EZ-CMS Potential Improvements

William Matheson

December 14th, 2012

There are numerous shortcomings in EZ-CMS – if this were to be a production CMS, most of them would have to be addressed.

All entities:

There is no interface handling of a case where a user tries to create / rename something to a name that already exists. The names are primary keys in the database and the database would complain. Perhaps there should even be a nonce primary key (like an auto-incrementing integer) in addition to making sure that the names are unique.

Areas:

When creating or editing an area, the user should be presented with the ordering as it exists so they can easily see where the area would fit in and assign its order accordingly.

If someone picks an order that already is occupied, the higher-numbered areas should all go up one number to give space for the new one.

Articles:

Articles can only point at zero or one pages and zero or one content areas. Ideally they should be able to point at zero to many of both. The database would need to be re-designed to have join tables of articles/pages and articles/areas.

When creating or editing an article, a user should be presented with lists of valid page and area assignments to choose from, if they wish to make such assignments.

Front End:

Links to "return" from the forms (without performing submissions) simply return the user to the default page ("page1"), not necessarily the page thon was viewing.

A lot of critical components are articles and therefore user-editable (the login, the link to CManager), and there isn't any functionality to restore the originals.

The links are an article but should instead be generated dynamically so that the editor doesn't have to add a new link every time a new page is added.

Wordsmith functions in a pleasing way except that it connects to an inadequate dictionary. Wiktionary would be far superior. Given time, a developer (hey, is that me?) could and probably should implement a custom widget to work with Wiktionary.

Security:

Who is logged in, from where, time of sign-in, time of sign-in or time-out, etc.. should be logged.

The authenticated status should be capable of timing-out after a period of inactivity, and the person implementing the CMS can then decide whether to have the time-out and/or how long the inactivity window should be.

Templates:

This is easily fixed by a designer who knows what thon is doing, but at the moment the "default" and "alternate" templates do not have even column heights. It can be achieved, but it requires some CSS trickery.

Users:

User privileges are stored in "Users" when they should be part of their own tables – a table of Privileges and their descriptions, plus a table joining users/privileges. For just a handful of permissions, having Boolean flags in Users is fine, but imagine doing Unix or Windows permissions that way. It would also be much more difficult to manage permissions *en masse*.