Programming the .NET Framework 4.5

Module 04 - Serialization

In This Chapter

- Motivation for serialization
- Marking a type for serialization
- BinaryFormatter
- Controlling serialization
- Custom serialization
- Overview of XML serialization
- Overview of DataContract serialization
- Lab

What's Serialization?

- *Serialization is the process of saving an object onto a storage medium (such as a file, or a memory buffer)…*

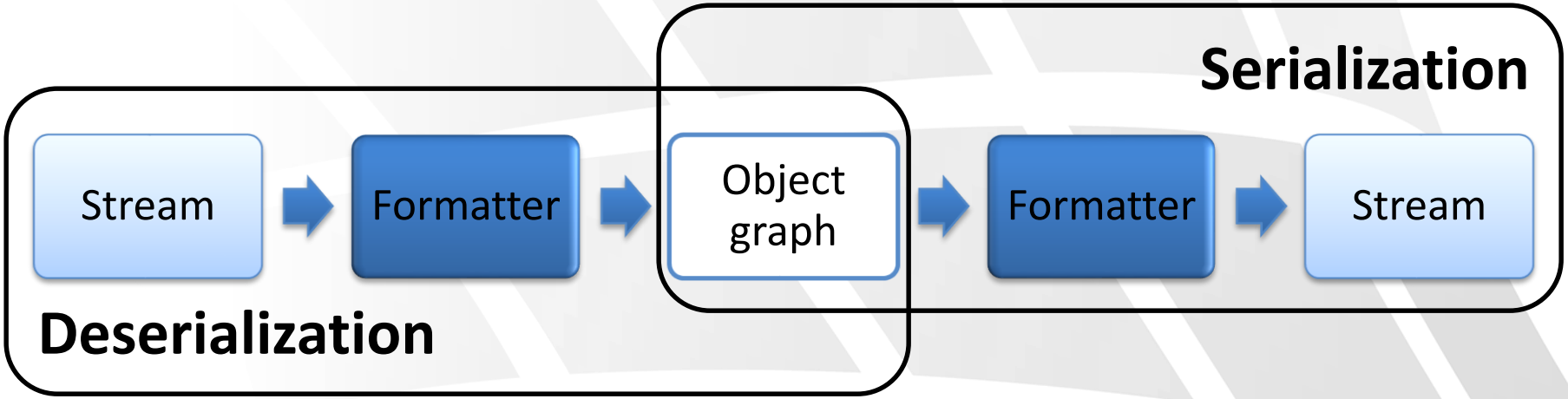
http://en.wikipedia.org/wiki/Serialization

Why Serialization?

- Persistence
- Communication
- Logging

.NET Serialization

- All primitive types are serializable
- Mark custom types with [Serializable]



Serializable User Class

```
1  [Serializable]
2  class User
3  {
4      private string _name;
5      private string _password;
6      private DateTime _loginTime;
7      private int _reputation;
9      public User(string name, string password)
10     {
11         _name = name;
12         _password = password;
13         _loginTime = DateTime.Now;
14         _reputation = ReputationDB.Reputation(name);
15     }
16 }
```

BinaryFormatter

```
1 User joe = new User("Joe", "123456");
2 BinaryFormatter formatter = new BinaryFormatter();
3 FileStream file = File.Create("joe.serialized");
4 formatter.Serialize(file, joe);
5 file.Close();
6
7 file = File.Open("joe.serialized", FileMode.Open);
8 joe = (User)formatter.Deserialize(file);
9 file.Close();
```

- Implements Formatter
- Serialize and Deserialize methods

Serialization with BinaryFormatter

Demo

Controlling Serialization

- Some fields can't be serialized
  - Volatile data
  - Sensitive data
- Use [NonSerialized]

```
1 [Serializable]
2 class User2
3 {
4     private string _name;
5     [NonSerialized] private string _password;
6     private DateTime _loginTime;
7     private int _reputation;
```

Serialization Callbacks

- [OnSerializing]
- [OnSerialized]
- [OnDeserializing]
- [OnDeserialized]

Deserialization Callbacks

Demo

Custom Serialization

- Implement `ISerializable`
- Add a protected constructor taking `SerializationInfo`, `StreamingContext`

Custom Serialization Example

```
1  [Serializable]
2  class User4 : ISerializable
3  {
4      void ISerializable.GetObjectData(
5        SerializationInfo info, StreamingContext context)
6      {
7          info.AddValue("Name", _name);
8          info.AddValue("Password", Encrypt(_password));
9      }
10     protected User4(
11       SerializationInfo info, StreamingContext context)
12     {
13         _name = info.GetString("Name");
14         _password = Decrypt(info.GetString("Password“));
15         InitLoginTimeAndReputation();
16     }
```

Serialization Alternatives

- `SoapFormatter` – deprecated
- `XmlSerializer` – public data, no cycles
- `Data Contract Serialization` – WCF
- JSON Serialization

Data Contract (WCF)

- System.Runtime.Serialization (reference)

```
[DataContract(Name="Foo")]
public class FooDataContract
{
    [DataMember(Name="id")]
    public int Id { get; set; }
    [DataMember]
    public string Name { get; set; }
    public string Pass { get; set; }
}
```

Data Contract (WCF)

- Newtonsoft.Json (NuGet)

```
[JsonObject(Title="Foo")]
public class FooDataContract
{
    [JsonProperty(PropertyName="id")]
    public int Id { get; set; }
    [JsonProperty]
    public string Name { get; set; }
    public string Pass { get; set; }
}
```

SOAP, XML and DataContract Serialization

Demo

Serialization Framework

Lab

Summary

- Motivation for serialization
- Marking a type for serialization
- BinaryFormatter
- Controlling serialization
- Custom serialization
- Overview of XML serialization
- Overview of DataContract serialization
- Lab

Questions ?