Jibran Syed

GAME DATA SERIALIZATION WITH YAML

What Is Serialization?

And why bother?

What is Serialization?

- The process of converting run-time data into a format that is fit for long term storage
  - "Long term" here implies outside the scope of the program
- Storage can be either:
  - Memory buffer (within scope of program)
  - File
  - Over the network

What is Serialization?

- Deserialization is the opposite
  - Taking stored data and converting it into run-time data
- Data can either be any type, including objects
- Format of stored data is dependent on its use in the application
  - Not the same for every app

What is Serialization?

Some Synonyms:

Serialization	Deserialization
Marshalling	Unmarshalling
Deflating	Inflating
Saving	Loading

Why Serialization?

- Different reasons to serialize data in game development:
  - User data generated by the game that needs to be retained
    - e.g.: Save files
  - Game data that developers need to make the game
    - e.g.: Scene data or shader parameters
  - Networked games need to send data to each other
    - e.g.: Message packets

Serialization Formats

Text vs. Binary

Serialization Formats

- Text-based formats:
  - Human readable
  - Human editable
  - But slower to parse
- Binary formats:
  - Only app readable
  - Only app editable
  - But very fast to parse!

Serialization Formats

- Text-based
  - XML
  - JSON
  - YAML
- Binary-based
  - Google Protocol Buffers
  - Apache Thrift
  - Or do it yourself ☺

Text-based Serialization

- XML
  - Most advanced (most features and toolsets)
  - Looks similar to HTML
  - Generally slowest to deserialize
    - (Depends on parser implementation)
  - Used in config files and networked messages

Text-based Serialization

- JSON
  - Least features
    - Meant purely as data interchange
    - No comments!
  - Looks similar to JavaScript
  - Generally fastest to deserialize
  - Mostly used in networked messages
    - Not recommended for config files

Text-based Serialization

- YAML
  - Easiest to read and write by a human
  - Looks a little similar to Python (not exactly)
  - Performance generally between XML's & JSON's
    - Not an issue if only deserializing at app's start
  - Great for config files
    - Not recommended for networked messages

Text-based Serialization

- Comparison of syntaxes

Original C++ Layout

```
struct Person
{
    string name;
    uint age;
    struct {
        string type;
        string number;
    } phoneNumber;
};
```

XML

```
<?xml version="1.0"?>
<person>
    <name>John Smith</name>
    <age>35</age>
    <phoneNumber>
        <type>Mobile</type>
        <number>(555)-555-5555</number>
    </phoneNumber>
</person>
```

JSON

```
{
   "name": "John Smith",
   "age": "35",
   "phoneNumber": {
       "type": "Mobile",
       "number": "(555)-555-5555"
   }
}
```

YAML

```
---
person:
    name: John Smith
    age: 35
    phoneNumber:
        type: Mobile
        number: (555)-555-5555
...
```

About YAML

Crash Course

About YAML

- 3 types of structuring:
  - Scalars
    - Basically any individual piece of data
    - String, int, float, etc.
  - Sequence
    - A list of scalars
    - Can also list sequences or maps!
  - Map
    - A key scalar paired with a value
    - The “value” can be a scalar, sequence, or map

About YAML

- Scalars
  - Ints
  - Floats
  - Booleans
  - Strings

```
27
0x54BEF
```

```
3.14
2.85e+02
```

```
true
yes
on
false
no
off
```

```
Plain Text Without Quotes
'With Single Quotes'
"With Double Quotes"
```

About YAML

- Sequences
  - In Block form

```
- Apples
- Oranges
- Bananas
- Limes
```

```
- 534.23
- 83.3532
- 9.176
- 334.2093
```

  - In Flow form

```
[Apples, Oranges, Bananas, Limes]
```

```
[534.23, 83.3532, 9.176, 334.2093]
```

About YAML

- Maps

```
Key: Value
```

  - In Block form

```
Name: Bob
Age: 31
Language: C++
```

  - In Flow form

```
{Name: Bob, Age: 31, Language: C++}
```

About YAML

- Comments

```yaml
# Whole line comment
Type: Combatant
Health: 9083
Lives: 1   # At end of line
```

- Nesting
  - Indents with spaces, not tabs!

```yaml
Front-Cannons:
    Ammo: 3047
Side-Cannons:
    Ammo: 5812
```

About YAML

- Three hyphens (---) indicates beginning of YAML document
- Three periods (...) indicates end of YAML stream
  - Optional
  - A "stream" is a sequence of multiple YAML documents

About YAML

- Example YAML file
  - 1 Stream
  - Multiple YAML documents in 1 file

```
---
Name: Sue
Job: Developer
---
Name: Tom
Job: Producer
---
Name: Joe
Job: Artist
---
Name: Liz
Job: Sound Designer
...
```

Demo

- Using **yaml-cpp** library to parse YAML files
  - https://github.com/jbeder/yaml-cpp
- Deserialize multiple cameras from 1 file
- Deserialize 1 Material per file

References

- YAML Reference Card
  - http://www.yaml.org/refcard.html
- YAML Basics and Parsing with yaml-cpp
  - http://www.gamedev.net/page/resources/_/technical/apis-and-tools/yaml-basics-and-parsing-with-yaml-cpp-r3508
- yaml-cpp Tutorial
  - https://github.com/jbeder/yaml-cpp/wiki/Tutorial
- http://www.yamllint.com/