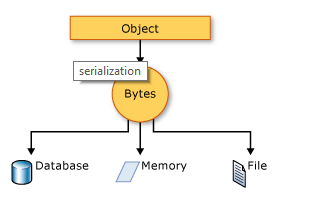
**Serialization and Deserialization**

Serialization is the process of converting an object into a stream of bytes in order to store the object or transmit it to memory, a database, or a file. Its main purpose is to save the state of an object in order to be able to recreate it when needed. The reverse process is called deserialization.

How Serialization Works

This illustration shows the overall process of serialization.



### **Uses for Serialization**

Serialization allows the developer to save the state of an object and recreate it as needed, providing storage of objects as well as data exchange. Through serialization, a developer can perform actions like sending the object to a remote application by means of a Web Service, passing an object from one domain to another, passing an object through a firewall as an XML string, or maintaining security or user-specific information across applications.

**Why Serialization:**   
Writing crucial data to the disk as TEXT is always dangerous. Any anonymous user can open the text file and easily read your data. With Object Serialization, you can reduce this danger to a certain extent. You can write any complex object directly to a filestream without converting values of individual properties into a text. You can make the data written, to the disk, at least not human readable. In order for the users to read your data files, they have to use your program. Like a File Open command which you may provide in your application.

**Types**

* Binary Serialization
* XML Serialization
* JSON Serialization

Ref:   
<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/serialization/>