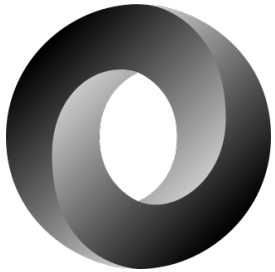
JSON



The Fat-Free Alternative to XML

```
{
  “Lecture”: 27,
  “Course”: “CSC375”,
  “Days”: ”TTh",
  “Instructor”: “Haidar Harmanani”
}
```

JSON as an XML Alternative

- JSON is a light-weight alternative to XML for data-interchange
- JSON = JavaScript Object Notation
  - It’s really language independent
  - most programming languages can easily read it and instantiate objects or some other data structure
- Defined in RFC 4627
- Started gaining tracking ~2006 and now widely used
- http://json.org/ has more information

JSON as an XML Alternative

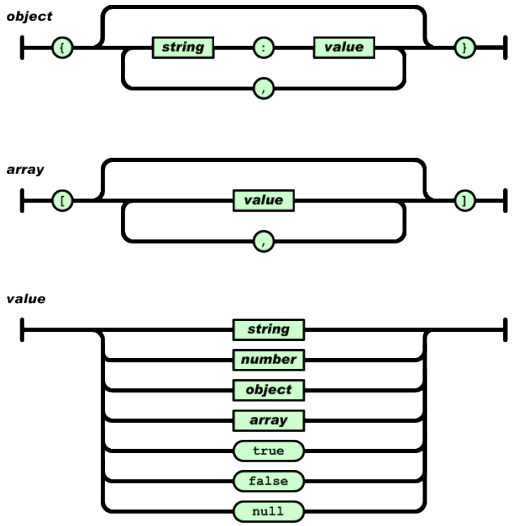
- **What is JSON?**
  - JSON is language independent
  - JSON is "self-describing" and easy to understand
  - *JSON uses JavaScript syntax for describing data objects, but JSON is still language and platform independent. JSON parsers and JSON libraries exists for many different programming languages.
- **JSON - Evaluates to JavaScript Objects**
  - The JSON text format is syntactically identical to the code for creating JavaScript objects.
  - Because of this similarity, instead of using a parser, a JavaScript program can use the built-in eval() function and execute JSON data to produce native JavaScript objects.

Example

```
{"firstName": "John",
 "lastName" : "Smith",
 "age"         : 25,
 "address"   :
   {"streetAdr” : "21 2nd Street",
    "city"        : "New York",
    "state"       : "NY",
    ”zip"         : "10021"},
 "phoneNumber":
   [{"type"  : "home",
    "number": "212 555-1234"},
   {"type"  : "fax",
    "number” : "646 555-4567"}]
 }
```

- This is a JSON object with five key-value pairs
- Objects are wrapped by curly braces
- There are no object IDs
- Keys are strings
- Values are numbers, strings, objects or arrays
- Aarrays are wrapped by square brackets

The BNF is simple



When to use JSON?

- SOAP is a protocol specification for exchanging structured information in the implementation of Web Services.
- SOAP internally uses XML to send data back and forth.
- REST is a design concept.
- You are not limited to picking XML to represent data, you could pick anything really (JSON included).

JSON example

```
{
	"firstName": "John",
	"lastName": "Smith",
	"age": 25,
	"address": {
		"streetAddress": "21 2nd Street",
		"city": "New York",
		"state": "NY",
		"postalCode": 10021
	},
	"phoneNumbers": [
		{
			"type": "home",
			"number": "212 555-1234"
		},
		{
			"type": "fax",
			"number": "646 555-4567"
		}
	]
}
```

JSON to XML

```
<?xml version="1.0" encoding="UTF-8"?>
<persons>
	<person>
		<firstName>John</firstName>
		<lastName>Smith</lastName>
		<age>25</age>
		<address>
			<streetAddress>21 2nd Street</streetAddress>
			<city>New York</city>
			<state>NY</state>
			<postalCode>10021</postalCode>
		</address>
		<phoneNumbers>
			<phoneNumber>
				<number>212 555-1234</number>
				<type>home</type>
			</phoneNumber>
			<phoneNumber>
				<number>646 555-4567</number>
				<type>fax</type>
			</phoneNumber>
		</phoneNumbers>
	</person>
</persons>
```

JSON vs XML size

- XML: 549 characters, 549 bytes
- JSON: 326 characters, 326 bytes

- XML ~68,4 % larger than JSON!

- But large data set is going to be large regardless of the data format you use.
- Most servers gzip or otherwise compress content before sending it out, the difference between gzipped JSON and gzipped XML isn't nearly as drastic as the difference between standard JSON and XML.

JSON Values

- JSON values can be:

- A number (integer or floating point)
- A string (in double quotes)
- A boolean (true or false)
- An array (in square brackets)
- An object (in curly brackets)
- null

Why JSON?

- For AJAX applications, JSON is faster and easier than XML:

- **Using XML**

- Fetch an XML document
- Use the XML DOM to loop through the document
- Extract values and store in variables

- **Using JSON**

- Fetch a JSON string
- eval() the JSON string

Security problems

- The eval() function can compile and execute any JavaScript. This represents a potential security problem.
- It is safer to use a JSON parser to convert a JSON text to a JavaScript object. A JSON parser will recognize only JSON text and will not compile scripts.
- In browsers that provide native JSON support, JSON parsers are also faster.
- Native JSON support is included in newer browsers and in the newest ECMAScript (JavaScript) standard.

JSON vs XML

- **Favor XML over JSON when any of these is true:**
- You need message validation
- You're using XSLT
- Your messages include a lot of marked-up text
- You need to interoperate with environments that don't support JSON
- **Favor JSON over XML when all of these are true:**
- Messages don't need to be validated, or validating their deserialization is simple
- You're not transforming messages, or transforming their deserialization is simple
- Your messages are mostly data, not marked-up text
- The messaging endpoints have good JSON tools

JSON Schema

- Describes your JSON data format
- http://jsonschemalint.com/
- http://json-schema.org/implementations
- http://en.wikipedia.org/wiki/JSON#Schema_and_Metadata

JSON vs XML

- "Some people, when confronted with a problem, think "I know, I'll use XML." Now they have two problems. "
- "XML is like violence. If it doesn't solve your problem, you're not using enough of it. "
- "I use JSON unless I'm required to use XML. "
- " I don't hate XML and I don't find it frustrating. I sure as hell hate people that try to solve every problem in the world with XML. "

Evaluation

- JSON is simpler than XML and more compact
  - No closing tags, but if you compress XML and JSON the difference is not so great
  - XML parsing is hard because of its complexity
- JSON has a better fit for OO systems than XML
- JSON is not as extensible as XML
- Preferred for simple data exchange by many
- Less syntax, no semantics
- Schemas? We don’t need no stinkin schemas!
- Transforms? Write your own.
- Worse is better