    SqlParameter parameterHoursWorked = new SqlParameter   
                    {  
                        ParameterName = "@HoursWorked",  
                        Value = employee.HoursWorked  
                    };  
                    cmd.Parameters.Add(parameterHoursWorked);  
                }  
  
                con.Open();  
                cmd.ExecuteNonQuery();  
            }  
        }  
    }  
}  
  
**Client WebForm1.aspx.cs**  
protected void btnGetEmployee\_Click(object sender, EventArgs e)  
{  
    EmployeeService.IEmployeeService client =  
        new EmployeeService.EmployeeServiceClient();  
    EmployeeService.EmployeeRequest request = new EmployeeService.EmployeeRequest("XYZ120FABC", Convert.ToInt32(txtID.Text));  
  
    EmployeeService.EmployeeInfo employee = client.GetEmployee(request);

    if (employee.Type == EmployeeService.EmployeeType.FullTimeEmployee)  
    {  
        txtAnnualSalary.Text = employee.AnnualSalary.ToString();  
        trAnnualSalary.Visible = true;  
        trHourlPay.Visible = false;  
        trHoursWorked.Visible = false;  
    }  
    else  
    {  
        txtHourlyPay.Text = employee.HourlyPay.ToString();  
        txtHoursWorked.Text = employee.HoursWorked.ToString();  
        trAnnualSalary.Visible = false;  
        trHourlPay.Visible = true;  
        trHoursWorked.Visible = true;  
    }  
    ddlEmployeeType.SelectedValue = ((int)employee.Type).ToString();  
  
    txtName.Text = employee.Name;  
    txtGender.Text = employee.Gender;  
    txtDateOfBirth.Text = employee.DOB.ToShortDateString();  
    lblMessage.Text = "Employee retrieved";  
}  
  
protected void btnSave\_Click(object sender, EventArgs e)  
{  
    EmployeeService.IEmployeeService client = new  
        EmployeeService.EmployeeServiceClient();  
    EmployeeService.EmployeeInfo employee = new EmployeeService.EmployeeInfo();  
  
    if (ddlEmployeeType.SelectedValue == "-1")  
    {  
        lblMessage.Text = "Please select Employee Type";  
    }  
    else  
    {  
        if (((EmployeeService.EmployeeType)Convert.ToInt32(ddlEmployeeType.SelectedValue))  
            == EmployeeService.EmployeeType.FullTimeEmployee)  
        {  
            employee.AnnualSalary = Convert.ToInt32(txtAnnualSalary.Text);  
            employee.Type = EmployeeService.EmployeeType.FullTimeEmployee;  
        }  
        else if (((EmployeeService.EmployeeType)Convert.ToInt32(ddlEmployeeType.SelectedValue))  
            == EmployeeService.EmployeeType.PartTimeEmployee)  
        {  
            employee.HourlyPay = Convert.ToInt32(txtHourlyPay.Text);  
            employee.HoursWorked = Convert.ToInt32(txtHoursWorked.Text);

            employee.Type = EmployeeService.EmployeeType.PartTimeEmployee;  
        }  
  
        employee.ID = Convert.ToInt32(txtID.Text);  
        employee.Name = txtName.Text;  
        employee.Gender = txtGender.Text;  
        employee.DOB = Convert.ToDateTime(txtDateOfBirth.Text);  
  
        client.SaveEmployee(employee);  
        lblMessage.Text = "Employee saved";  
    }  
}  
  
protected void ddlEmployeeType\_SelectedIndexChanged  
    (object sender, EventArgs e)  
{  
    if (ddlEmployeeType.SelectedValue == "-1")  
    {  
        trAnnualSalary.Visible = false;  
        trHourlPay.Visible = false;  
        trHoursWorked.Visible = false;  
    }  
    else if (ddlEmployeeType.SelectedValue == "1")  
    {  
        trAnnualSalary.Visible = true;  
        trHourlPay.Visible = false;  
        trHoursWorked.Visible = false;  
    }  
    else  
    {  
        trAnnualSalary.Visible = false;  
        trHourlPay.Visible = true;  
        trHoursWorked.Visible = true;  
    }  
}