**Implementing Service Contracts**

A service is a class that exposes functionality available to clients at one or more endpoints. To create a service, write a class that implements a Windows Communication Foundation (WCF) contract. You can do this in one of two ways. You can define the contract separately as an interface and then create a class that implements that interface. Alternatively, you can create the class and contract directly by placing the [ServiceContractAttribute](http://msdn.microsoft.com/en-us/library/system.servicemodel.servicecontractattribute.aspx) attribute on the class itself and the [OperationContractAttribute](http://msdn.microsoft.com/en-us/library/system.servicemodel.operationcontractattribute.aspx) attribute on the methods available to the clients of the service.

**Creating a service class**

The following is an example of a service that implements an **IMath** contract that has been defined separately.

C#

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl03_code');" \o "Copy Code)

// Define the IMath contract.

[ServiceContract]

public interface IMath

{

[OperationContract]

double Add(double A, double B);

[OperationContract]

double Multiply (double A, double B);

}

// Implement the IMath contract in the MathService class.

public class MathService : IMath

{

public double Add (double A, double B) { return A + B; }

public double Multiply (double A, double B) { return A \* B; }

}

Alternatively, a service can expose a contract directly. The following is an example of a service class that defines and implements a **MathService** contract.

C#

[Copy Code](javascript:CopyCode('ctl00_MTCS_main_ctl04_code');" \o "Copy Code)

// Define the MathService contract directly on the service class.

[ServiceContract]  
class MathService

{

[OperationContract]  
 public double Add(double A, double B) { return A + B; }

[OperationContract]

private double Multiply (double A, double B) { return A \* B; }

}

Note that the preceding services expose different contracts because the contract names are different. In the first case, the exposed contract is named "**IMath**" while in the second case the contract is named "**MathService**".

You can set a few things at the service and operation implementation levels, such as concurrency and instancing. For more information, see [Designing and Implementing Services](http://msdn.microsoft.com/en-us/library/ms729746.aspx).

After implementing a service contract, you must create one or more endpoints for the service. For more information, see [Endpoint Creation Overview](http://msdn.microsoft.com/en-us/library/ms735096.aspx). For more information about how to run a service, see [Hosting Services](http://msdn.microsoft.com/en-us/library/ms730158.aspx).