Serialization in .NET

Article • 09/23/2022 • 2 minutes to read

Serialization is the process of converting the state of an object into a form that can be persisted or transported. The complement of serialization is deserialization, which converts a stream into an object. Together, these processes allow data to be stored and transferred.

.NET features the following serialization technologies:

- JSON serialization serializes only public properties and does not preserve type fidelity. JSON is an open standard that is an attractive choice for sharing data across the web.
- Binary serialization preserves *type fidelity*, which means that the complete state of the object is recorded and when you deserialize, an exact copy is created. This type of serialization is useful for preserving the state of an object between different invocations of an application. For example, you can share an object between different applications by serializing it to the Clipboard. You can serialize an object to a stream, to a disk, to memory, over the network, and so forth. Remoting uses serialization to pass objects "by value" from one computer or application domain to another.
- XML and SOAP serialization serializes only `public` properties and fields and does not preserve type fidelity. This is useful when you want to provide or consume data without restricting the application that uses the data. Because XML is an open standard, it is an attractive choice for sharing data across the Web. SOAP is likewise an open standard, which makes it an attractive choice.

Reference

System.Text.Json
Contains classes that can be used to serialize objects into JSON format documents or streams.

System.Runtime.Serialization
Contains classes that can be used for serializing and deserializing objects.

System.Xml.Serialization
Contains classes that can be used to serialize objects into XML format documents or streams.