
<XML/> without the X **The return of** **{{Textual}} markup**

Dave Beckett
www.dajobe.org
Yahoo! Inc.

This talk is personal opinion. I am not speaking on behalf of Yahoo!

Slides presented using [Slidy](#) by [Dave Raggett](#)

Overview

- Web Textual formats and XML
- Case Study 1: Turtle
- Case Study 2: JSON-RDF

Textual Formats

Textual Formats are

*Markup that is
NOT (SGML, HTML or XML markup)*

Early Web was built on Text

- Many textual *protocols* and *formats*
- `View source` able (view protocol)
- Easy to read and write
- Binary protocols died - ASN.1, X.*, ISO protocols†

† (yes some binary formats and protocols still exist in such things as X.509 certificates, but they remain horrible)

Two Types of Web Markup

1. Presentational Markup
2. Descriptive or Semantic Markup

XML is...

1. a web data format
2. a web document markup format
... sort of, if you ignore that most of the web is HTML *Tag Soup*
3. a protocol when XML documents are exchanged over a network

let's concentrate on #1

XML for Web Data

- **Tags** elements and attributes
- **Text** inside elements and attribute values
- **Tree** of elements and text or...
a sequence of (elements, text and sequences)
unless order is ignored - but you cannot tell when
in which case it is a (sequence or hash) of (sequence or hash) ...
- **Optional validation** with DTDs, W3C xml schema, RelaxNG, schematron, DSDL, ...

XML Rocks

- standard
- mature: 10+ years old, it's bugs are well known
- in widespread use
- lots of tools
- part of software infrastructure
- Unicode support

XML Sucks

- too many standards
- old: 10+ years old, SGML legacy junk such as DTDs, Entities, PIs
- hard for humans to read: the angle brackets, hard to scan
- hard for humans to write: balancing start/end brackets
- verbose and ugly (subjective!)
- namespaces
- no built-in datatypes
- no built-in hyperlinks or URIs (thus not webby)
- the built-in DTD validation sucks
- makes it easy to design bad formats such as OPML and RSS 2.x
- often used in broken form such as non-WF, invalid against DTD

Goals of Textual Formats

1. Human readable and human writable
2. Simple

These are very different from the requirements for XML.

Applications of Web Textual Formats

- Application protocols - HTTP, SMTP
- Generic data formats - JSON, YAML, serialized PHP
- Application-specific formats - iCalendar, vCard
- Markup formats for HTML - wiki, Markdown, Textile, CSS
- Data encoded in HTML - Microformats
- Query languages - SQL, XQuery, SPARQL
- Schema languages - RelaxNG
- Programming and scripting languages - Ruby, Python, ...
- Mixed data/markup formats - DBPedia, semantic wikipedia

Main Uses of Web Textual Formats in 2007

Supporting Rich Web Applications

Asynchronous, continuous connections

Small data packets

Between browser client and website HTTP server

Providing Data for HTTP Web Services

Enabling (lowercase) web services: not WS-deathstar

GET ting data

Trends in Use of Textual Formats

- Small sized data
- Short-lived data
- Exchanging mainly *application data*, not markup
- Goal: human readable data
- Goal: enable fast development

When Textual Formats Go Bad

When Textual Formats Go Bad: Unicode

- Often initially very little or poor support (Perl, Python, PHP)
- When added later, are optional, a library or immature

XML always had this and requires support for it.

When Textual Formats Go Bad: Extension Mechanisms

Extending a format includes:

- adding a new key/field name
- adding new datatypes
- adding new structure forms (sequence, hash, ...)
- ...even having extensions allowed

Textual formats

- Tend to gain baroque retrofitted extensions mechanisms
- Some grow a huge set of syntax possibilities: YAML, wiki, Perl
- Wiki formats are asymptotic over time to 100% of HTML
- Rare to have a coherent built-in extension mechanism

XML's X is eXtensible and it has attributes, elements and namespaces. *must*

ignore also often used (outside XML formats based on validation).

When Textual Formats Go Bad: Proliferation of similar alternatives

- There are dozens, if not hundreds, of similar Wiki markup languages
- Lots of programming languages too, but that is another story

At least there is only one XML†

† (ignoring XML 1.1 which has little traction)

When Textual Formats Go Bad: Used beyond their sweet spot

The sweet spot for JSON is serializing simple data structures for transfer between programming languages. If you need more complex data structures (maybe with some kind of schema for validation), use XML

Simon Willison,

<http://simonwillison.net/2006/Dec/20/json/>

When Textual Formats Go Bad:

ESPECTOOLARGE

```
$ lynx -dump http://yaml.org/spec/current.html | wc -w  
30862
```

```
$ lynx -dump http://www.w3.org/TR/REC-xml/ | wc -w  
19526
```

```
$ lynx -dump http://www.w3.org/TR/REC-xml-names/ | wc -w  
3955
```

When Textual Formats Go Bad: Not thinking big

Often missing the technology needed for bigger projects and problems:

- not allowing local and global identifiers
- having no naming conventions or namespaces to organise names
- lack of modules, packages, inclusion mechanisms for large scale data/code

XML has a mature namespaces specification, *must ignore* and inclusion mechanisms.

When Textual Formats Go Bad: Validation

- Love it or hate it, some people want validation.
- Rare to even find in a textual format.
- Fails when working in a process driven by code/data generation (WS-deathstar) or when data structure is tied to database schema ... unless there is special support for it
- Everyone must write application-specific custom code to validate.

XML has many choices here such as none, DTD, WXS, RelaxNG, ... to declare constraints and to share them.

When Textual Formats Go Bad: Tools

Some of the textual formats can be baroque to deal with

Complex syntax detail and whitespace rules (YAML, Wiki, iCalendar, vCard, programming languages)

Each format needs its own tools

For example there are no JSON databases, standard JSON query languages, out of the box JSON APIs in standard SDKs for all major programming languages and operating systems.

If they even exist, they are all add-ons.

XML is a single common format and for tool support beats all textual format tools combined. It's syntax detail is widely implemented.

When Textual Formats Go Bad: Performance: Speed and Size

- XML has years of optimising work on it that make it perform despite it's issues
- Although some textual formats are a lot simpler than XML, the optimising work has generally not been done
- XML has been scaled and used with immense data sizes

When Textual Formats Go Bad: Whitespace with semantics

<rant>What is this, the 1950s when where you punched a hole in a card at a column number mattered? </rant>

Culprits: `make(1)` , YAML, Ruby, Python, FORTRAN 77

```
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
d e f   l e p p a r d ( y e a r ) :
      i f   y e a r   = =   2 0 0 7 :
          p r i n t   " t h i s   i s   s i l l y "
```

Case Study 1: The Turtle RDF

syntax

Main RDF Syntaxes

RDF Syntax: a way to write the RDF triples data model

RDF/XML: W3C Recommended XML syntax

OK for computers but has its own pros and cons.

See [several other](#) talks

N-Triples: simplest textual format for RDF

Parsimonious but verbose:

```
<http://www.dajobe.org/foaf.rdf#i> <http://xmlns.com/foaf/0.1,
```

Still used in tests for OWL, SPARQL, GRDDL, ...

Definitive but very verbose way to write down any RDF

RDF Syntaxes Timeline

- 1999-2004 [RDF/XML](#) (revised) - I edited this
- 2000 [Notation 3 / N3](#) - Tim Berners-Lee
- 2002-2004 [N-Triples](#) - I edited this
- 2004 [Turtle](#) - I created this
- 2004 [Regular XML RDF \(RXR\)](#) - proposal by me in XML Europe 2004
- 2007 You Are Here

Turtle – the Terse RDF Triple Language

N-Triples is too verbose for humans, so extend it but:

1. Add to the syntax only what was useful for people
2. Remain within the RDF data model
3. Only add abbreviations to:
 1. reduce verbosity
 2. remove duplication
 3. add convenience

Giving N-Triples+ or *Turtle – the Terse RDF Triple Language*

Turtle Abbreviations 1

1. Prefixed names (Verbosity) to make the URIs shorter

```
<http://www.dajobe.org/foaf.rdf#i> foaf:homepage <http://www.c
```

2. Allow relative URIs (Verbosity)

```
<#i> foaf:homepage </> .
```

3. Abbreviate repeated triple subjects (Verbosity, Duplication)

```
<#i> foaf:homepage </> ;  
    foaf:name "Dave Beckett" .
```

4. Abbreviate repeated triple subjects and predicates (Verbosity, Duplication)

```
<#i>  
    foaf:homepage </> ;  
    foaf:name "Dave Beckett" ,  
             "David Beckett" .
```

Turtle Abbreviations 2

5. Blank Nodes (Duplication)

```
<#i>
  foaf:knows [
    a foaf:Person;
    foaf:name "Edd Dumbill";
  ]
```

6. RDF Collections (Verbosity, Convenience)

```
<#i>
  tenthings:ihateaboutyou (
    "I hate the way you talk to me, and the way you cut your hair."
    "I hate the way you drive my car."
    "I hate it when you stare."
    # 7 more
  ) .
```

7. Common Data Types and Predicates (Convenience)

8. Long strings (Convenience) allow ""-multi-line quoting

Turtle: Summary

- Widely used.
- Widely implemented in Jena, Sesame, [Redland](#) (mine), rdflib, javascript, ...
- Mature - only minor changes for SPARQL in 2006
- Few open bugs and issues
- Has hit the appropriate complexity level

i.e. It is time to standardise this.

Case Study 2: Designing a new textual format: RDF-JSON

RDF-JSON

Purpose: to better enable rich web applications using RDF

- JSON is an object serializing format
- RDF's model is triples with three components parts
- Therefore: make a JSON format that is a sequence (set) of 3-part structures.

RDF-JSON - Triples-Centric Style

(based on the SPARQL [Serializing SPARQL Query Results in JSON, W3C WG Note, 2006-10-04](#))

```
{
  "subject" : {
    "value" : "http://www.dajobe.org/foaf.rdf#i",
    "type" : "uri"
  },
  "predicate" : {
    "value" : "http://xmlns.com/foaf/0.1/homepage",
    "type" : "uri"
  },
  "object" : {
    "value" : "http://www.dajobe.org/",
    "type" : "uri"
  }
}
```

Boy, that's pretty verbose. It's N-Triples again!

RDF-JSON - Resource Centric

```
{
  "prefixes" : {
    "rdf" : "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
    "foaf" : "http://xmlns.com/foaf/0.1/",
    ...
  },

  "http://www.dajobe.org/foaf.rdf#i" : {
    "rdf:type" : [ { "value" : "http://xmlns.com/foaf/0.1/Person" },
    "foaf:homepage" : [ { "value" : "http://www.dajobe.org/", "type" : "uri" },
    "foaf:name" : [ { "value" : "Dave Beckett", "type" : "literal" },
    ...
  },
}
```

Actually this looks more like Turtle or RDF/XML

RDF-JSON - Exhibit Style

```
{
  "items" : [
    {
      "label" : "i",
      "name" : "Dave Beckett",
      ...
    }
  ],
  "types" : {
    "Person" : {
      "uri" : "http://xmlns.com/foaf/0.1/Person"
    }
  },
  "properties" : {
    "name" : {
      "uri" : "http://xmlns.com/foaf/0.1/name",
      "valueType" : "item"
    },
    ...
  }
}
```

```
}  
}
```

Plug: For more on Exhibit see [XML-powered Exhibit: A Case Study of JSON & XML Coexistence](#) by Chimezie Ogbuji of Cleveland Clinic Foundation at 16:00 TODAY in the Applications track.

RDF-JSON - Choices

Clearly more work needs to be done here

- Like writing XML markup, there are multiple ways to do it
- Consider the use cases for in-browser use
- Test these ideas
- Which ones are easy to use in JavaScript?
- There is a JavaScript RDF parser that generates a different JSON with 6-element arrays - is that a better choice?

Conclusions

- Have good reasons to make a new textual format replacing XML
- Making a format for people to read/write **is** a very good reason.
- Take care! XML handles many needs well.

Thanks

Questions?

Slides (to appear at):

<http://www.dajobe.org/talks/200705-textual/>

If you have enjoyed this talk...

- <http://www.dajobe.org/>
- [Turtle Terse RDF Triple Language](#), December 2006
- D. Beckett, [Modernising Semantic Web Markup](#), 20 April 2004, XML Europe 2004, Amsterdam

You've gone too far

STOP NOW