Subject: Hacking into the XSLT process
Posted by jsmeaton on Fri, 22 Aug 2008 12:54:56 GMT
View Forum Message <> Reply to Message

One of our requirements in a project I'm working on is the need to export data to portable XML files. Specifically, the client wants to be able to work offline and upload data back into the system on return.

An idea I've had is to hijack the XSLT process and instead write the generated HTML to file. I'd probably do this by including another button in the sub-menu (add/update/delete/export to file) and generating an entry point to the various screens in the exported package.

I'm not very good with xsl/xml but I've had a look at std.buttons.xsl and assume this is what generates the menu. Where does it get the information on what buttons to generate and what 'action' to give them?

Basically I'm after:
How do you print the buttons?
How do you code behind the buttons?

Your help would be greatly appreciated!

Subject: Re: Hacking into the XSLT process
Posted by AJM on Fri, 22 Aug 2008 15:17:05 GMT
View Forum Message <> Reply to Message

If you want to export data in XML format then why bother to go through an XSLT process? Why can't you have a process which outputs an XML file directly without including an XSL transform? You won't then have to modify an XSL stylesheet to do what you want.

There are already several examples in the framework of functions which export data to non-database files (such as the dictionary export and the menu subsystem export), so why can't you use one of those as a model? It is very easy with PHP to create an XML document and then write it out to disk.

All of the xsl templates get their data from the same source - the XML file which is generated from the PHP code just before the XSLT process is executed. The PHP code which creates the XML document can be found within file 'include.xml.php?.inc'.

Subject: Re: Hacking into the XSLT process
Posted by jsmeaton on Sun, 24 Aug 2008 05:35:23 GMT
View Forum Message <> Reply to Message

Basically what we want to do is export the data and screens for viewing/updating/inserting offline (without access to a database) with limited functionality of course. Instead of writing data to the database (on insert/update) we'd just write to an XML file then load that back in.

I was wondering though, how the buttons in the sub-menu are created and where to place code behind the buttons. For example, how is the 'add' button created and where would the code behind that button be?

If I see an example of how an existing button works, I can create my own also.

re. the include.xml.php file, thanks, I'll look into that tonight.

Subject: Re: Hacking into the XSLT process
Posted by AJM on Sun, 24 Aug 2008 10:20:53 GMT
View Forum Message <> Reply to Message

That submenu is known as the navigation bar and is populated using entries on the mnu_nav_button table. Each of these entries points to another task which is activated when the button is pressed.

All you need do is create a task which does what you want, then put it on the navigation bar of the relevant parent task.

If you look close enough you will see that the framework is littered with hundreds of examples, including the ones I have already identified for you in my previous post.

Subject: Re: Hacking into the XSLT process
Posted by jsmeaton on Mon, 25 Aug 2008 04:07:14 GMT
View Forum Message <> Reply to Message

Thanks for that Tony, much help.

Actually, what I would really appreciate your advice on is this... how would you go about creating an 'offline mode' for standalone users? We need to provide the functionality to export data & views/forms to a laptop, have someone go away for days at a time doing data entry on that data, then coming back and syncing.

This one is gonna be keeping me up nights if I don't get it sorted soon!

Of course if we implemented it we'd pass it on for you! Sharing IS caring after all. Cheers mate.

Subject: Re: Hacking into the XSLT process
Posted by AJM on Mon, 25 Aug 2008 10:07:10 GMT
View Forum Message <> Reply to Message

One way I know of to have both online and offline versions of the same application is to use

XFORMS, but I have never used this technology so am I unable to advise you on how to go about it.

An easy alternative would be to put a copy of Apache/PHP/MySQL on each laptop, with the appropriate databases. All you then have to worry about is synchronising the databases to keep them in step. One method of copying from a laptop to the central server may be to use the contents of the AUDIT database.
