XOOPS Documentation Team

for

XOOPS 2.6

Module - *xoops_version.php*

by

Simon Antony Roberts [wishcraft]
wishcraft@users.sourceforge.net

Publication Date: 24/08/17

Foreword

This document is to outline the xoops_version.php file for modules in XOOPS 2.6 – the system is to reflect this dynamic change to the module system to support congruent systems and overlay of existing infrastucture and to allow the encompassing variables allowed for the environmental settings as well as features of the modules included in XOOPS 2.6

Variables

The following list is compiled and comprises of the `$modversion` array that the xoops_version.php in the root of /modules/dirname/ is found to define the module settings and variables in the XOOPS 2.6 release.

$modversion["name"]

This variable contains the name of the module, that is 2 – 4 words describing it title.

$modversion["version"]

This variable contains the version which is just major and minor revision numbers to the run of x.xx

$modversion["description"]

This variable contains the textual description of the module along with the name

$modversion["dirname"]

This variable contains the dirname of the module that is the path that follows /modules/thisfolder

$modversion["releasedate"]

This variable contains the release date in format YYYY/MM/DD

$modversion["status"]

This variable is an enumerator that consists of the following variables: alpha, beta, testing, release canditate, release, final minor, final major, final

$modversion["credits"]

This variable contains a text string for any 3rd party credits above any current desciption of the concurrent fields.

$modversion["help"]

This variable contains the path to the help reference for the control panel.

$modversion["license"]

This variable contain the license code that the module is distributed under ie. GPL1, GPL2, GPL3, MIT

$modversion["academic"]

This variable is a boolean field if it is also an acedemic licensed module.

$modversion["official"]

This variable is a boolean field if it has been made official by the XOOPS Council.

$modversion["image"]

This variable contains a resolvable path within the module for the icon of the module for the admin.

$modversion["dirmoduleadmin"]

This variable contains a resolvable path around the module for the system admin diradmin path

$modversion["icons16"]

This variable contains a resolvable path around the module for the system 16x16 icons

$modversion["icons32"]

This variable contains a resolvable path around the module for the system 32x32 icons

$modversion["author"]["handle"]

This variable contains the author handle/alias/nickname/username

$modversion["author"]["realname"]

This variable contains the authors realname in full

$modversion["author"]["website"]["url"]

This variable contains the authors website URL

$modversion["author"]["website"]["name"]

This variable contains the authors website name

$modversion["author"]["email"]

This variable contains the authors website email

$modversion["author"]["word"]

This variable contains the authors word or saying about the module or anything descriptive like a biograph of the author

$modversion["author"]["feed"]

This variable contains the authors url for their RSS Feed.

$modversion["warning"][“install”]

This variable can contain straight string or HTML to display as a warning before installing it like a secondary confirmation.’

$modversion["warning"][“update”]

This variable can contain straight string or HTML to display as a warning before updating it like a secondary confirmation.’

$modversion["warning"][“uninstall”]

This variable can contain straight string or HTML to display as a warning before uninstalling it like a secondary confirmation.’

$modversion["demo"]["url"]

This variable contains the URL for the demo site for the module.

$modversion["demo"]["name"]

This variable contains the demo site name for the module

$modversion["demo"]["icon"]

This variable contains the demo site icon for the module, No more than 128x128

$modversion["support"]["url"]

This variable contains the support site URL where you can raise support or bug tickets..

$modversion["support"]["name"]

This variable contains the support site name for raising support.

$modversion["support"]["icon"]

This variable contains the support site icon for raising support. No more than 128x128

$modversion["submit"]["form"]["feature"]

This variable contains the URL for submitting a feature request; this is normally cURL and placed within your module there is a passing variable which is added to it by XOOPS in the form url $_POST[‘return’] which you put the return url to be generated in the form; so when it submits it returns to your site or development. The other variables that are added to the existing URL seeing are from the current logged in $GLOBAL[‘xoopsUser’] is $_POST[‘uname’], $_POST[‘email’], $_POST[‘name’], $_POST[‘uid’] as well as $_POST[‘xoops_url’], $_POST[‘xoops_version’], $_POST[‘xoops_db_type’], $_POST[‘xoops_db_version’], $_POST[‘xoops_module_dirname’], $_POST[‘xoops_module_version’], $_POST[‘callback-url’]

$modversion["submit"]["form"]["bug"]

This variable contains the URL for submitting a bug ticket; this is normally cURL and placed within your module there is a passing variable which is added to it by XOOPS in the form url $_POST[‘return’] which you put the return url to be generated in the form; so when it submits it returns to your site or development.The other variables that are added to the existing URL seeing are from the current logged in $GLOBAL[‘xoopsUser’] is $_POST[‘uname’], $_POST[‘email’], $_POST[‘name’], $_POST[‘uid’] as well as $_POST[‘xoops_url’], $_POST[‘xoops_version’], $_POST[‘xoops_db_type’], $_POST[‘xoops_db_version’], $_POST[‘xoops_module_dirname’], $_POST[‘xoops_module_version’], $_POST[‘callback-url’]

$modversion["people"]["developers"]

This variable is an array of developers, each line is normally ...[1][“name”]; ...[1][“email”], ...[1][“handle”]; ...[1][“version”] extending the elements of this array.

This array is maintained by releases.xoops.org or manually keyed in.

$modversion["people"]["testers"]

This variable is an array of testers, each line is normally ...[1][“name”]; ...[1][“email”], ...[1][“handle”]; ...[1][“version”] extending the elements of this array.

This array is maintained by releases.xoops.org or manually keyed in.

$modversion["people"]["translators"]

This variable is an array of translators, each line is normally ...[1][“name”]; ...[1][“email”], ...[1][“handle”]; ...[1][“version”], ...[1][“language”] extending the elements of this array.

This array is maintained by releases.xoops.org or manually keyed in.

$modversion["people"]["documenters"]

This variable is an array of documenters, each line is normally ...[1][“name”]; ...[1][“email”], ...[1][“handle”]; ...[1][“version”] extending the elements of this array.

This array is maintained by releases.xoops.org or manually keyed in.

$modversion["keys"]["module"]

This variable is a defining hash key for the module version specific, it is an identity hash that needs to change with every new version release, it is also maintained by releases.xoops.org but can be keyed manually there is no set length but maximum length is 128 bytes.

$modversion["keys"]["release"]

This variable is a defining hash key for the module itself specific, it is an identity hash that never to change with every new version release, always stays the same identifing the module key, it is also maintained by releases.xoops.org but can be keyed manually there is no set length but maximum length is 128 bytes.

$modversion["minimal"]["php"]

This variable is for the minimal php version that can be used with the module the legacy variable from XOOPS 2.5 is min_php.

$modversion["minimal"]["xoops"]

This variable is for the minimal xoops core that can be used with the module the legacy variable from XOOPS 2.5 is min_xoops.

$modversion["minimal"]["db"]

This variable is for the minimal database kernels that can be used with the module the array breaks down into the following `array('mysql' => '5.0.1', 'mysqli' => '5.1.2);` the legacy XOOPS 2.5 varaibles is min_db.

$modversion["minimal"]["admin"]

This variable is for the minimal version of the admin the module will work with the legacy XOOPS 2.5 variable is min_admin.

$modversion["sqlfile"]

This variable is an array of path resolvable SQL flat files for the generation of the tables, this works with conjunction with the constraints that are applied later in the generation of the SQL, the array works like so `array('mysql' => '/sql/mysql.sql', 'mysqli' => '/sql/mysqli.sql'…).`

$modversion["tables"]

This variable is an array of all the tables in the SQL file as a flat array, this is listed without prefix.

$modversion["hasMain"]

This variable is an boolean field for if it has a main console, that is none admin user interface.

$modversion["hasAdmin"]

This variable is an boolean field for if it has a admin webmaster console pages and system menu.

$modversion["adminindex"]

This variable is an is a resolvable path within the module for the root admin/webmaster page.

$modversion["adminmenu"]

This variable is a resolvable path for the menu in the admin that provide a system array with it.

$modversion["system_menu"]

This variable is boolean field with the rendering of the system menu.

$modversion["hasSearch"]

This variable is a boolean field with the support for a search function.

$modversion["search"]["file"]

This variable is a resolvable path within the module to a php file that contains the search functions.

$modversion["search"]["func"]

This variable is string for the function name that conducts search algorithms of the module.

$modversion["hasComments"]

This variable is boolean for the function of whether the module uses the XOOPS Commenting System.

$modversion["onInstall"]

This variable is a resolvable path for all the extra install functions for the module.

$modversion["onUpdate"]

This variable is a resolvable path for the extra update functions for the module.

$modversion["onUninstall"]

This variable is a resolvable path for the extra uninstallation functions for the module..

$modversion["sub"]

This variable is an array for submenu on the main menu on the user side it is a listed array of: …[1][‘name’]; [1][‘url’]… and so on…

$modversion["blocks"]

This variable is an array for blocks nothing has changed here since XOOPS 2.5 the variables are as so: …[1][‘file’]; …[1][‘name’]; …[1][‘description’]; …[1][‘show_func’]; …[1][‘options’]; …[1][‘edit_func’]; …[1][‘template’]; this is the variable for the sub-element of each block, below is an example of this being populated.

```
$modversion["blocks"][1] = array(
        'file' => "forum_block.php",
        'name' => _MI_XFORUM_BLOCK_TOPIC_POST,
        'description' => "Shows recent replied topics",
        'show_func' => "b_xforum_show",
        'options' => "time|5|360|0|1|0",
        'edit_func' => "b_xforum_edit",
        'template' => 'xforum_block.html');
```

$modversion["configcat"]

This variable is an array for populating grouping of categories for preferences for modules and settings, the subsquent array name correlates with the ‘config’ variable sub-element ‘category’ so preferences are not one massive drill down list they have groups then. An example of this being populated would be: `array('cat1' => array('name' => 'Search Engine Optimization', 'description' => 'this is all the settings to do with URL rewriting and SEO functions!');`

$modversion["config"]

This variable is an array for populating preferences that is configuration settings in XOOPS 2.6, the only additional field in this is ‘category’ that reflects the group the setting belongs to defined in ‘configcat’ below is an example of this being populated:-

```
$modversion['config'][] = array(
          'name' => 'htaccess',
          'title' => '_MI_HTACCESS',
          'description' => '_MI_HTACCESS_DESC',
          'formtype' => 'yesno',
          'valuetype' => 'int',
          'default' => 0,
          'category' => 'cat1');
```

$modversion["hasNotification"]

This variable is boolean field to indicate if notification exist!

$modversion["notification"]

This variable is multiple depth elemented array, which has changed in no way since XOOPS 2.5..

$modversion["hasFeeds"]

This variable is boolean field to indicate it has feed functions to provide XOOPS_ROOT_PATH / backend.php which mixed data sorted in the date field..

$modversion["feeds"]

This variable is array that points to functions and settings for providing feeds aggregated through the XOOPS Framework by the module.

```
$modversion['feed'][] = array(
        'file' => 'include/feed-functions.php',
        'func' => 'getModuleDirnameFeed',
        'many' => 'backend-number' /*         This variable can be a physical integer or a variable name
                                               of a config defined name for number of items to return */);
```

The function returns an array which the base array element names are a UNIX_TIMESTAMP() of the sub-elements of the array, this is for sorting the output feed in order by array_merge() all the components then resorting via the key names… any subsquent field ie the following output array would be within the ‘item‘ key element reserve keyword, `array('112653671' => array('item' => array('title' => 'feed article title', 'description' => 'html containing feed article', 'guid' => '2876-eywr36-7rg8-7ddg'...)));` and so on.

the backend has to make RSS XML Elements based on the sub-array key name ie <title/>, <descrption/> of any definition as there is some extra information like media embeding that needs to be done.

if there are attributes to be added say on the title field like you would be embeding media like with a wordpress RSS Feed it would be resolve an array like so.. `array('217653671' => array('item' => array('title' => array('attributes' => array('nodea' => 'media:http://xoops.org/images/logo.gif'), 'value' => 'feed article title'), ...)...);` this would result in a title xml element that would look like this in the item:

```
<title nodea=' media:http://xoops.org/images/logo.gif'>feed article title</title>….
```

Subsquently you can also apart from having UNIX_TIMESTAMP() as the base element keyname use the reserved keyname ‘header’ or the third and only other resever primary key id which is ‘channel‘ for putting things in the header of the RSS feed starting with:

```
array('header' => array('xml' => array('rss' => array('attributes' =>
array('xmlns:content'=>"http://purl.org/rss/1.0/modules/content/",
'xmlns:wfw=>"http://wellformedweb.org/CommentAPI/"),...);

array('channel' => array('atom:link' => array('attributes' =>
array('href'=>"https://internetfounder.wordpress.com/feed/", 'rel'=>"self", 'type'=>"application/rss+xml")),
'sy:updatePeriod' => 'hourly', 'sy:updateFrequency' =>'1')),...);
```

outputting this into the reserved ‘header’ element will make the header of the RSS Add the following lines to it:

```
<?xml version="1.0" encoding="UTF-8"?>
        <rss version="2.0"
                xmlns:content="http://purl.org/rss/1.0/modules/content/"
                xmlns:wfw="http://wellformedweb.org/CommentAPI/">
                <channel>
                        <atom:link href="https://internetfounder.wordpress.com/feed/" rel="self" type="application/rss+xml" />
                        ….
                        <sy:updatePeriod>hourly</sy:updatePeriod>
                        <sy:updateFrequency>1</sy:updateFrequency>
```

$modversion["hasRewrite"]

This variable is boolean field to indicate it has apache2 mod-rewrite supported as seo.

$modversion["rewrite"]

This variable is an array for defining the constraints of the .htaccess to be navigated and updated whenever the 'reserved' element name 'config' sub-element which is the name of a $modversion['config'] variable is updated, you leave the old ones there, when it first writes to XOOPS_ROOT_PATH / .htaccess it write in from the top down after it locates the comment it adds once of:

```
## XOOPS Mod Rewrite   Autoscipting
```

once the comment is located if any of the variable names listed as the keyname in $modversion['rewrite']['config'] are updated in the preference it writes them in again the full module .htaccess leaving the previous one below it, as long as they use a 301 redirect this will remove the old links from the search engine and update the new path, but you leave the older variable configuration settings below this always writing in from the top down.

Lets look at setting this array now:-

```
$modversion["rewrite"]["config"]["base"] = '%base';
$modversion["rewrite"]["config"]["html"] = '%html';
$modversion["rewrite"][1]["raw"] = 'RewriteCond %{REQUEST_FILENAME} !-f';
$modversion["rewrite"][2]["raw"] = 'RewriteCond %{REQUEST_FILENAME} !-d';
$modversion["rewrite"][3]["path"] = '^%base/index%html';
$modversion["rewrite"][3]["resolve"] = './modules/%dirname/index.php';
$modversion["rewrite"][3]["state"] = 'L,NC,QSA';
$modversion["rewrite"][4]["path"] = '^%base/([0-9]+)/([0-9]+)/(.*?)/index%html';
$modversion["rewrite"][4]["resolve"] = './modules/%dirname/index.php?start=$1&limit=$2&base=$3';
$modversion["rewrite"][4]["state"] = 'L,NC,QSA';
$modversion["rewrite"][5]["path"] = '^%base/uploads%html'
$modversion["rewrite"][5]["resolve"] = './modules/%dirname/uploads.php';
$modversion["rewrite"][5]["state"] = 'L,NC,QSA';
```

Now what is happening with this array is the 'config' reserved key name with sub-key of configuration variables names, which the value of the 'config' in 'rewrite' is the string which is str_replace(); in the raw, path + resolve of any other element except this reserved keyword key name.. there is also a reserved %dirname to replace with the $modversion['dirname'] variable as well with str_replace(); when they are updated say the configuration called 'base' was updated to in the preference say: fontier as the base path, xoops will now write up top down a new .htaccess leaving the old one in the place from the top of the comment so what would be writen to .htaccess is the following after `## XOOPS Mod Rewrite   Autoscipting`.

```
## Module: Module Name
## Module Path: %dirname
## Module Version: x.xx
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^fontier/index.html ./modules/fontier/index.php [L,NC,QSA]
RewriteRule ^fontier/([0-9]+)/([0-9]+)/(.*?)/index.html ./modules/fontier/index.php?start=$1&limit=$2&base=$3 [L,NC,QSA]
RewriteRule ^fontier/uploads.html$ ./modules/convert/uploads.php [L,NC,QSA]
## End Module mod-rewrite for %dirname written   YYYY-MM-DD H:i:s
```

$modversion["hasConstraints"]

This variable is boolean field to indicate the module has database automation constraints defined, these need to be added after the tables are created and removed before the tables are deleted.

$modversion["constraints"]

This variable is array which defines the global database constraints that are defined, the name always has dirname_ put in front of it from the module $modversion['dirname'] variable for unique basis of definition, all constraints must have unique names in MySQL Innodb…

Now lets look at setting this variable:-

```
$modversion["constraints"][1]['name'] = 'users_consent_agreements';
$modversion["constraints"][1]['tables'] = array('users', 'consent_agreements') ;
$modversion["constraints"][1]['user']['index'] = array('uid');
$modversion["constraints"][1]['consent_agreements']['index'] = array('uid');
$modversion["constraints"][1]['user']['foreign`] = array('uid');
$modversion["constraints"][1]['user']['reference`]['  consent_clientel'] = array('uid')
$modversion["constraints"][1]['user']['delete'] = 'cascade' /* enumerator can be: no action, cascade
                                                             or restrict */
$modversion["constraints"][1]['user']['update'] = 'cascade' /* enumerator can be: no action, cascade
                                                             or restrict */
```

After the module has installed and run the SQL file it has to add the constraints to the database, It will also have to remove these when uninstalling the module before deleting the table, the following SQL will be fired on installation of this module to allow for database automation constraints.

```
ALTER TABLE `users` ADD INDEX `dirname_users_consent_agreements_uid_users` (`uid` ASC);
ALTER TABLE `consent_agreements` ADD INDEX
`dirname_users_consent_agreements_uid_consent_agreements` (`uid` ASC);
ALTER TABLE `users`
ADD CONSTRAINT `dirname_users_consent_agreements`
  FOREIGN KEY (`uid`)
  REFERENCES `consent_agreements` (`uid`)
  ON DELETE CASCADE
  ON UPDATE CASCADE;
```

When the module has being uninstalled, it will have to run the following routine in SQL before it delete the tables of the module.

```
DROP INDEX `dirname_users_consent_agreements_uid_users` ON `users`;
DROP INDEX `dirname_users_consent_agreements_uid_consent_agreements` ON `users`;
ALTER TABLE `users` DROP CONSTRAINT `dirname_users_consent_agreements`;
```