Serialization

Saving and loading complex data structures

Contents

- What is Serialization?
- The serialization process
- XML Files
- Serialization and Deserialization in C#
- Summary

Canberra Institute of Technology

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

What is Serialization?

- Serialization is the process of transforming a complex data structure in code into a format that can be saved to a file or sent across a stream (such as a network)

- Once in a serialized format, the object can be deserialized to create a clone of the original object

Serialization

- One of the main uses of serialization is to save and load data such as save game information, game level data, and options data such as what resolution the game is running at, key bindings, etc.

File I/O or Serialization?

- If you're using file text I/O libraries to save and load game data and then using the data stored in the file to reconstruct the object – you are already using serialization – just with your own code instead!

- C# provides us with some handy built in libraries that allow us to serialize our objects extremely easily

Serialization Formats

- C# allows us to serialize to different formats, the two main ones are:
  - XML Serialization
  - Binary Serialization

- XML is the format we will focus on this lesson

XML Serialization

- XML serialization transforms your game objects into XML files which store data.

- XML files are human readable text files so they are easy to edit if we need to change any values

XML Serialization

- Serializing an object is pretty simple:

```
//Class must be public to allow serialization
public class Player
{
    //Only public fields get serialized
    public int health;
    public string name;
    protected bool isAlive;

    public Player()
    {
    }

    public Player(int health, string name, bool isAlive)
    {
        Console.WriteLine("Constructing");
        this.health = health;
        this.name = name;
        this.isAlive = isAlive;
    }

    public void Serialize()
    {
        XmlSerializer mySerializer = new XmlSerializer(typeof(Player));

        StreamWriter streamWriter = new StreamWriter(name + ".xml");

        mySerializer.Serialize(streamWriter, this);

        streamWriter.Close();
    }
}
```

```
static void Main(string[] args)
{
    Player p = new Player(100, "Player 1", true);

    p.Serialize();

    Console.ReadLine();
}
```

Canberra Institute of Technology

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

XML Serialization

- And the resulting .xml file:

```
 <?xml version="1.0" encoding="utf-8"?>
<Player xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <health>100</health>
  <name>Player 1</name>
</Player>
```

Canberra Institute of Technology

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

XML Serialization

- To serialize an object, a few things criteria needs to be met:
  - The class itself needs to be public
  - Public methods are the only thing that get serialized
  - The class needs a default (parameter-less) constructor

XML Deserialization

- XML Deserialization is the process of converting the xml file into a clone of the original object
- Because only public fields are saved out you will need to calculate or provide default values for private members
- Apart from that the process is almost identical to serialization

XML Deserialization

```
public static Player Deserialize(string name)
{
    XmlSerializer mySerializer = new
XmlSerializer(typeof(Player));

    StreamReader streamReader = new StreamReader(name + ".xml");

    Player p = mySerializer.Deserialize(streamReader) as Player;

    streamReader.Close();

    return p;
}
```

```
static void Main(string[] args)
{
    Player p = new Player(100, "Player 1", true);

    p.Serialize();

    Console.ReadLine();
}
```

Canberra Institute of Technology

aie
SPECIALIST EDUCATORS IN GAMES, ANIMATION & FILM VFX

Other Serialization Types

- It is also possible to serialize to other formats, not just XML.
  - You can serialize to Binary, SOAP, etc.

- Binary serialization is just like XML except the file is saved as raw byte data.
  - This makes it very hard to edit the binary file which may be preferable if you don't want a user of your application to edit your data by hand

Summary

- Serialization is the process of converting an object into a stream of data that can be used to re-create a clone the object at a later date
- Only public methods can be serialized, so any private data needs to be reconstructed or otherwise given default values

References

- Microsoft, 2014, *XML and SOAP Serialization*
  - https://msdn.microsoft.com/en-us/library/90c86ass(v=vs.110).aspx