BinaryFormatter

SoapFormatter

XmlSerializer

The BinaryFormatter type serializes your object’s state to a stream using a compact binary format.

This type is defined within the System.Runtime.Serialization.Formatters.Binary namespace that is part of **mscorlib.dll**. If you want to gain access to this type, you can specify the following C# using

directive:

// Gain access to the BinaryFormatter in mscorlib.dll.

**using System.Runtime.Serialization.Formatters.Binary;**

The SoapFormatter type persists an object’s state as a SOAP message (the standard XML format for passing messages to/from a web service). This type is defined within the

System.Runtime.Serialization.Formatters.Soap namespace, which is defined in a *separate*

*assembly*.

Thus, to format your object graph into a SOAP message, you must first set a reference to

System.Runtime.Serialization.Formatters.Soap.dll using the Visual Studio Add Reference dialog box

and then specify the following C# using directive:

// Must reference **System.Runtime.Serialization.Formatters.Soap.dll.**

**using System.Runtime.Serialization.Formatters.Soap;**

Finally, if you want to persist a tree of objects as an XML document, you can use the XmlSerializer type.

To use this type, you need to specify that you are using the System.Xml.Serialization namespace and set a reference to the assembly System.Xml.dll. As luck would have it, all Visual Studio project templates automatically reference System.Xml.dll; therefore, all you need to do is use the following

namespace:

// Defined within **System.Xml.dll.**

**using System.Xml.Serialization;**

However, the BinaryFormatter and SoapFormatter types do support common members through theimplementation of the IFormatter and IRemotingFormatter interfaces (strange as it might seem, theXmlSerializer implements neither).

**System.Runtime.Serialization.IFormatter** defines the core Serialize() and Deserialize()