Syndicated content on your web pages

Jon Warbrick
University of Cambridge Computing Service
jw35@cam.ac.uk / @jw35

What?

- Fetching ***other people's*** content and including it in your web pages
- RSS, Atom, JavaScript, Server-side, client-side
- We'll look at
  - Fetching pitfalls
  - Interpretation pitfalls
  - Inclusion pitfalls

All a bit low level and geeky...

- Mostly low-level stuff
- May be handled for you by your development environment
- But how do you otherwise know what to look for?

Fetching pitfalls

Including an RSS feed

- When rendering the page
  - grab the feed
  - parse it
  - mark it up
  - include it
- What could possibly go wrong?

http://bavotasan.com/2010/display-rss-feed-with-php/

```php
<?php
    $rss = new DOMDocument();
    $rss->load('http://wordpress.org/news/feed/');
    $feed = array();
    foreach ($rss->getElementsByTagName('item') as $node) {
        $item = array (
            'title' => $node->getElementsByTagName('title')->item(0)->nodeValue,
            'desc' => $node->getElementsByTagName('description')->item(0)->nodeValue,
            'link' => $node->getElementsByTagName('link')->item(0)->nodeValue,
            'date' => $node->getElementsByTagName('pubDate')->item(0)->nodeValue,
            );
        array_push($feed, $item);
    }
    $limit = 5;
    for($x=0;$x<$limit;$x++) {
        $title = str_replace(' & ', ' &amp; ', $feed[$x]['title']);
        $link = $feed[$x]['link'];
        $description = $feed[$x]['desc'];
        $date = date('l F d, Y', strtotime($feed[$x]['date']));
        echo '<p><strong><a href="'.$link.'" title="'.$title.'">'.$title.'</a></strong><br />';
        echo '<small><em>Posted on '.$date.'</em></small></p>';
        echo '<p>'.$description.'</p>';
    }
?>
```

What could go wrong? All of these things:

- Slow (connecting, parsing XML, ...)
- Load on local server
- Load on source server
- What if it's unreachable?
- What do you display if it goes wrong?

Better:

- Seperate fetching and page rendering
- Periodic fetch, or cache on demand
  - support conditional fetch
- Validate result, keep last known good
  - warn of failure
- Cache pre-parsed, or even pre-rendered

Downsides

“A web developer had a problem.
He said “I know, I’ll use caching!”
Now he had two problems.”

Interpretation pitfalls

Computers can’t handle text

It’s all numbers

Content-Type: text/html; charset=ISO-8859-1

[In passing...]

- You really want to be using
  - The Universal Character Set (AKA Unicode)
  - The UTF8 encoding
  - http://www.cl.cam.ac.uk/~mgk25/unicode.html

ASCII

32	sp	33	!	34	"	35	#	36	$	37	%	38	&	39	'
40	(	41	)	42	*	43	+	44	,	45	-	46	.	47	/
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	<	61	=	62	>	63	?
64	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[	92	\	93	]	94	^	95	_
96	`	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{	124	|	125	}	126	~		

Mojibake

The fee will be Â£210

NescafÃ©

â€˜She saidâ€”not without some trepidationâ€”â€ŸTime to goâ€?â€™

الإعلان العالمى لحقوق الإنسان

ï»¿Ø§Ù„Ø¥Ø¹Ù„Ø§Ù† Ø§Ù„Ø¹Ø§Ù„Ù…Ù…
Ù‰ Ù„Ø-Ù‚Ùˆ Ø§Ù„Ø¥Ù†Ø³Ø§Ù†

Bottom line

Can't take 'text as numbers' and stuff it into other 'text as numbers' (unless they happen to both use the same encoding)

Either: convert numbers → text → numbers
Or: directly convert numbers → numbers

An HTML alternative

- ‘Numeric entities’
  - e.g. ‘pound’ can always be £
  - Numbers always from UCS, whatever the encoding of the document
  - So your document need only contain ASCII
  - But should still declare a character encoding

Inclusion pitfalls

The RSS “Is it HTML?” question

```
<?xml version="1.0" encoding="utf-8" ?>
<rss version="2.0">
  <channel>
    <title>Latest news</title>
    <link>http://www.cam.ac.uk/news</link>
    <item>
      <title>Fun Lab: a big hit at the Big Weekend!</title>
      <link>http://www.cam.ac.uk/public-engagement/news/</link>
      <description>Some had just wandered in to see what all
        the fuss was about, as children and adults queued
        to take part in the hands-on activities, while
        others had heard about the Fun Lab on the radio
        or read about it in the Cambridge Evening News.
      </description>
    </item>
    <item>
      ...
    </item>
  </channel>
</rss>
```

So, if you find

`&lt;strong&gt;Boo!&lt;/strong&gt;`

do you render it as

`<strong>Boo!</strong>`

or as '**Boo!**' ?

[Atom gets this right]

```
<title type="text">AT&amp;amp;T bought by SBC!</title>
```

```
<title type="html">
  AT&amp;amp;T bought &lt;b&gt;by SBC&lt;/b&gt;!
</title>
```

```
<title type="xhtml">
  <div xmlns="http://www.w3.org/1999/xhtml">
    AT&amp;amp;T bought <b>by SBC</b>!
  </div>
</title>
```

- If the answer is ‘plain text’ that that’s easy:
  - encode ‘<‘ and ‘>’ as ‘<’ and ‘>’
- But if not, you have a new problem
  - you can’t include HTML verbatim from an untrusted source
  - http://www.bbc.co.uk/news/technology-12608651

And the answer is...

- Only include safe content
  - note: not ‘Exclude dangerous content’
- parse → filter → serialise
- And worry about character encodings
  - Who’s have thought that `+ADw-script+AD4-` could be dangerous?

Promiscuous JavaScript considered dangerous

- Common JavaScript document.write() trick
- Attractive - no server-side work required
- Hands entire page to JS author
  - ...or anyone else
- Actually applies to any 3rd party JS

So, to sum up...

- It's harder than it looks
- Think about (at least):
  - How you retrieve content
  - How you intrepret it
  - How you include it

If you have been,

thanks for listening.

Any questions?