**How to: Create a One-Way Contract**

This topic shows the basic steps to create methods that use a one-way contract. Such methods invoke operations on a Windows Communication Foundation (WCF) service from a client but do not expect a reply. This type of contract can be used, for example, to publish notifications to many subscribers. You can also use one-way contracts when creating a duplex (two-way) contract, which allows clients and servers to communicate with each other independently so that either can initiate calls to the other. This can allow, in particular, the server to make one-way calls to the client that the client can treat as events. For detailed information about specifying one-way methods, see the [IsOneWay](http://msdn.microsoft.com/en-us/library/system.servicemodel.operationcontractattribute.isoneway.aspx) property and the [OperationContractAttribute](http://msdn.microsoft.com/en-us/library/system.servicemodel.operationcontractattribute.aspx) class.

For more information about creating a client application for a duplex contract, see [How to: Access WCF Services with One-Way and Request-Reply Contracts](http://msdn.microsoft.com/en-us/library/ms732232.aspx). For a working sample, see the [One-Way](http://msdn.microsoft.com/en-us/library/ms751496.aspx) sample.

**To create a one-way contract**

1. Create the service contract by applying the [ServiceContractAttribute](http://msdn.microsoft.com/en-us/library/system.servicemodel.servicecontractattribute.aspx) class to the interface that defines the methods the service is to implement.
2. Indicate which methods in the interface a client can invoked by applying the **OperationContractAttribute** class to them.
3. Designate operations that must have no output (no return value and no out or ref parameters) as one-way by setting the **IsOneWay** property to **true**. Note that the operations that carry the **OperationContractAttribute** class satisfy a request-reply contract by default because the **IsOneWay** property is **false** by default. So you must explicitly specify the value of the attribute property to be **true** if you want a one-way contract for the method.

**Example**

The following code example defines a contract for a service that includes several one-way methods. All of the methods have one-way contracts except **Equals**, which defaults to request-reply and returns a result.

Visual Basic

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl06_code');" \o "Copy Code)

<ServiceContract(Namespace:="http://Microsoft.ServiceModel.Samples", SessionMode:=SessionMode.Required)> \_

Public Interface ICalculatorSession

<OperationContract(IsOneWay:=True)> \_

Sub Clear()

<OperationContract(IsOneWay:=True)> \_

Sub AddTo(ByVal n As Double)

<OperationContract(IsOneWay:=True)> \_

Sub SubtractFrom(ByVal n As Double)

<OperationContract(IsOneWay:=True)> \_

Sub MultiplyBy(ByVal n As Double)

<OperationContract(IsOneWay:=True)> \_

Sub DivideBy(ByVal n As Double)

<OperationContract()> \_

Function Equal() As Double

End Interface

C#

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl07_code');" \o "Copy Code)

[ServiceContract(Namespace="http://Microsoft.ServiceModel.Samples", SessionMode=SessionMode.Required)]

public interface ICalculatorSession

{

[OperationContract(IsOneWay=true)]

void Clear();

[OperationContract(IsOneWay = true)]

void AddTo(double n);

[OperationContract(IsOneWay = true)]

void SubtractFrom(double n);

[OperationContract(IsOneWay = true)]

void MultiplyBy(double n);

[OperationContract(IsOneWay = true)]

void DivideBy(double n);

[OperationContract]

double Equals();

}