This page is specific to

Microsoft Visual Studio 2010/.NET Framework 4

**Using Bindings to Configure Services and Clients**

Bindings are objects that specify the communication details required to connect to an endpoint. More specifically, bindings contain configuration information that is used to create the client or service runtime by defining the specifics of transports, wire-formats (message encoding), and protocols to use for the respective endpoint or client channel. To create a functioning Windows Communication Foundation (WCF) service, each endpoint in the service requires a binding. This topic explains what bindings are, how they are defined, and how a particular binding is specified for an endpoint.

**What a Binding Defines**

The information in a binding can be very basic or very complex. The most basic binding specifies only the transport protocol (such as HTTP) that must be used to connect to the endpoint. More generally, the information a binding contains about how to connect to an endpoint falls into one of the categories in the following table.

Protocols

Determines the security mechanism being used, either reliable messaging capability or transaction context flow settings.

Transport

Determines the underlying transport protocol to use (for example, TCP or HTTP).

Encoding

Determines the message encoding, for example, text/XML, binary, or Message Transmission Optimization Mechanism (MTOM), which determines how messages are represented as byte streams on the wire.

**System-Provided Bindings**

WCF includes a set of system-provided bindings that are designed to cover most application requirements and scenarios. The following classes represent some examples of system-provided bindings:

* [BasicHttpBinding](http://msdn.microsoft.com/en-us/library/system.servicemodel.basichttpbinding.aspx): An HTTP protocol binding suitable for connecting to Web services that conforms to the WS-I Basic Profile 1.1 specification (for example, ASP.NET Web services [ASMX]-based services).
* [WsHttpBinding](http://msdn.microsoft.com/en-us/library/system.servicemodel.wshttpbinding.aspx): An HTTP protocol binding suitable for connecting to endpoints that conform to the Web services specifications protocols.
* [NetNamedPipeBinding](http://msdn.microsoft.com/en-us/library/system.servicemodel.netnamedpipebinding.aspx): Uses the .NET binary encoding and framing technologies in conjunction with the Windows named pipe transport to connect to other WCF endpoints on the same machine.
* [NetMsmqBinding](http://msdn.microsoft.com/en-us/library/system.servicemodel.netmsmqbinding.aspx): Uses the .NET binary encoding and framing technologies in conjunction with the Message Queuing (also known as MSMQ) to create queued message connections with other WCF endpoints.

For a complete list of system-provided bindings, with descriptions, see [System-Provided Bindings](http://msdn.microsoft.com/en-us/library/ms730879.aspx).

**Custom Bindings**

If the system-provided binding collection does not have the correct combination of features that a service application requires, you can create a [CustomBinding](http://msdn.microsoft.com/en-us/library/system.servicemodel.channels.custombinding.aspx) binding. For more information about the elements of a **CustomBinding** binding, see [<customBinding>](http://msdn.microsoft.com/en-us/library/ms731377.aspx).

**Using Bindings**

Using bindings entails two basic steps:

1. Select or define a binding. The easiest method is to choose one of the system-provided bindings and use its default settings. You can also choose a system-provided binding and reset its property values to suit your requirements. Alternatively, you can create a custom binding and set every property as required.
2. Create an endpoint that uses this binding.

**Code and Configuration**

You can define or configure bindings through code or configuration. These two approaches are independent of the type of binding used, for example, whether you are using a system-provided or a **CustomBinding** binding. In general, using code gives you complete control over the definition of a binding when you compile. Using configuration, on the other hand, allows a system administrator or the user of a WCF service or client to change the parameters of bindings. This flexibility is often desirable because there is no way to predict the specific machine requirements and network conditions into which a WCF application is to be deployed. Separating the binding (and addressing) information from the code allows administrators to change the binding details without having to recompile or redeploy the application. Note that if the binding is defined in code, it overwrites any configuration-based definitions made in the configuration file. For examples of these approaches, see the following topics:

* [How to: Host a WCF Service in a Managed Application](http://msdn.microsoft.com/en-us/library/ms731758.aspx) provides an example of creating a binding in code.
* [How to: Configure a Basic Windows Communication Foundation Client](http://msdn.microsoft.com/en-us/library/ms734663.aspx) provides an example of creating a client using configuration.