Kyle Rainville
Littleton Coin Company

What is JSON?

- Javascript Object Notation (a subset of)
- Data Interchange Format
- Provides a way for communication between platforms & languages
- Derived from Javascript
- Key-value pairs, usually rendered in curly braces
- Language independent

History

- “Discovered” by Douglas Crockford (State Software) – 2001
- Popularized its use
- Named it as he considered it part of the Javascript language
- Later discovered that JSON was in use by Netscape as early as 1996
- Needed browser-to-server communication without Java applets or Flash plugins – provided the impetus
- Established as a standalone ECMA standard in 2013

Analogy

- Islands that desire to communicate

Popular uses

- Web service APIs make data available to third-party apps
- AJAX communication uses JSON to transmit data between browser-server
  - Client-side manipulation of JSON required, usually with Javascript
  - If server needs to perform actions with JSON, server-side programming language will need to handle it

Data Structures

- Essentially key-value pairs
- Formatted into two data structures
  - JSON object – collection of name-value pairs
  - JSON array – list of objects or values
- Straightforward (normally)
- Complexity begins – values can contain numbers, Booleans, strings, nulls, and even nested arrays/objects
- Keys can only be strings

Example

```
{
   “first_name” : “Sammy”,
   “last_name” : “Shark”,
   “location” : “Ocean”,
   “online” : true,
   “followers” : 987
}
```

JSON Syntax

- Have a colon between key and value
- Every key-value pair separated by a comma
- In the previous example, the first key-value pair is “first_name” : “Sammy”.
- Keys → left, values → right
- Keys needs to be wrapped in double-quotes
- Each ‘key’ must be unique per object
- Key strings *can include* whitespace, though *may not be* best practice (underscores preferable)

JSON Values

- Values are to the right of the colon
- Required to be one of six simple types
  - Strings
  - Numbers
  - Objects
  - Arrays
  - Booleans
  - Null
- Each type passed will maintain their syntax – e.g. strings will be in quotes but numbers won’t

JSON vs. XML

- XML has served a purpose similar to JSON's
- JSON requires much less formatting
  - More readable (for humans), as a result
- JSON is more lightweight
- JSON now a common replacement for XML within AJAX requests

Validating JSON

- Quick check - use JSONLint.com

```
{
   "key": "value",
   "veracity": false,
   "num": 6
}
```

- Is this valid JSON?

```
{
   'color': "Red",
   'material': "Blood",
   'worst_doctor': "Harold Shipman",
}
```

- How about this?

- Can also use JSON Schema

.json files

- JSON can exist as an external file
- Generally, the format will be extended across multiple lines
- Not a requirement, JSON can be one line

```
{ "first_name" : "Sammy", "last_name": "Shark",  "online" : true, }
```

- Multiple lines more readable, especially with large data sets
- Whitespace between elements ignored – make as readable as possible

```
{
  "first_name"  :  "Sammy",
  "last_name"   :  "Shark",
  "online"      :  true
}
```

Javascript Object and JSON

- JSON *is not* the same format as a JavaScript object, despite that JSON is often considered a subset
- There are small differences between them
  - Javascript's string rules are different, e.g. line-terminators not allowed
  - Javascript object rules are also different, e.g.:
    - Can include functions as member of object
    - Property key can be non-quoted or single-quoted

Complex JSON

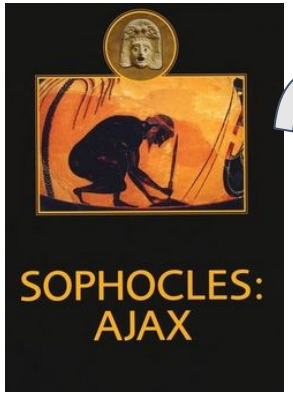
- JSON can become more complex than simple key-value pairs
- JSON can also contain nested objects and nested arrays
- Though these may appear to be more complex, they are still assigned as values in the key-value paradigm

Complex object – nested arrays/objects

-JSON object containing an array of objects
-Square bracket → array
-Curly brace → object
-Though the key of ‘contentManagementSystems’ contains
a nested array, it still represents a name-value pair


```
{
 "contentManagementSystems" : [
  {
  "name": "WordPress",
  "percentMarketShare": 58.9
  },
  {
  "name": "Joomla",
  "percentMarketShare": 6.1
  },
  {
  "name": "Drupal",
  "percentMarketShare": 4.9
  }
 ]
}
```

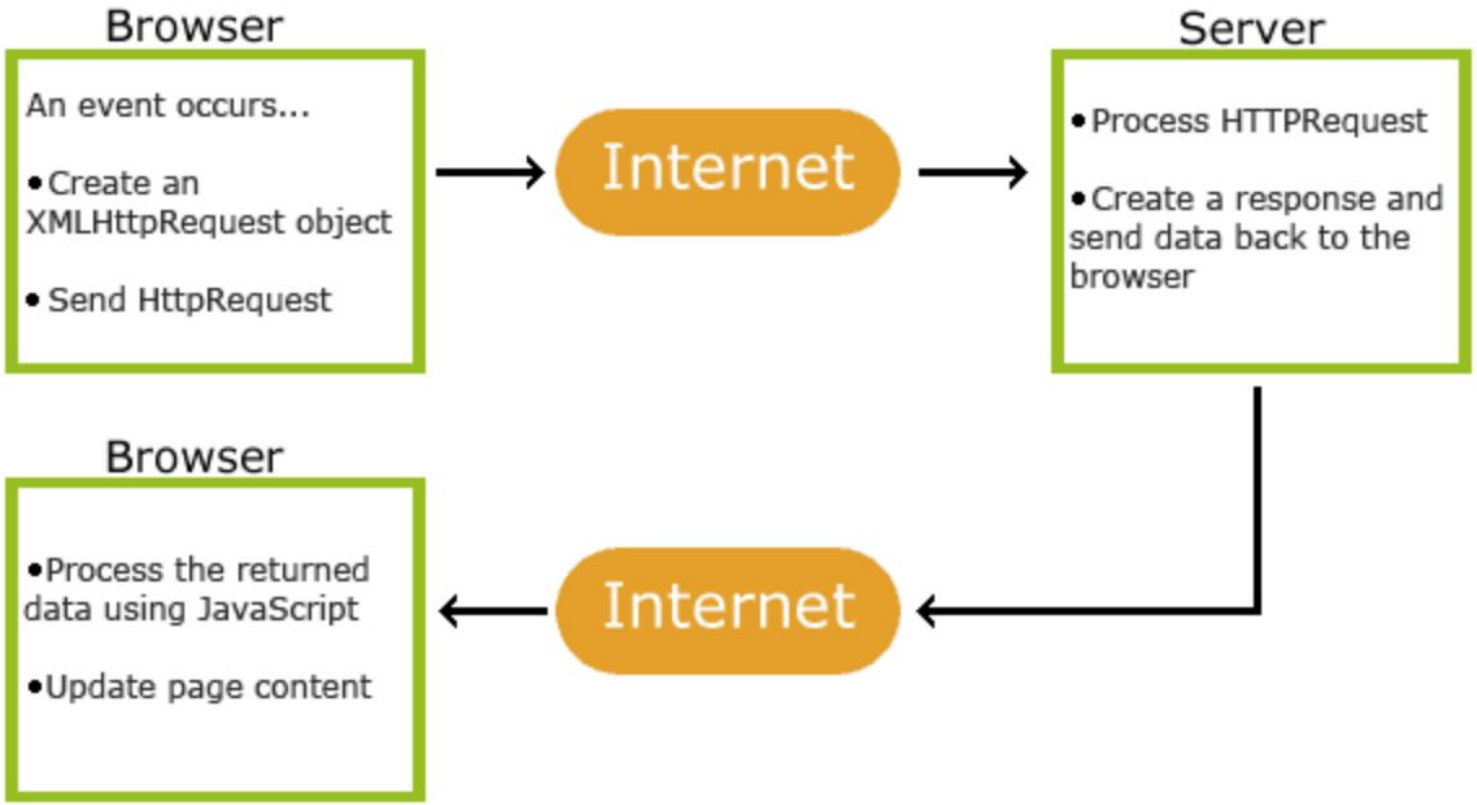
AJAX



- AJAX - Asynchronous JavaScript and XML
- Was part of the Web 2.0 revolution
- Allows for asynchronous communication between web-page/browser and server
- e.g. Google's 'Google Instant' recommended searches

How AJAX Works

- jQuery can provide abstractions to make life easier



JSON Schema

- Allows you to annotate and validate JSON
- Still in draft form (not yet adopted by IETF)
- Describes your existing data format
- clear, human- and machine-readable documentation
- complete structural validation, useful for
  - automated testing
  - validating client-submitted data
- http://json-schema.org/

JSON Schema Example

```
{
    "title": "Person",
    "type": "object",
    "properties": {
        "firstName": {
            "type": "string"
        },
        "lastName": {
            "type": "string"
        },
        "age": {
            "description": "Age in years",
            "type": "integer",
            "minimum": 0
        }
    },
    "required": ["firstName", "lastName"]
}
```

Demo - Album comments

- With ajax
  - http://localhost/example.php
- Without ajax - form submission
  - http://localhost/example-form.php

Demo - OpenWeather API

http://localhost/weather2.html

Cross Origin Resource Sharing (CORS)

- Normally, cross-domain AJAX requests are forbidden by browsers' same-origin policy
- How did the above work?
- CORS - header sent from the server
- There are other ways to bypass same-origin policy
  - JSONP
  - WebSockets

JSON as a Configuration File

- Becoming popular as config data
  - Wide support, ease of parsing, and human readability
- Config files used often for software - allowing changes without recompiling
- Several popular formats - e.g. INI, XML
- Example use: node.js

Configuration Example Comparison

```
[general]
playIntro=false
mouseSensitivity=0.54

[display]
complexTextures=true
brightness=4.2
widgetsPerFrame=326
mode=windowed

[sound]
volume=1
effects=0.68
```

```
<?xml version="1.0" encoding="UTF-8" ?>
<settings>
    <general>
        <playIntro>false</playIntro>
        <mouseSensitivity>0.54</mouseSensitivity>
    </general>
    <display>
        <complexTextures>true</complexTextures>
        <brightness>4.2</brightness>
        <widgetsPerFrame>326</widgetsPerFrame>
    </display>
    <sound>
        <volume>1</volume>
        <effects>0.68</effects>
    </sound>
</settings>
```

```
{
    "general": {
        "playIntro": false,
        "mouseSensitivity": 0.54
    },
    "display": {
        "complexTextures": true,
        "brightness": 4.2,
        "widgetsPerFrame": 326,
        "mode": "windowed"
    },
    "sound": {
        "volume": 1,
        "effects": 0.68
```

Pros & Cons to Different Config Types

- INI
  - Very human readable
  - Not good at handling complex or nested data
- XML
  - Less human-readable
  - Proficient at handling complex data
- JSON
  - Human readable (not as human readable as INI)
  - Good at handling complex data (not as many options as XML, but more options than INI)

Conclusion

- JSON is popular for data interchange as many systems are modeling data as objects
- Works great within Javascript and its popularity is a result of that
- Excellent at communicating with APIs
- As with any tool - has advantages and disadvantages
- Boils down to - what is the right tool for the job?

References

- https://www.digitalocean.com/community/tutorials/an-introduction-to-json
- https://www.w3schools.com/js/js_json_intro.asp
- http://json-schema.org/
- https://www.copterlabs.com/json-what-it-is-how-it-works-how-to-use-it/
- https://www.w3schools.com/js/js_json_syntax.asp
- https://openweathermap.org/
- Bassett, Lindsay. *Introduction to Javascript Object Notation: a to-the-Point Guide to JSON*. O'Reilly, 2015.