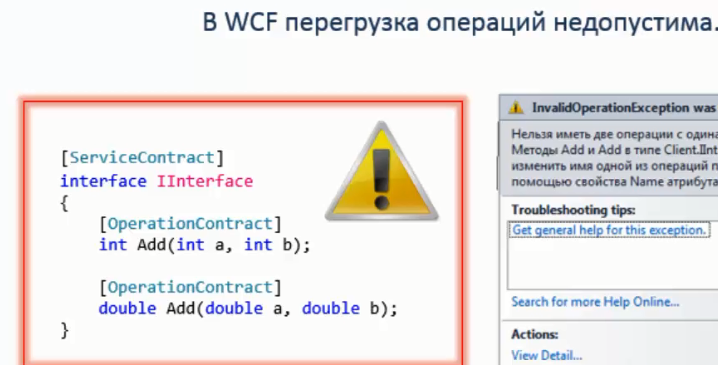
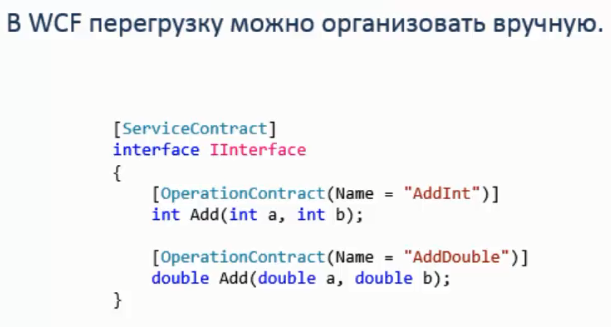


OperationContext.Current.RequestContext.RequestMessage.ToString()

Класс отвечающий за запросы от клиента, в данном случае показывает в SOAP данные которые пришли

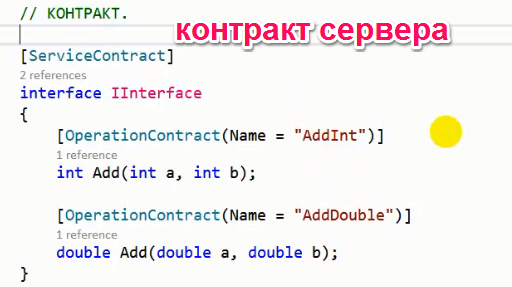


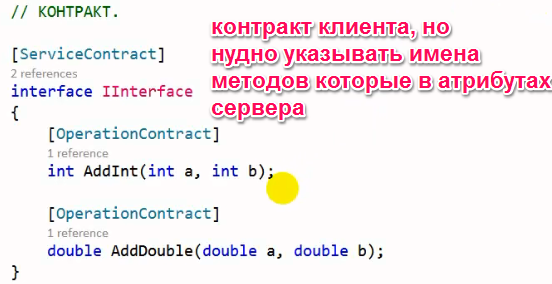


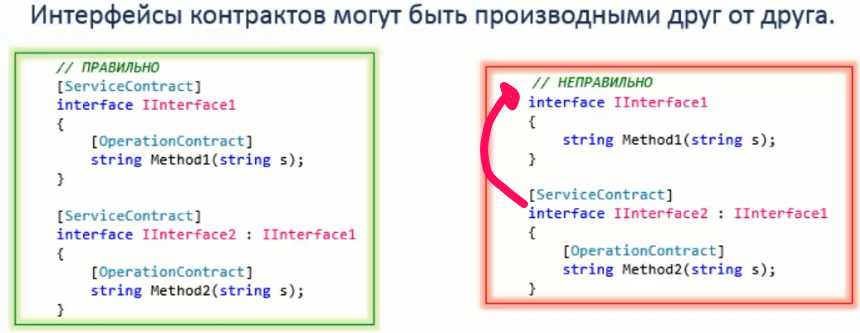
Перегрузка методов с использованием атрибутов

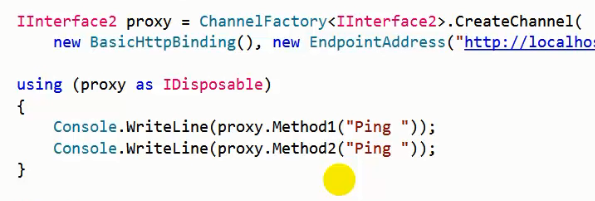
1.имена в атрибутах могут быть на стороне сервера, тогда на стороне клиента либо должны быть

указаны также атрибуты либо с такими же именами методы (можно и на оборот атрибуты на клиенте реальные имена на сервере)









Под каждый контракт создается порт

серевер

ServiceHost host = new ServiceHost(typeof(MyService));

host.AddServiceEndpoint(typeof(IInterface1), new BasicHttpBinding(), "http://localhost:80");

host.AddServiceEndpoint(typeof(IInterface2), new BasicHttpBinding(), "http://localhost:81");

host.Open();

клиент

IInterface1 proxy1 = ChannelFactory<IInterface1>.CreateChannel(

new BasicHttpBinding(), new EndpointAddress("http://localhost:80"));

IInterface2 proxy2 = ChannelFactory<IInterface2>.CreateChannel(

new BasicHttpBinding(), new EndpointAddress("http://localhost:81"));

using (proxy1 as IDisposable)

{

Console.WriteLine(proxy1.Method1("Ping ") + "\n");

}

using (proxy2 as IDisposable)

{

Console.WriteLine(proxy2.Method1("Ping "));

Console.WriteLine(proxy2.Method2("Ping "));

}

В данном случае под одним портом два сервиса, но разные имена сервисов на каждый контракт

ServiceHost host = new ServiceHost(typeof(MyService));

host.AddServiceEndpoint(typeof(IInterface1), new BasicHttpBinding(), http://localhost:80/A");

host.AddServiceEndpoint(typeof(IInterface2), new BasicHttpBinding(), "http://localhost:80/B");

host.Open();

клиент

IInterface1 proxy1 = ChannelFactory<IInterface1>.CreateChannel(

new BasicHttpBinding(), new EndpointAddress("http://localhost:80/A"));

IInterface2 proxy2 = ChannelFactory<IInterface2>.CreateChannel(

new BasicHttpBinding(), new EndpointAddress("http://localhost:80/B"));

using (proxy1 as IDisposable)

{

Console.WriteLine(proxy1.Method1("Ping -> ") + "\n");

}

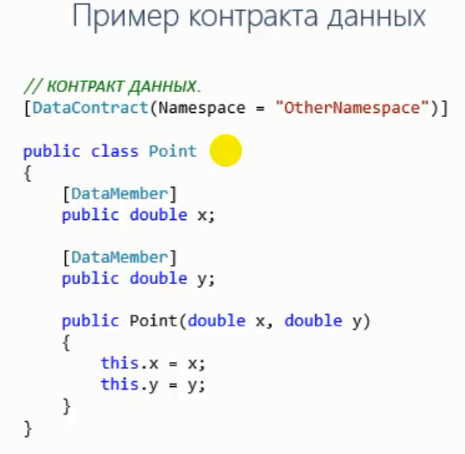
using (proxy2 as IDisposable)

{

Console.WriteLine(proxy2.Method1("Ping -> "));

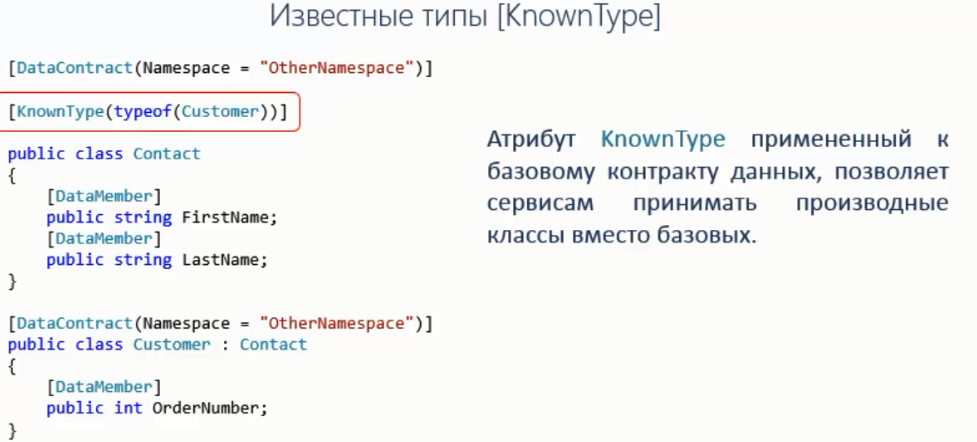
Console.WriteLine(proxy2.Method2("Ping -> "));

}

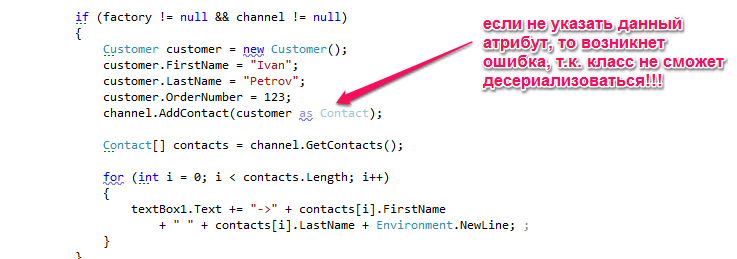


Атрибут [KnownType(typeof(Customer))]

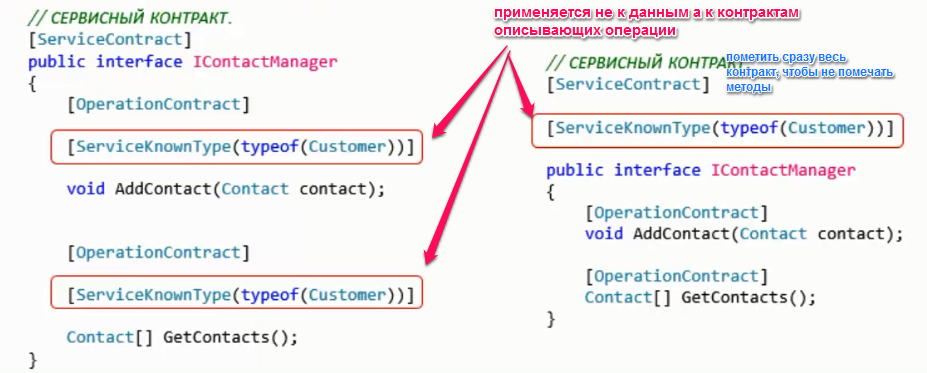
public class Contact



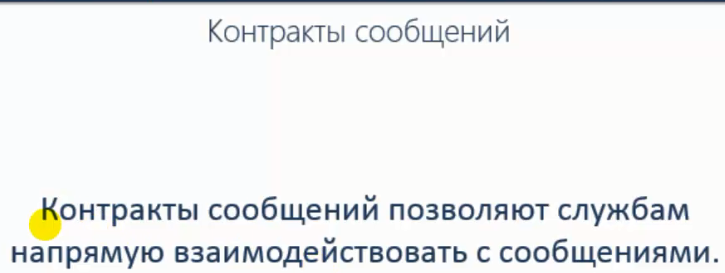
На стороне клиента передаем на сторону сервера



[ServiceKnownType(typeof(Customer))]



Контракты сообщений



Класс который будет контрактом данных (передаваться в методы) помечается атрибутом

// КОНТРАКТ СООБЩЕНИЯ.

[MessageContract]

public class MyMessage

а его свойства

[MessageHeader]

public string Operation

{

get { return operation; }

set { operation = value; }

}

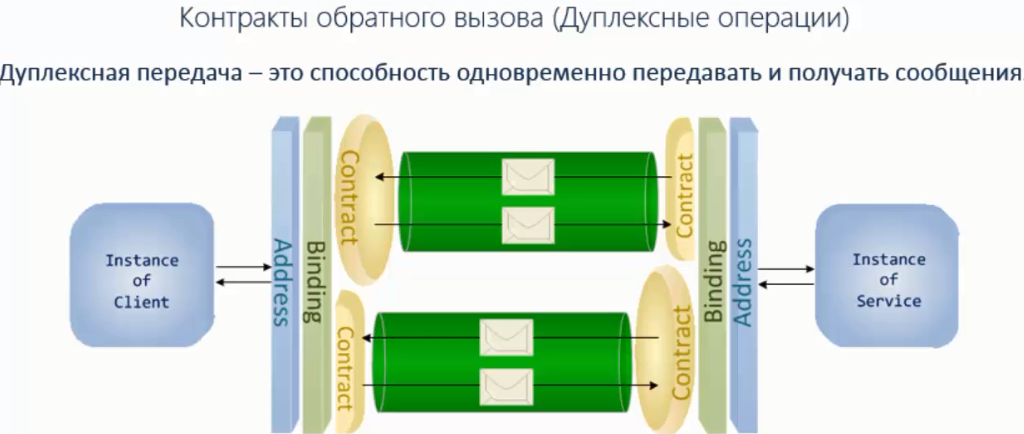
[MessageBodyMember]

public double N1

{

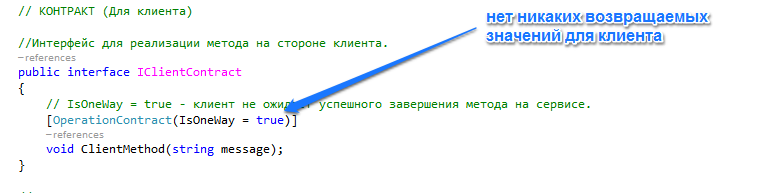
get { return n1; }

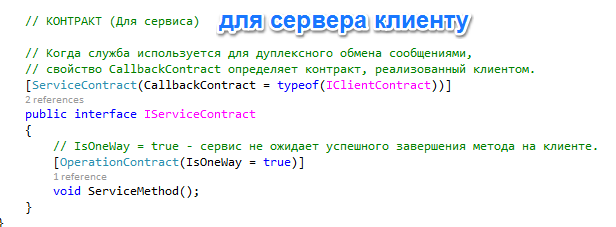
set { n1 = value; }

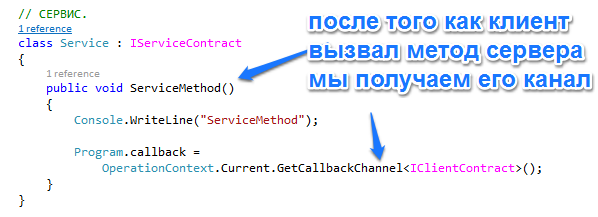


IsOneWay = true - клиент не ожидает успешного завершения метода на сервисе.

[OperationContract(IsOneWay = true)]



,



Сервер



Сторона клиента

