

Data

ian.mcloughlin@gmit.ie

Sending data via HTTP

Sending data is what HTTP is used for.

HTTP sets out rules for sending data.

Rules allow computers that know very little about each other to communicate without lengthy negotiations.

Sometimes HTTP is not enough. We need more rules.

Building on top of HTTP, we can send complex data — not just HTML, CSS and JavaScript.

Commonly these days we send images, videos, binary files, and even instances of objects.

Why send objects over HTTP?

Databases store a lot of the world's data.

Data are usually stored using some sort of structure, so that they're reusable.

Object usually means blueprint, the plans of the house.

Instance usually means realisation of an object, like an actual house built from the plans.

JavaScript doesn't make the distinction too clear.

Often we would like to send a small chunk of structured data from a database server to a user's machine. Then their browser can display it nicely.

HTTP with JSON can do this.

JSON

JavaScript Programming language.

Object Groups of name–value pairs.

Notation Set of rules for representing objects.

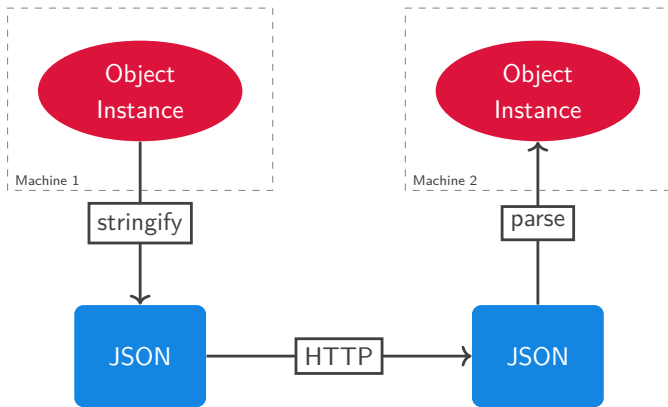
JSON is just text — text that conforms to a syntax.

Influenced by JavaScript's syntax, but it is useable in all languages.

Represents information in text form.

Popular because it is easy to send over HTTP and parse in JavaScript.

Sending JSON



JSON Example

```
{  
  "employees": [  
    {"firstName": "John", "lastName": "Doe"},  
    {"firstName": "Anna", "lastName": "Smith"},  
    {"firstName": "Peter", "lastName": "Jones"}  
  ]  
}
```

Using JSON in JavaScript

The key here is that an object instance on one machine can be turned into JSON text, transmitted, and then reconstructed into an object instance on another machine.

```
// Turning text into a JavaScript object.
```

```
var obj = JSON.parse(text);
```

```
// obj is an object instance.
```

```
// Turning a JavaScript object into text.
```

```
var text = JSON.stringify(obj);
```

```
// text is a string.
```

JSON Syntax

Objects identified by curly braces.

```
{ }
```

Name:Value pairs within objects separated by a comma.

```
{ "name": "Ian", "fingers": 10 }
```

Lists identified by square brackets.

```
[ "Ian", "Marco", "Sam" ]
```

Double quotes around all names and strings.

```
"Ian"
```


JSON Types

- Numbers

`123.456`

- Strings

`"Hello, world!"`

- Boolean

`true`

- Arrays

`[1,2,3]`

- Objects

`{"name": "Ian"}`

- null

`null`

eXtensible Markup Language

XML is a JSON alternative.

eXtensible Designed to accommodate change.

Markup Annotates text.

Language Set of rules for communication.

JSON seems to be used more frequently than XML in most modern web applications.

XML seems to be a little more verbose.

XML was designed with more uses in mind.

HTML and XML look similar.

XML Example

```
<?xml version="1.0" encoding="UTF-8"?>  
<book isbn-13="978-0131774292" isbn-10="0131774298">  
  <title>Expert C Programming: Deep C Secrets</title>  
  <publisher>Prentice Hall</publisher>  
  <author>Peter van der Linden</author>  
</book>
```

Using XML

Tags are not pre-defined tag names – you make them up yourself.

Syntax follows a tree-like pattern. Tags can be nested within other tags.

Document Object Model (DOM) is a concept related to XML.

XML Syntax

Declaration XML documents should have a single line at the start stating that they are XML, the version of XML used, and an encoding.

Elements are the main form of structure in XML, and are enclosed in angle brackets.

Root element XML must have a single root element that wraps all others.

Attributes Elements can have attributes, which are name–value pairs within the angle brackets. A given attribute name can only be specified once per element.

Entity references Certain characters must be escaped with entity references, e.g. `<` for `<`.

Case sensitive Everything in XML is case sensitive.

XML Syntax Example

```
<?xml version="1.0" encoding="UTF-8"?>  
<parent-element attribute-name="attribute-value">  
  <child name="value">Text</child-element>  
  <child name="value">Text</child-element>  
  <child name="value">Text</child-element>  
  <lone-warrior />  
</parent-element>
```

Document Object Model

The DOM is a programming interface for HTML and XML documents.

Models the document as a structured group of nodes that have properties and methods.

Connects web pages to scripts or programming languages.

Use `document.createElement` , `document.createTextNode` and `document.element.appendChild` to add to the DOM.

Use `document.getElementById` to access elements of the DOM.

Extensively used in web applications.

AngularJS, jQuery and React have different ideas about the developers relationship with the DOM.

Using JSON and XML

JSON and XML don't do anything by themselves — they're just text.

HTTP can be used to retrieve JSON (or XML) from a server.

Developers can use JavaScript to retrieve JSON (or XML) in the background of a web application.

The DOM of the web page currently displayed in a browser can then be manipulated to display this information.

Web applications are really just web pages that use this trick.

The mechanism by which a web application does this is called AJAX.

Facebook uses this to display new posts once the user has scrolled to the bottom of the screen.

Asynchronous JavaScript and XML

AJAX stands for Asynchronous JavaScript and XML.

Asynchronous Doesn't block our other JavaScript code.

JavaScript Programming language for the web.

XML eXtensible Markup Language.

Page refreshes happen when the webpage is removed from the view, and a new one is rendered.

JavaScript can be used to change this behaviour. On clicking a link, and AJAX call can be triggered instead, and the web page remains.

Location bar can even be manipulated.

More about AJAX

AJAX allows us to make a HTTP request from JavaScript, receive the response from that request and deal with it.

Despite the name, we don't have to receive XML — we can use JSON or anything else.

Calls happen asynchronously — other code can run while waiting.

HTTP requests are usually relatively slow.

Callback functions are used when HTTP transactions are complete.

JavaScript libraries provide easy-to-use AJAX functions.

Raw AJAX Example

```
var xmlhttp = new XMLHttpRequest();

xmlhttp.onreadystatechange = function() {
    if (xmlhttp.readyState == 4) {
        var mydiv = document.getElementById("mydivid");
        mydiv.innerHTML = xmlhttp.responseText;
    }
};

xmlhttp.open("GET", "https://goo.gl/2GCp1C");
xmlhttp.send();
```

AJAX Example Explained

XMLHttpRequest is a built-in class that provides AJAX functionality in JavaScript.

onreadystatechange should be set to a function to run every time something happens in our HTTP call.

open is called to initialize the request.

send is used to send the request to the server.

readyState changes when the state of the AJAX call changes. This triggers a call to `httpRequest.onreadystatechange`.

Using jQuery

jQuery is much easier to use.

```
<script src="jquery.min.js"></script>
```

```
$.get("https://goo.gl/2GCplC", function(data) {  
    $("#mydivid").html(data);  
});
```