Kenny R N Wilson

Data Encoding

In memory data structures live in objects, collections, arrays etc. In order to send data over a network or persist it in a database the in-memory data structures must be converted to byte streams. We call such conversion encoding or serialization. The reverse process is called decoding or deserialization. There are several encoding formats. In this document we look at JSON and Protocol Buffers.

Json

Source Code

The JSON format is a text-based format. Look at the following amazingly simple piece of JSON as it appears in Visual Studio.

```
{
    "SomeLongIntegerFieldName" : 1234567888
}
```

If I look at the same piece of JSON inside Notepad++ with a hex extension added I see the following. Note how the number 1234567888 is stored. I have highlighted it in bold. It takes 10 ascii characters or 10 bytes to hold it. Also note we need to send the whole field name SomeLongIntegerFieldName because we do not have any schema.

7b 0d 0a 20 20 20 20 22 53 6f 6d 65 4c 6f 6e 67 49 6e 74 65 67 65 72 46 69 65 6c 64 4e 61 6d 65 22 20 3a 20 **31 32 33 34 35 36 37 38 38 38** 0d 0a 7d

Clearly this is an inefficient protocol for the following reasons.

- All field name much be send with the corresponding field values.
- Numbers are sent as strings which often required more bytes.
- Representing numbers as strings requires decoding and encoding at each end.

Protocol Buffers