Getting Started with JSON in C# Using Json.NET

The Case for JSON and Json.NET

Xavier Morera

@xmorera | www.xaviermorera.com

The Case for JSON and Json.NET

Objectives – After This Course You Will

- Understand JSON
- Serialize & Deserialize
- Settings & Attributes
- LINQ To JSON
- Performance & BSON
- XML & Json.NET Schema

Where to Go?

- M1: Case for JSON and Json.NET
- M2: Serialization Fundamentals
- M3: Settings & Attributes
- M4: Custom Serialization
- M5: Performance Tips
- M6: LINQ to JSON
- M7: JSON & XML
- M8: Binary JSON (BSON)
- M9: Json.NET Schema
- Wrap Up

What Is JSON?

What Is JSON?

JavaScript Object Notation

Data Interchange Format

Human Readable Text

Open Standard

Data Objects as

Key Value Pairs

Hello JSON!

```
{
  "name": "Xavier Morera",
  "courses": [
        "Solr",
        "dotTrace"
        ],
  "since": "2014-01-14T00:00:00",
  "happy": true,
  "issues": null,
  "car": {
         "model": "Land Rover Series III",
         "year": 1976
        }
}
```

History of JSON

- By Douglas Crockford (2001)
  - http://www.Json.org
- Described by two competing standards

RFC 7159

- Specification
- Semantic
- Security considerations

ECMA-404

- Minimal specification
- Only allowed grammar syntax

Douglas Crockford

Computer programmer

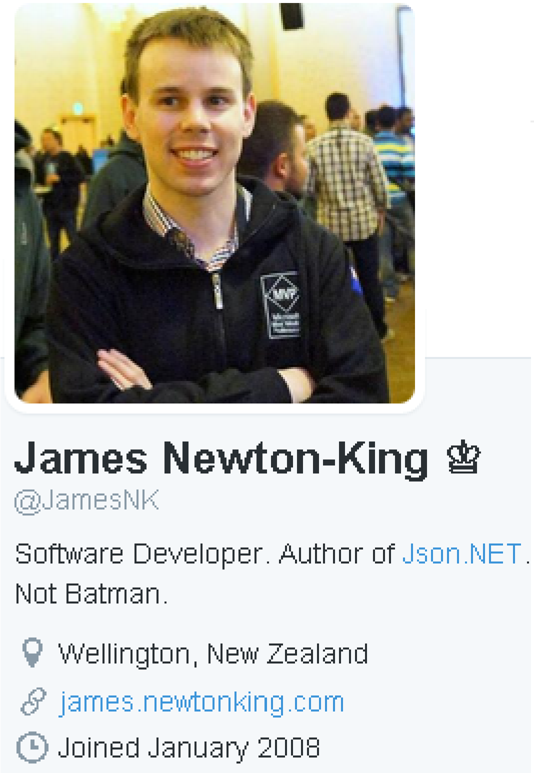
Douglas Crockford is an American computer programmer and entrepreneur who is best known for his ongoing involvement in the development of the JavaScript language, for having popularized the data format ... Wikipedia

Born: Minnesota

Known for: JSON

Education: San Francisco State University

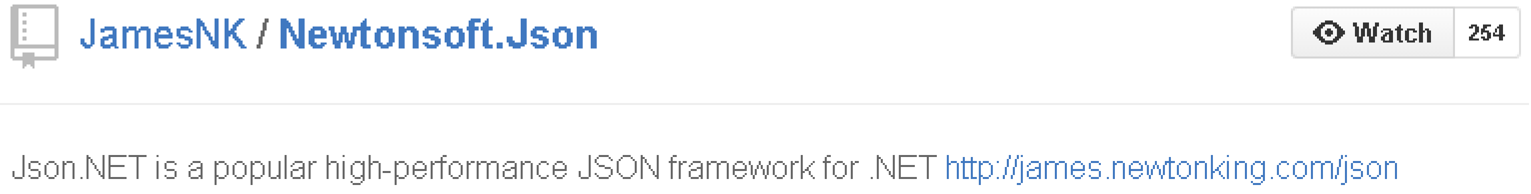
History of Json.NET



James Newton-King ♔
@JamesNK

Software Developer. Author of Json.NET. Not Batman.

Wellington, New Zealand
james.newtonking.com
Joined January 2008



JamesNK / Newtonsoft.Json

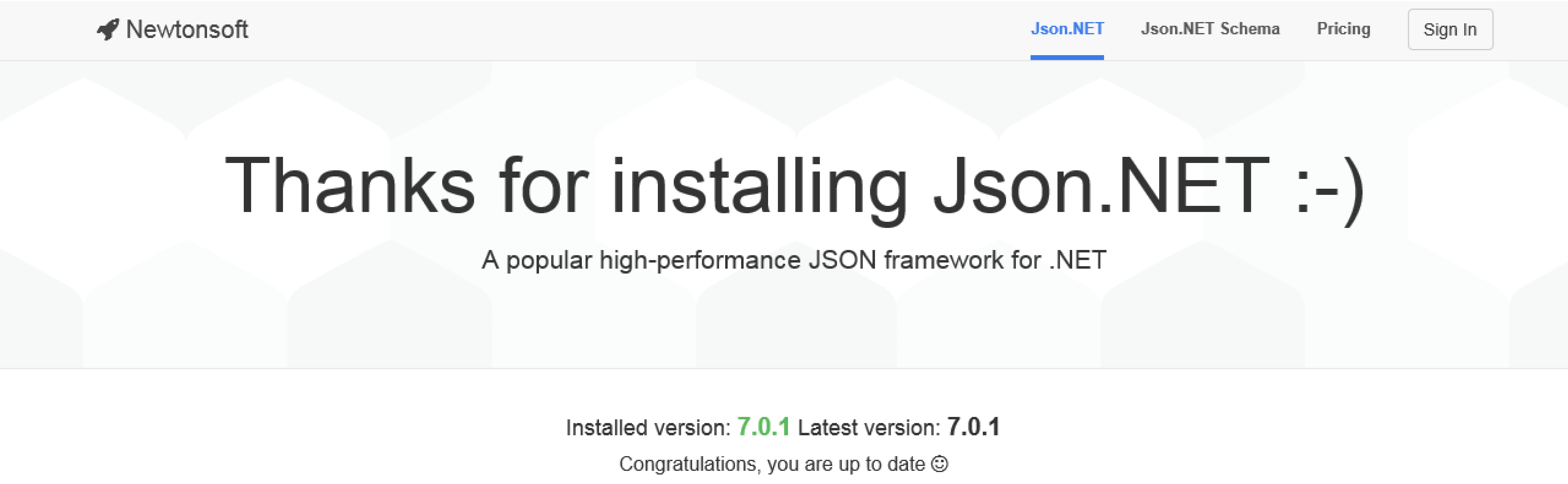
Json.NET is a popular high-performance JSON framework for .NET http://james.newtonking.com/json

Json.NET

Popular high-performance JSON framework for .NET

Version 7.0.1 -

Getting Started Is Easy



Thanks for installing Json.NET :-)

A popular high-performance JSON framework for .NET

Installed version: 7.0.1 Latest version: 7.0.1

Congratulations, you are up to date ☺

Demo 1: JSON in the Real World

Web sites

Mobile applications

REST APIs

Databases

And much more

Why Is JSON Important?

Supported Natively in JavaScript

```
var authors = [{
    firstName: 'Xavier',
    lastName: 'Morera',
    courseCount: 4
}, {
    firstName: 'Peter',
    lastName: 'Shaw',
    courseCount: 1
}, {
    firstName: 'Jason',
    lastName: 'Alba',
    courseCount: 16
}];
```

```
<script type="text/javascript">
var theAuthor = document.getElementById("oneAuthor");
var theAuthors = document.getElementById("allAuthors");
var countAuthors = document.getElementById("totalAuthors");
theAuthor.innerHTML = authors[0].firstName + " " + authors[0].lastName;
theAuthors.innerHTML = authors[0].firstName + ", " +authors[1].lastName;
countAuthors.innerHTML = "Total authors: " + authors.length;
</script>
```

Developers Want Out of the Syntax Business

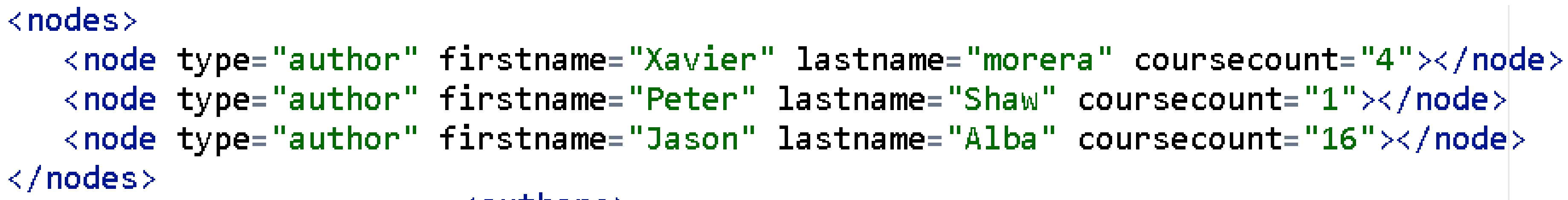
```xml
<author first-name="Xavier" last-name="Morera"/>

<author>
   <first-name>Xavier</first-name>
   <last-name>Morera</last-name>
</author>

<object type="author">
   <property name="first-name">Xavier</property>
   <property name="last-name">Morera</property>
</object>
```

Fat Free Alternative to XML

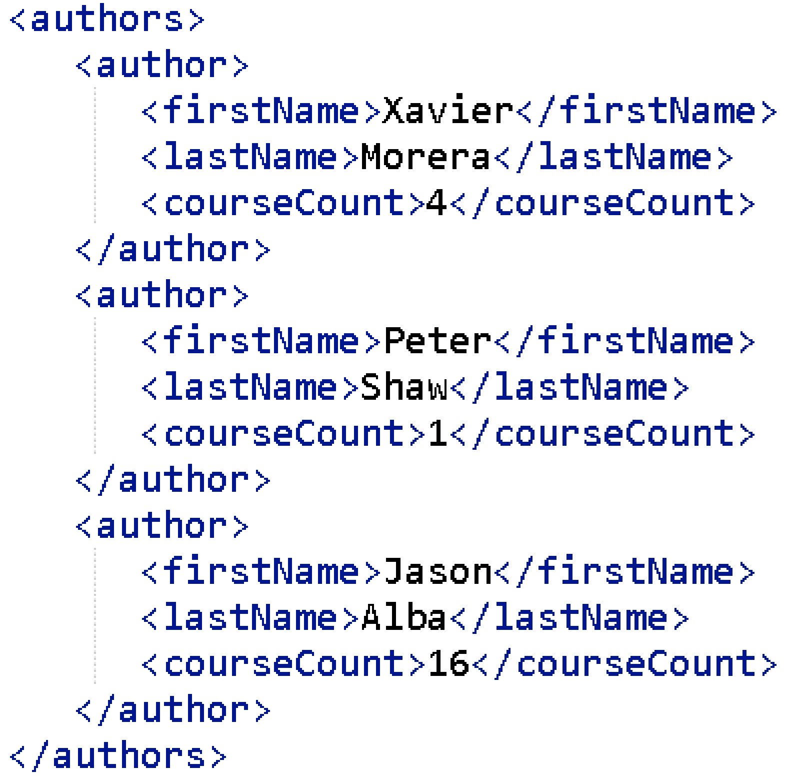
```json
{
    "authors": {
        "author": [
            {
                "firstName": "Xavier",
                "lastName": "Morera",
                "courseCount": "4"
            },
            {
                "firstName": "Peter",
                "lastName": "Shaw",
                "courseCount": "1"
            },
            {
                "firstName": "Jason",
                "lastName": "Alba",
                "courseCount": "16"
            }
        ]
    }
}
```

```xml
<authors>
   <author>
      <firstName>Xavier</firstName>
      <lastName>Morera</lastName>
      <courseCount>4</courseCount>
   </author>
   <author>
      <firstName>Peter</firstName>
      <lastName>Shaw</lastName>
      <courseCount>1</courseCount>
   </author>
   <author>
      <firstName>Jason</firstName>
      <lastName>Alba</lastName>
      <courseCount>16</courseCount>
   </author>
</authors>
```

Fat Free Alternative to XML



```xml
<nodes>
   <node type="author" firstname="Xavier" lastname="morera" coursecount="4"></node>
   <node type="author" firstname="Peter" lastname="Shaw" coursecount="1"></node>
   <node type="author" firstname="Jason" lastname="Alba" coursecount="16"></node>
</nodes>
```



```xml
<authors>
   <author>
      <firstName>Xavier</firstName>
      <lastName>Morera</lastName>
      <courseCount>4</courseCount>
   </author>
   <author>
      <firstName>Peter</firstName>
      <lastName>Shaw</lastName>
      <courseCount>1</courseCount>
   </author>
   <author>
      <firstName>Jason</firstName>
      <lastName>Alba</lastName>
      <courseCount>16</courseCount>
   </author>
</authors>
```

Demo 2: JSON in JavaScript

Blends Right In

JSON Specification

- Collection of name value pairs
- Four primitive types:
  - Strings
  - Numbers
  - Booleans
  - Null
- Two structured types:
  - Objects
  - Arrays

Duplicate Keys

ECMA-404

- No mention of unique keys

```
{
 “author”: “Xavier”,
 “author”: “Xavier”
}
```

RFC 7159

- “The names within an object SHOULD be unique”
- Parse string with Douglas Crockford’s original implementation fails
- Json.NET fails with an exception
- Dictionary

JSON Data Types

Number

```
"courses": 5,
"weight": 205.5,
"likeJSONOverXML": 7.02e+23,
```

String

```
"fullName": "Xavier Morera",
```

Boolean

```
"developer": true,
```

Array

```
"counts": [
  "one",
  2,
  "three",
  4
]
```

Object

```
"author": {
  "firstName": "Xavier",
  "lastName": "Morera"
}
```

null

```
"healthyDiet": null,
```

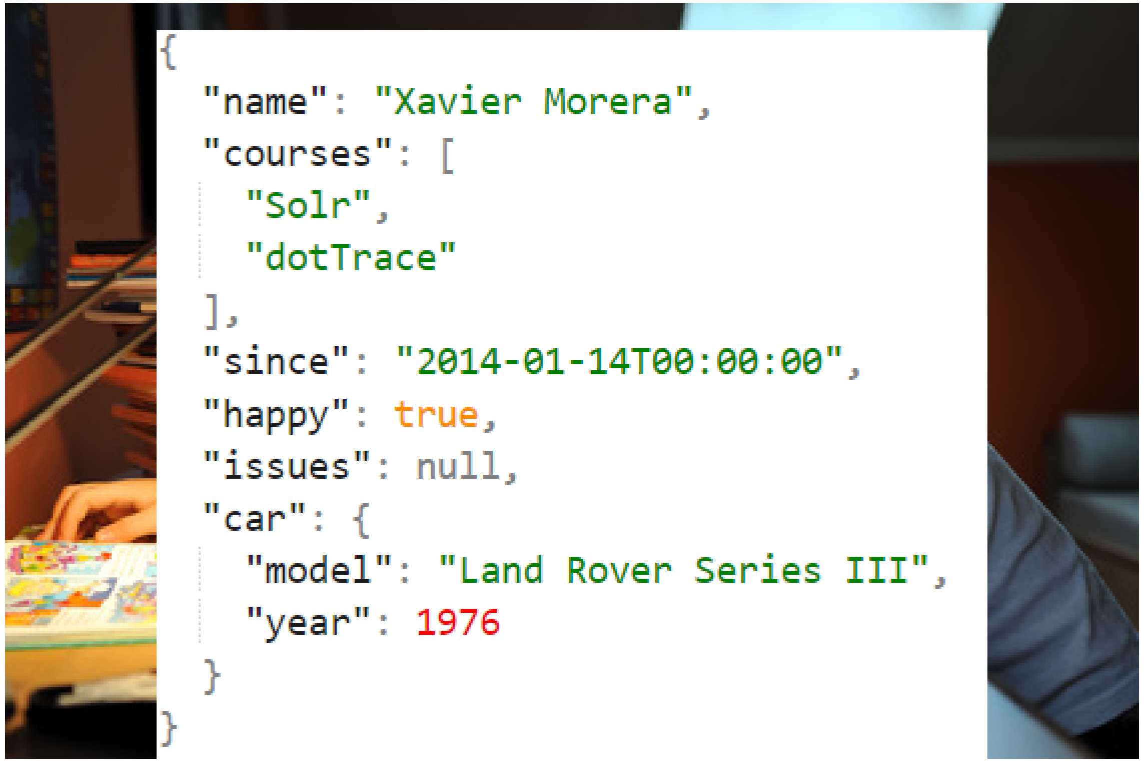
Whitespaces : `"spaces"    :      "ignore them outside of key value pairs"`

```json
{
  "fullName": "Xavier Morera",
  "developer": true,
  "courses": 5,
  "weight": 205.5,
  "healthyDiet": null,
  "likeJSONOverXML": 7.02e+23,
  "author": {
    "firstName": "Xavier",
    "lastName": "Morera"
  },
  "counts": [
    "one",
    2,
    "three",
    4
  ],
  "spaces": "ignore them outside of key value pairs"  airs"
}
```

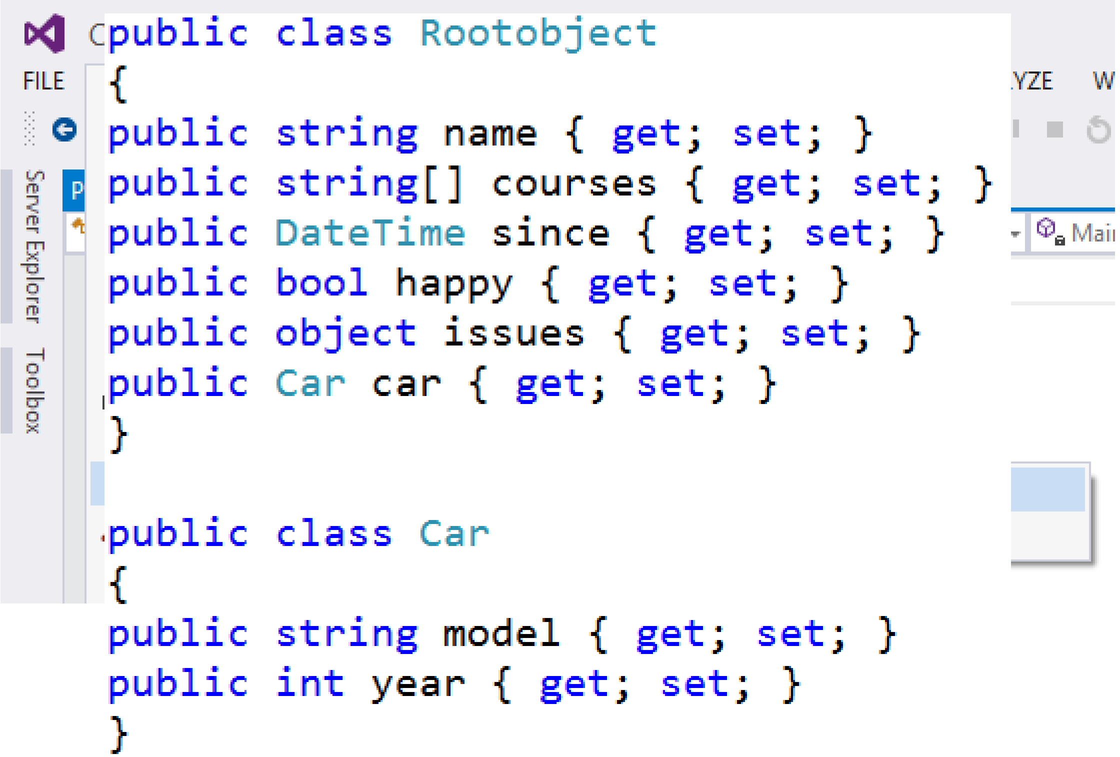
Tip: Paste as JSON in Visual Studio

- Tired of writing your classes?

Ctrl + C is your friend



```
{
  "name": "Xavier Morera",
  "courses": [
    "Solr",
    "dotTrace"
  ],
  "since": "2014-01-14T00:00:00",
  "happy": true,
  "issues": null,
  "car": {
    "model": "Land Rover Series III",
    "year": 1976
  }
}
```



```
public class Rootobject
{
public string name { get; set; }
public string[] courses { get; set; }
public DateTime since { get; set; }
public bool happy { get; set; }
public object issues { get; set; }
public Car car { get; set; }
}

public class Car
{
public string model { get; set; }
public int year { get; set; }
}
```

Takeaway

JavaScript Object Notation

(JSON)

Human Readable

Data Interchange Format

Key/Value Pairs

RFC 1759

ECMA 404

Developers Out Of Syntax Business!

Fat Free Alternative to XML

Json.NET