Another RC4 Update - Release Status

NEWS_PDF_AUTHOR: onokazu

NEWS_PDF_DATE: 2002/11/25 23:20:00

The followings are some of the big changes, and I am sure there are lots of other small and big changes that are not even listed here. Please read on for more.

**** Image Manager ****
- This will allow you to upload images for inclusion in news articles, forum posts, etc. This module will also help in organizing images through categorization and group permission features.

**** Avatar Manager ****
- This will allow you to upload avatar images from which users can select in profile page, and view custom avatar images that have been uploaded by users.

**** Theme Set Manager ****
We have been working tremendously on the new theme structure utilizing the Smarty template system, and we will be changing the old Nuke theme concept to a more robust, scalable, and customizable theme management system, namely **Theme Set Manager**.

What we have been calling **themes** will be called **skins** which consist a subset of a **theme set**. A theme set consists of skin files (skin.html, stylesheet files, skin image files), and optionally module template files. Skin, image, template files for a theme set are all stored in database which are injected/compiled to cache directories whenever needed. Therefore there will be no database interaction when calling these files from the user side.

Theme Set Manager enables you to **edit/preview/download/upload** skin files and **edit/preview/delete/download/upload/generate** each of the template files. Also it will allow you to **clone/delete** a complete theme set, or **download/upload theme set tarballs**. In addition, image files within a theme set are managed in group as an image set and can be **replaced/deleted/added** online as well. Currently image files that are used in skins are managed like this, but we will be implementing this feature to image files in modules as well, so that images such as icons in the forums module can be switched via Theme Set Manager.

A theme set tarball should have a directory structure as shown below (the ones in red are required)

```
[b][color=ff0000]your_theme_name/[/color][/b]
        --- themeset.xml (themeset definition file)
        [b][color=ff0000]skins/[/color][/b]
            --- [b][color=ff0000]skin.html[/color][/b] (defines the layout of the site)
            --- style.css (default style sheet)
            --- styleNN.css (used for Netscape Navigator)
            --- styleMAC.css (used for Mac browsers)
            images/
                --- logo.gif
                ...

        templates/
            system/
                --- userinfo.html
                --- rss.html
                --- imagemanager.html
                --- imagemanager2.html
                blocks/
                    --- block_login.html
                    --- block_user.html
                    --- block_online.html
                    ...

            news/
                --- newsindex.html
                --- article.html
                --- archive.html
                ...
                blocks/
                    --- block_topics.html
                    ...

            newbb/
                --- forumindex.html
                ...

            ...
```

Yes, and the old theme files will still work fine as before by placing them under the **themes** directory

**** Enhanced module installer and config genearator ****
- The installer will automatically add entries in the cofiguration table based on the contents of xoops_version.php, which will be used to genearate config option form dynamically. This will elimiate the need of creating a cache folder/file within each module directory.

**** Enhanced blocks management ****
You can:
- select which page a block should be displayed on now
- cache block contents by Smarty with settable caching expirations
- edit module block contents via Theme Set Manager

**** XML-RPC for News module ****
XOOPS will support the following XML-RPC APIs
- All Blogger API methods
- All Metaweblog API methods
- MoveableType API
- XOOPS API

In RC4, only the News module has XML-RPC feature implemented. However, in the future release when we introduce the new sections module, you would be able to post not only news but downloads, links, reviews, and other type of items.

**** Fully redesigned core class structure ****
What we have been doing for the past several months is to separate data objects from its data resource access mechanisms in XOOPS. Until RC3, data objects and its data resource accessor was too tightly coupled which prevented the use of other data storage systems, such as LDAP, XML files, etc. By de-coupling the tight integration, we can in the future switch to other data resources fairly easily, which is in general just replacing the data accessor object classes.

The following classes have now been converted to the new structure

[b][u]RC3[/u][/b] [b][u]RC4[/u][/b]

XoopsObject XoopsObject (abstract data object)
XoopsObjectManager (abstract data accessor object)

XoopsModule XoopsModule (data object)
XoopsModuleManager (data accessor object)

XoopsGroup XoopsGroup (data object)
XoopsGroupManager (data accessor object)
XoopsMembership (data object)
XoopsMembershipManager (data accessor object)
XoopsGroupPerm (data object)
XoopsGroupPermManager (data accessor object, child class of XoopsGroupManager)
XoopsModulePermManager (data accessor object, child class of XoopsGroupManager)
XoopsImgCatPermManager (data accessor object, child class of XoopsGroupManager)
(Developers can create their own group pemission accessor class, and call it dynamically using a helper function)

XoopsUser Not yet

XoopsBlock Not yet

XoopsUserSession XoopsSession (data object)
XoopsSessionManager (data accessor object)

None XoopsConfig (data object)
XoopsConfigManager (data accessor object)

None XoopsConfigCategory (data object)
XoopsConfigCategoryManager (data accessor object)

None XoopsConfigOption (data object)
XoopsConfigOptionManager (data accessor object)

None XoopsImage (data object)
XoopsImageManager (data accessor object)

None XoopsImageCategory (data object)
XoopsImageCategoryManager (data accessor object)

None XoopsAvatar (data object)
XoopsAvatarManager (data accessor object)

XoopsPM XoopsPrivMessage (data object)
XoopsPrivMessageManager (data accessor object)

None XoopsThemeset (data object)
XoopsThemesetManager (data accessor object)

None XoopsImgset (data object)

XoopsImgsetManager (data accessor object)

None XoopsImgsetImage (data object)
XoopsImgsetImageManager (data accessor object)

All data object classses and data accessor object classes are derived from the XoopsObject abstract class and XoopsObjectManager class respectively. In addition, all data accessor object classes are called via the XoopsManagerFactory class, which further eliminates the need of calling the accessor classes directly.

XOOPS Team

The followings are some of the big changes, and I am sure there are lots of other small and big changes that are not even listed here. Please read on for more.

**** Image Manager ****
- This will allow you to upload images for inclusion in news articles, forum posts, etc. This module will also help in organizing images through categorization and group permission features.

**** Avatar Manager ****
- This will allow you to upload avatar images from which users can select in profile page, and view custom avatar images that have been uploaded by users.

**** Theme Set Manager ****
We have been working tremendously on the new theme structure utilizing the Smarty template system, and we will be changing the old Nuke theme concept to a more robust, scalable, and customizable theme management system, namely **Theme Set Manager**.

What we have been calling **themes** will be called **skins** which consist a subset of a **theme set**. A theme set consists of skin files (skin.html, stylesheet files, skin image files), and optionally module template files. Skin, image, template files for a theme set are all stored in database which are injected/compiled to cache directories whenever needed. Therefore there will be no database interaction when calling these files from the user side.

Theme Set Manager enables you to **edit/preview/download/upload** skin files and **edit/preview/delete/download/upload/generate** each of the template files. Also it will allow you to **clone/delete** a complete theme set, or **download/upload theme set tarballs**. In addition, image files within a theme set are managed in group as an image set and can be **replaced/deleted/added** online as well. Currently image files that are used in skins are managed like this, but we will be implementing this feature to image files in modules as well, so that images such as icons in the forums module can be switched via Theme Set Manager.

A theme set tarball should have a directory structure as shown below (the ones in red are required)

[b][color=ff0000]your_theme_name/[/color][/b]
--- themeset.xml (themeset definition file)
[b][color=ff0000]skins/[/color][/b]
--- [b][color=ff0000]skin.html[/color][/b] (defines the layout of the site)

```
        --- style.css (default style sheet)
        --- styleNN.css (used for Netscape Navigator)
        --- styleMAC.css (used for Mac browsers)
        images/
            --- logo.gif
            ...

    templates/
        system/
            --- userinfo.html
            --- rss.html
            --- imagemanager.html
            --- imagemanager2.html
            blocks/
                --- block_login.html
                --- block_user.html
                --- block_online.html
                ...

        news/
            --- newsindex.html
            --- article.html
            --- archive.html
            ...
            blocks/
                --- block_topics.html
                ...

        newbb/
            --- forumindex.html
            ...

        ...
```

Yes, and the old theme files will still work fine as before by placing them under the **themes** directory

**** Enhanced module installer and config genearator ****
 - The installer will automatically add entries in the cofiguration table based on the contents of xoops_version.php, which will be used to genearate config option form dynamically. This will elimiate the need of creating a cache folder/file within each module directory.

**** Enhanced blocks management ****
You can:
- select which page a block should be displayed on now
- cache block contents by Smarty with settable caching expirations
- edit module block contents via Theme Set Manager

**** XML-RPC for News module ****
XOOPS will support the following XML-RPC APIs
- All Blogger API methods
- All Metaweblog API methods
- MoveableType API
- XOOPS API

In RC4, only the News module has XML-RPC feature implemented. However, in the future release when we introduce the new sections module, you would be able to post not only news but downloads, links, reviews, and other type of items.

**** Fully redesigned core class structure ****
What we have been doing for the past several months is to separate data objects from its data resource access mechanisms in XOOPS. Until RC3, data objects and its data resource accessor was too tightly coupled which prevented the use of other data storage systems, such as LDAP, XML files, etc. By de-coupling the tight integration, we can in the future switch to other data resources fairly easily, which is in general just replacing the data accessor object classes.

The following classes have now been converted to the new structure

[b][u]RC3[/u][/b] [b][u]RC4[/u][/b]

XoopsObject XoopsObject (abstract data object)
XoopsObjectManager (abstract data accessor object)

XoopsModule XoopsModule (data object)
XoopsModuleManager (data accessor object)

XoopsGroup XoopsGroup (data object)
XoopsGroupManager (data accessor object)
XoopsMembership (data object)
XoopsMembershipManager (data accessor object)
XoopsGroupPerm (data object)
XoopsGroupPermManager (data accessor object, child class of XoopsGroupManager)
XoopsModulePermManager (data accessor object, child class

of XoopsGroupManager)
XoopsImgCatPermManager (data accessor object, child class of XoopsGroupManager)
(Developers can create their own group pemission accessor class, and call it dynamically using a helper function)

XoopsUser Not yet

XoopsBlock Not yet

XoopsUserSession XoopsSession (data object)
XoopsSessionManager (data accessor object)

None XoopsConfig (data object)
XoopsConfigManager (data accessor object)

None XoopsConfigCategory (data object)
XoopsConfigCategoryManager (data accessor object)

None XoopsConfigOption (data object)
XoopsConfigOptionManager (data accessor object)

None XoopsImage (data object)
XoopsImageManager (data accessor object)

None XoopsImageCategory (data object)
XoopsImageCategoryManager (data accessor object)

None XoopsAvatar (data object)
XoopsAvatarManager (data accessor object)

XoopsPM XoopsPrivMessage (data object)
XoopsPrivMessageManager (data accessor object)

None XoopsThemeset (data object)
XoopsThemesetManager (data accessor object)

None XoopsImgset (data object)
XoopsImgsetManager (data accessor object)

None XoopsImgsetImage (data object)
XoopsImgsetImageManager (data accessor object)

All data object classses and data accessor object classes are derived from the XoopsObject abstract class and XoopsObjectManager class respectively. In

addition, all data accessor object classes are called via the XoopsManagerFactory class, which further eliminates the need of calling the accessor classes directly.

XOOPS Team