

# **Building an ultrathin CMS with XSL and Atom**

J-P Stacey

# Atom is in two parts

- **An XML feed definition (“XML”)**
- “Natural successor to RSS”
- Schema-aware
- Can embed any other XML format, or escaped HTML
- **A publishing protocol (“PP”)**
- Based on HTTP
- add, edit, delete
- Workspace with collections and members
- Very basic

# Building a new website

- I want total, peevish control over content
- So build own publishing system
- Where to start?
- Atom PP = basic workflow. If it does Atom, it can at *least* publish
- But what about CMS – the *management*?
- Separate editing interface discards all usefulness of proven Atom PP – so how?

# How to go from Atom to a CMS?

[demonstration]

# Problems encountered

- **With Atom PP**

- Browsers don't do HTTP PUT, DELETE
- Successful actions just return 200 OK
- A collection of 1 item looks like a member

- **With premise**

- Atom is very granular – REST
- XSL, JS don't understand URLs
- Browser's own XSL
- View source - oh

# Workarounds

- **PUT/DELETE:**  
“proxy” pages
- **200 OK:** Javascript traps form submits
- **Bootstrapping:**  
elements feedtype, datatype
- & Javascript config
- **Granular:** lots of XMLHttpRequest()
- **URLs:** elements feedtype, datatype; JS config
- **Atom sniffing:**  
long XML comment
- **View source:**  
Firebug helps – but build simple HTML!

# Summary

- Atom *PP* is a good starting point for CM
- Atom *XML* can be transformed in the browser
- Browsers have limited support for HTTP, and hence for Atom. Javascript also inevitable (granularity, trapping 200 OKs)
- With some tweaks, can build a basic CMS
- Could other XML sources be treated this way? An in-browser php-update? Flickr-style click & edit for whole pages?