



The JSON Saga

Douglas Crockford
Yahoo! Inc.

I am a heretic.

You have been warned.

I Discovered JSON

- I do not claim to have invented JSON. It already existed in nature.
- I do not claim to have been the first to discover it.
- I gave it a specification and a little website.
- The rest happened by itself.

VISI

2001

State Software



2002

The Very First JSON Message

April 2001

```
<html><head><script>
document.domain = 'fudco.com';
parent.session.receive(
    {to:"session", do:"test",
    text:"Hello world"}
);
</script></head></html>
```

The Very First JSON Message

April 2001

```
<html><head><script>
document.domain = 'fudco.com';
parent.session.receive(
    {to:"session", do:"test",
    text:"Hello world"}
);
</script></head></html>
```

The Very First JSON Message

April 2001

```
<html><head><script>
document.domain = 'fudco.com';
parent.session.receive(
    {to:"session", do:"test",
    text:"Hello world"}
);
</script></head></html>
```

The Very First JSON Message

April 2001

```
<html><head><script>
document.domain = 'fudco.com';
parent.session.receive(
    {to:"session", do:"test",
     text:"Hello world"}
);
</script></head></html>
```

The Very First JSON Message

April 2001

```
<html><head><script>
document.domain = 'fudco.com';
parent.session.receive(
    {to:"session", do:"test",
    text:"Hello world"}
);
</script></head></html>
```

The Very First JSON Message

April 2001

```
<html><head><script>
document.domain = 'fudco.com';
parent.session.receive(
    {to:"session", do:"test",
    text:"Hello world"}
);
</script></head></html>
```

The unquoted name problem

- ES3 has a wack reserved word policy.
- Reserved words must be quoted.
- I did not want to put the list of reserved words in the JSON spec, so...
- All keys must be quoted.
- It significantly simplified JSON.
- This conforms to Python.

Nested HTML

```
<html><head><script>  
document.domain = 'fudco.com';  
parent.session.receive(  
    {"to": "session", "do": "test",  
     "text": "</script>"}  
);  
</script></head></html>
```

Nested HTML

```
<html><head><script>
document.domain = 'fudco.com';
parent.session.receive(
    {"to": "session", "do": "test",
     "text": "<\/script>"}
);
<\/script><\/head><\/html>
```

JSML

JavaScript Message Language

JSON

JavaScript Object Notation

JSON was really useful

- Browser/server communication.
- Interserver communication.
- Configuration.
- JSON database.

Our customers said

- “Never heard of it.”
- “Sorry, our company just committed to XML.”
- “It is not a standard.”

I bought JSON.org

- A one-page web site that described JSON.
- Grammar three ways
 - Simplified BNF
 - Railroad diagrams
 - Informal English
- A Java reference implementation.
- And then I retired.

And that's all I did.

A message format in a bottle.

Contributors

Languages

- ActionScript
- C
- C++
- C#
- ColdFusion
- D
- Delphi
- E
- Eiffel
- Erlang
- Fan
- Flex
- Haskell
- haXe
- Java
- JavaScript
- Lasso
- Lisp
- LotusScript
- Lua
- Objective C
- Objective CAML
- OpenLaszlo
- Perl
- PHP
- Pike
- pl/sql
- PowerShell
- Prolog
- Python
- R
- REALbasic
- Rebol
- Ruby
- Scheme
- Squeak
- Tcl
- Visual Basic
- Visual FoxPro

JSON is the intersection of modern programming languages

- Simple values

 - number

 - string

 - boolean

- Sequence of values

 - array, vector, list

- Collection of named values

 - object, record, struct, map, hash, property list

Recursive descent

```
value = function () {  
  
    // Parse a JSON value. It could be an object, an array,  
    // a string, a number, or a word.  
  
    white();  
    switch (ch) {  
    case '{':  
        return object();  
    case '[':  
        return array();  
    case '"':  
        return string();  
    case '-':  
        return number();  
    default:  
        return ch >= '0' && ch <= '9' ? number() : word();  
    }  
};
```

State Machine

```
state = 'go';
stack = [];
try {
    for (;;) {
        r = tx.exec(source);
        if (!r) {
            break;
        }
        if (r[1]) {
            action[r[1]][state]();
        } else if (r[2]) {
            value = +r[2];
            number[state]();
        } else {
            value = debackslashify(r[3]);
            string[state]();
        }
        source = source.slice(r[0].length);
    }
}
```

Eval

```
if (/^[\\],:{}_s]*$/ .test(text.replace(
    /\\(?:[\"\\\/bfnrt]|u[0-9a-fA-F]{4})/g, '@') .replace(
/"^[^"\\\/n\r]*"|true|false|null|-?\d+(?:\.\d*)?(?:[eE][+\-]?\d+)?/
g, ' ')) .
    replace(/(?:^|:|,)(?:\s*\[)+/g, ''))) {

// In the third stage we use the eval function to compile the
// text into a JavaScript structure. The '{' operator is subject
// to a syntactic ambiguity in JavaScript: it can begin a block
// or an object literal. We wrap the text in parens to eliminate
// the ambiguity.

    j = eval('(' + text + ')');
```

JSON.parse

- Part of ECMAScript, Fifth Edition
- Available now in better browsers everywhere.
- Very fast. Very reliable.

Languages

- Arabic
- Bulgarian
- Chinese
- Czech
- Dutch
- French
- German
- Greek
- Hebrew
- Hungarian
- Indonesian
- Italian
- Japanese
- Korean
- Persian
- Polish
- Portuguese
- Russian
- Slovenian
- Spanish
- Turkish
- Vietnamese

Ajax!

2005

Improvements

- Removed comments.
 - Dangerous practices
 - Unnecessary complexity
 - Alignment with YAML
- Added **e** notation to numbers.

No version number.

- JSON will not be changed.
- Stability is more important than any feature we can think of.
- Perhaps someday it will be replaced.

RFC 4627

application/json

Minimalism

It can fit on the back of a business
card.

The less we need to agree on,
the easier it is to interoperate.

Influences

Lisp

S-expressions

1958

Rebol

JavaScript
Python
NewtonScript

NeXT

OpenStep Property Lists

1993

XML

The High Priced Spread

HTML

Always bet on angle brackets.

Ask not if it is good enough.
Ask if it can be popular enough.

Maybe only something
this simple could work.

John Seely Brown

CTO Forum, San Francisco

April 2002

Maybe only something this
complicated could work.

InfoWorld Next-Generation Web
Services II: The Applications

Santa Clara

September 2002

XMLsucks.org

Why XML is technologically
terrible, but you have to use it
anyway

XML is the standard so shut up.

SHUT UP!

XML Alternatives

- JSON config YAML CanonML HDF SSYN OGDL SDL DL Boulder ONX SMEL Property lists ConfigObj GroovyMarkup ATerms LNML GODDAG JITTs Esis/Pyxie ConciseXML SML TexMecs A specification language Waterken doc UBF Xqueueze Ool atx Grutatxt APT txt2docbook txt2tags AsciiDoc reStructuredText Epytext EtText AFT txt2html Setext Latte Confluence Markdown SmartyPants Textile Atox CDuce MarkupMatrix WikiMI IWML SEXP sfsexp Lambda markup language SXML Better markup tXML SOX SLiP ezex Tanga (NBML) XSLScript & TerseXML Lx NiceXSL PXSL ShoXS XSCS SML MIN MINML ESPX PXML GMarkup ASN BLOB SDXF CTX ASDL WDDX REBOL
- <http://www.pault.com/pault/pxml/xmlalternatives.html>

Disruption

Threats

It's not even XML!
Who did this travesty?
Let's find a tree and string
them up. Now.

Dave Winer, 2006-12-20

any damn fool could produce
a better data format than XML

James Clark, 2007-04-06

Use the right tool
for the right job.

When your only tool is a wrench,
every problem looks like a nail.

Where did the idea come from
that data should be represented
by a document format?

RUNOFF

.SK 1

Text processing and word processing systems typically require additional information to be interspersed among the natural text of the document being processed. This added information, called "markup", serves two purposes:

.TB 4

.OF 4

.SK 1

1.#Separating the logical elements of the document; and

.OF 4

.SK 1

2.#Specifying the processing functions to be performed on those elements.

.OF 0

.SK 1

GML

:h1.Chapter 1: Introduction

:p.GML supported hierarchical containers, such as

:ol

:li.Ordered lists (like this one),

:li.Unordered lists, and

:li.Definition lists

:eol.

as well as simple structures.

:p.Markup minimization (later generalized and formalized in SGML),

allowed the end-tags to be omitted for the "h1" and "p" elements.

:eol.

::ol.

Brian Reid's Scribe

@Quote (Any damn fool)

() [] { }
< > " " ' '

@Begin (Quote)

Any damn fool

@End (Quote)

1980

Scribe

```
@techreport(PUB, key="Tesler",  
author="Tesler, Larry",  
title="PUB: The Document Compiler",  
year=1972, number="ON-72", month="Sep",  
institution="Stanford University  
Artificial Intelligence Project")
```

```
@book(Volume3, key="Knuth",  
author="Knuth, Donald E.",  
title="Sorting and Searching",  
publisher="Addison-Wesley",  
year=1973, volume=3,  
series="The Art of Computer Programming",  
address="Reading, Mass.")
```

License

MIT

The Software shall be used
for Good, not Evil.

I'm looking at you,
Osama bin Laden.

I give permission to IBM,
its customers, partners, and
minions, to use JSLint for evil.

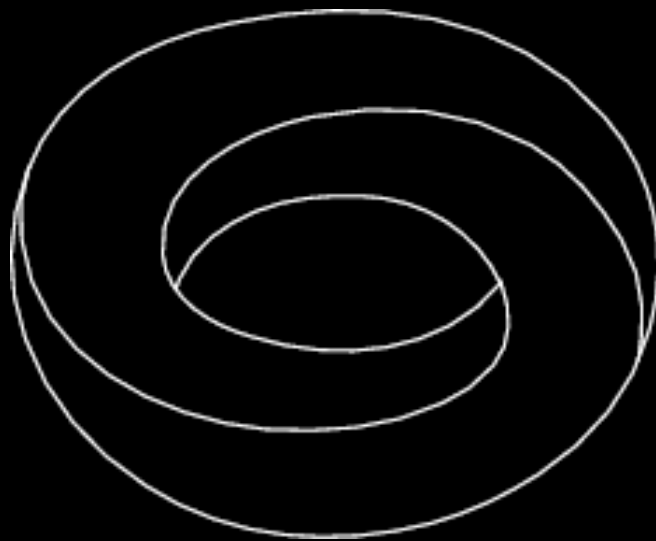
Thanks very much, Douglas!

Staff Attorney, IP Law
IBM Corporation

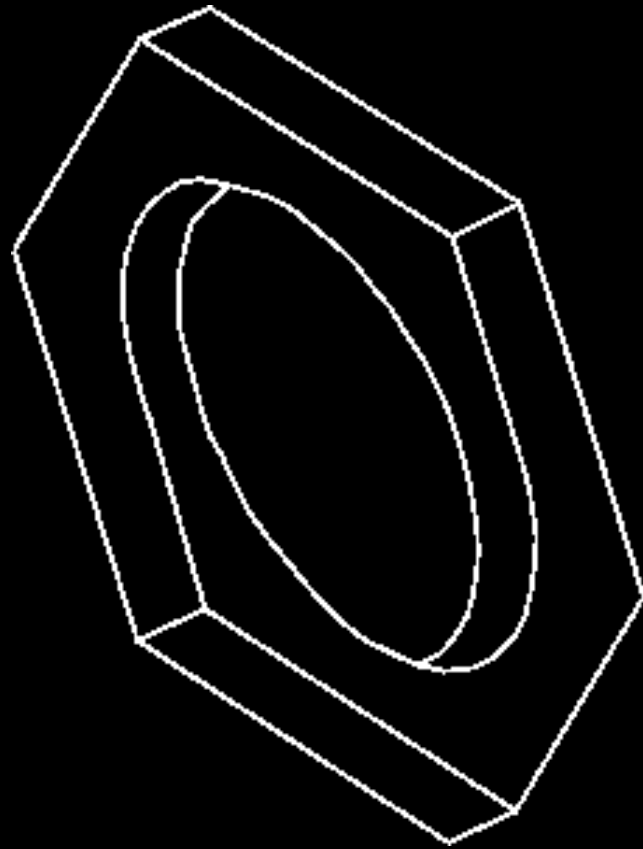
The JSON Logo



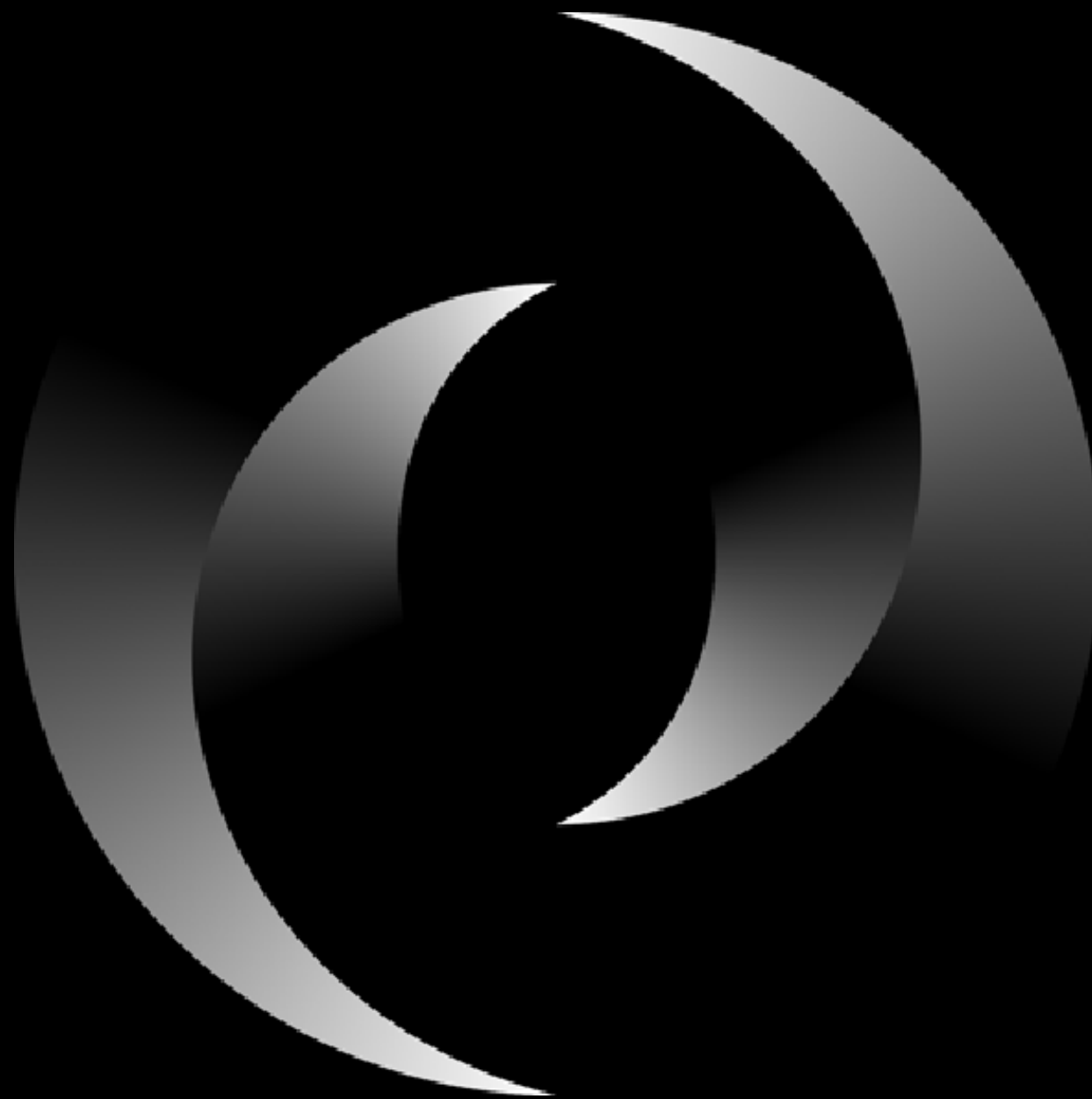
The Impossible Torus

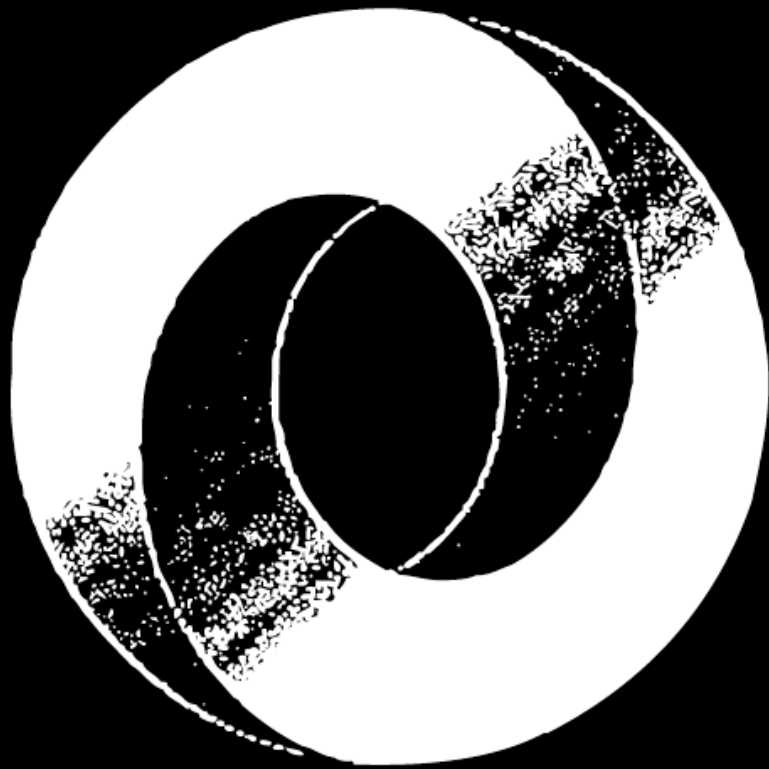


Ambihelical Hexnut









JSON
Data Interchange Format



www.JSON.org

