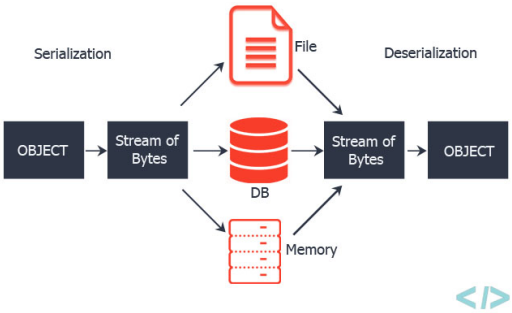
# Lab 1: Introduction to Serialization

Neil Johari

May 14, 2019

## 1 Introduction




pc: codenuclear.com

Serialization (also known as marshalling) is a powerful data storage technique employed to deal with data that is intended to transmit or store the state of objects in a computer program.

During serialization, data structures and objects are converted into a stream of bytes, which then can be stored on a disk or shared over a network. Serialization becomes useful when data must be reconstructed via deserialization (or unmarshalling): the serialized stream is used to recreate the original object in memory, and the new object is semantically equivalent to when it was serialized.

In this lab we will explore a brief history of various serialization techniques and understand how computer files work. Then we will investigate how we can implement these techniques into microcontroller driven systems. In the following labs, we will actually implement two examples of serialization in ArduinoC and corresponding deserialization.

## 2 History

In the 1990s, Extensible Markup Language (XML) was developed by the W3C as a file format intended to structure data documents such that they were both easily readable by humans and also parsable in computer programs. Please note that XML is not the same thing as HTML; HTML is used to describe how a page should be presented in browsers, while XML describes content.

Later, Javascript Object Notation (JSON) and YAML were pushed as light-weight alternatives to XML. JSON is often used to send data back and forth between web clients and servers, while YAML has become popular for configuration files. Since YAML 1.2, YAML is fully compatible with JSON as an official subset. [1].

Listing 1: Sample XML document [2])

```
<food>
    <name>Belgian Waffles</name>
    <price>$5.95</price>
    <description>
           Two of our famous Belgian Waffles with plenty of real
               maple syrup
    </description>
    <calories>650</calories>
</food>
```

Listing 2: Sample YAML document [3]

```
json:
  - rigid
  - better for data interchange
yaml:
  - slim and flexible
  - better for configuration
object:
   key: value
  array:
    - null_value:
    - boolean: true
```

```
    - integer: 1
```

Listing 3: Sample JSON document [3]

```
json:
  - rigid
  - better for data interchange
yaml:
  - slim and flexible
  - better for configuration
object:
   key: value
  array:
    - null_value:
    - boolean: true
    - integer: 1
```

## 2.1 Streams

## 2.2 Computer Files

# 3 Serialization in Microcontrollers

## 3.1 Constraints

## 3.2 Libraries

### 3.2.1 ArduinoJson

### 3.2.2 Nanopb

# References


[1] O. Ben-Kiki, C. Evans, and I. d. Net, *YAML Ain't Markup Language (YAML™) Version 1.2*, Jan 2009. [Online]. Available: https://yaml.org/spec/1.2/spec.html

[2] "Xml tutorial." [Online]. Available: https://www.w3schools.com/xml/

[3] “Yaml to json.” [Online]. Available: https://www.json2yaml.com/convert-yaml-to-json