RandomInterestingSnippetsFromBugzilla

WTF?

This page is an intermediate dumping place for snippets of general interest from bugzilla, so that they don't slip from the radar when the corresponding bugs are closed.

DHTML/Cocoon musings

```
8) the admin section is heavily DHTML-ized. It works on all 6th
generation browsers and the user is able to upload images and such
without requiring rountripping

[by setting the 'src' property of an img with the value of the <input
type="file"> it is possible to visualize the image the user is uploading
right from disk, without requiring any rountripping. this improves the
user experience *immensively*!]

[also, using dom cloneNode() it is possible to keep on adding images in
the form without requiring roundtripping.]

My experiences in DHTML programming with 6th browsers has been very
positive. IE and Moz are pretty much compatible, there are only a few
issues that really bug me:

 1) if you have <div class="blah"> blah is found in div.class in mozilla
but div.className in IE! sometimes I think those guys in redmond just
need more sunshine. I ended up testing for 'document.all' to
discriminate between the two DOM types.

 2) the event model is *completely* different. there are tricks to make
the same code work on both, but things like onclick(), onchange() and
onfocus() never seems to work the same way. I ended up using
onpropertychange() which is IE-specific. It is amazing how two different
browsers can claim compliance with a spec, but still be completely not
interoperable.

To be honest, I have to say this is not only Redmond's fault: the DOM
spec lacks any connection with the system. So, it says that events are
pushed, and what is their names, but not *how* and *in what order*!

As a result, I wrote some 300 lines of javascript for the client side
and only 4 of them were browser-dependent.

 9) I used Mozilla for development. If you are using another browser to
develop your sites, throw it away and use mozilla. If you haven't done
so, please read:
http://devedge.netscape.com/viewsource/2003/mozilla-webdev/

but even better, go to http://livehttpheaders.mozdev.org and download
that awesome plugin that shows you the dump of all the headers that flow
thru between you and the server. This saved me hours of cocoon-view
based debugging, expecially on multipart forms. Not counting the ease of
use of the javascript console compared to the stinking useless IE error
popup window. yuck.

 10) at the end, this is a web site designed with *extreme usability* in
mind. I spent endless hours trying to remove *everything* possible from
the site without sacrificing the information that the site needed to
transmit. Also, the site is completely manageable by people who are
barely able to read email.

 11) despite the ability of IE6 on all machines that need to access the
internal CMS, I decided *NOT* to use any contentEditable solution but to
use pure forms for two reasons:

   a) their content is always structured
```

b) web users are used to forms, but not to inline editable pages (yet, at least). Forms provide visual semantics which are generally understood, inline editing is not standardized and without proper visual indications, users assume that if it's not in a textarea, the content is simply not editable. I think that even in a future of advanced inline editing, forms will still have their pretty consistent usage because of the clear visual semantics that are associated to them.

- o -

Lessons learned:

a) cocoon can be very useful for small sites, but the hard part is *not* to use all its features and get down to easy stuff like a velocity template that generates XHTML. Still, it can provide very useful features even in that case (think style wrapping instead of header/footer inclusion)

b) flowscript rocks the planet. it will rock even more combined with hot-deployable avalon components.

c) LiveConnect (the glue between javascript and java) makes it hard to abuse the flow to write business logic in it. Even after a few lines of having to call the classes by their full package names, you spin it off into their own classes and call them. This turns out to be very straightforward and keeps the flow *very* clean.

d) flow + inputmodules + redirection from flow totally substitute the need for actions in the sitemap. The elegance of the resulting solution is not even close to be action-based equivalent.

e) sessions and continuations do need to cohexist and they do very well already.

f) cocoon needs an xml repository as an avalon component accessible from the flow! the use of protocols is simply not enough for the kind of data manipulation required in seriously roundtripping webapps that have to mix, match and change stored xml content. This doesn't need to be an xml database, it could be a virtual file system on top of a blob-capable DB or a CVS view.

Hope this helps.

--
Stefano.

- posted by GregorRothfuss, http://issues.apache.org/bugzilla/show_bug.cgi?id=19575