Yi Xiao

A L.A.M.P. based Syndication Module Design and Implementation under XOOPS Web Content Management System Platform

Bachelor Thesis based on the examination and study regulations for the Bachelor of Engineering degree programme
Information Engineering
at the Department of Information and Electrical Engineering
of the Faculty of Engineering and Computer Science
of the University of Applied Sciences Hamburg

Supervising examiner : Prof. Dr. Ing Dieter Müller-Wichards
Second examiner : Prof. Dr.rer.nat Thomas Klinker

Day of delivery June 18th 2007

Yi Xiao

Title of the Bachelor Thesis

A L.A.M.P. based Syndication Module Design and Implementation under XOOPS Web Content Management System Platform

Keywords

Web2.0, L.A.M.P., RSS, Feed, XML, Syndication, CMS, XOOPS,Linux, Apache, MySQL, PHP, PEAR

Abstract

Inside this report the easiest way of building a L.A.M.P. based dynamic website and a Syndication XOOPS module design process are described.

Yi Xiao

Thema der Bachelorarbeit

Ein auf L.A.M.P. basiertes Syndikatsbildung-Modul-Design und Implementierung unter XOOPS Content Management System Plattform

Stichworte

Web2.0, L.A.M.P., RSS, Feed, XML, Syndikatsbildung, CMS, XOOPS,Linux, Apache, MySQL, PHP, PEAR

Kurzzusammenfassung

Innerhalb dieser Arbeit wird geschrieben,dass das Errichten einer L.A.M.P. gegründeten dynamischen Web site und ein Moduldesignprozeß der Syndikatsbildung XOOPS auf einfache Weise geschehen kann.

Contents

Chapter 1

Introduction

1.1 Web2.0

Internet is changing the whole world.

World Wide Web is getting to the second generation: web2.0, a phrase coined by O'Reilly Media in 2003[1] and popularized by the first Web 2.0 conference in 2004 [2] .

Web2.0 is indeed not a technital item, it does not refer to an update to web technical specifications, but to changes in the ways system developers have used the web platform. [3]

For most of the Web2.0 websites , the main job of the web editors and web developers is to provide a robust ,reliable and flexeble platform to users , which encourages them to add values to the platform as they use it.And users own the contents they generate inside this platform and exercise control over it .

1.2 Web2.0 and dynamic website

Most of the web2.0 sites are using dynamic website technology to provide a rich interactive user-friendly interface to their users . In contrast to a static website a dynamic website is one whose content is regenerated every time a user visits, reloads or updates their content. Contents of the websites are changing all the time on an automatic time basis .

Dynamic websites generate content from information stored in databases. Site content can be managed using a simple web browser like Internet Explorer giving you control over the information that is displayed to users. This allows content to be updated more frequently by your or someone at your organization so that you can offer features such as updated product information, press releases, news and events.

Dynamic content can dramatically improve a users experience when visiting your website because content is updated easily and on a regular basis. Updated content encourages users to return to your website to view the latest information. [4]

1.3 Keep everything in sync

People are getting information from internet with different ways , and currently many online information sources, including company portals, weblogs and news services, now broadcast their content to their visitors in so-called "syndicated feeds" or "syndication feeds" with a new technology like Really Simple Syndication (RSS) . People do not have to go to those information sources every time to get the information they like, they use aggregator software and online aggregation services to collect those information which they are really interested in by the rules defined by themselves. Obviously they want the information those tools collect are always up to date or identical to those corresponding information sources. On a regular time basis, the information of the sources will be synchronized to this software and online services .That is to say, both sides should be in sync.

Get only the information you really want and keep everything up to date. This really makes a lot of fun, and saves a lot of time and traffic.

1.4 Aim of this thesis

Let us get to the very simple question : how to build such a dynamic web2.0 based website and provide the syndication services to users with minimum time and money investment , minimum server side deployments ,and minumen IT specific knowledge?

That is the key problem we will discuss and answer in this thesis.Fortunately , we worked out one of the solutions to that question .

1.5 Structure of this thesis

We will first discuss L.A.M.P.(or LAMP) architecture,which refers to four free or open-source software : Linux, the operating system; Apache, the Web server; MySQL, the rational database management system (or database server); PHP, the programming language ,Then we will have a look at the RSS(Really Simple Syndication) technology .We will introduce the content management system(CMS) in Chapter 4 . One instant of CMS:XOOPS dynamic web CMS will come out , and we will discuss the key features of XOOPS and why we choose XOOPS CMS in Chapter 5 , then we will go through the installation and module design and implementation processes of XOOPS CMS in Chapter 6 and chapter 7.Finally we get to our conclusion and review in Chapter 8 followed by Chapter 9 References and Appendix.

Chapter 2

L.A.M.P. Architecture

2.1 What is L.A.M.P. Architecture?

L.A.M.P. is an acronym for **L**inux, **A**pache, **M**ySQL, **P**HP (or **P**ERL or **P**ython) and refers to 4 different pieces of free or open source software. Sometimes referred to as "The L.A.M.P. Stack"

- L = Linux operating system
- A = Apache web server
- M = MySQL database
- P = PHP/Python/Perl scripting languages

Let us get into the details of these four components. They are structured in a layered way. Each layer provides a critical part of the entire software stack:

- **Linux.** Linux is the lowest-level layer and provides the operating system. Linux actually runs each of the other components. You are not specifically limited to Linux, however; you can easily run each of the other components on Microsoft®; Windows®;, Mac OS X, or UNIX® if you need to.
- **Apache.** The next layer is Apache, the Web server. Apache provides the mechanics for getting a Web page to a user. Apache is a stable, mission-critical-capable server, and it runs more than 65 percent of all Web sites on the Internet. The PHP component actually sits inside Apache, and you use Apache and PHP together to create your dynamic pages.
- **MySQL.** MySQL provides the data-storage side of the L.A.M.P. system. With MySQL, you have access to a very capable database suitable for running large and complex sites. Within your Web application, all your data, products, accounts, and other types of information will reside in this database in a format that you can easily query with the SQL language.
- **PHP.** PHP is a simple and efficient programming language that provides the glue for all the other parts of the L.A.M.P. system. You use PHP to write dynamic content capable of accessing the data in the MySQL database and some of the features that Linux provides. [5]

Each of the core components of L.A.M.P. has a major commercial or nonprofit foundation behind it:

- Linux: Red Hat, Novell
- Apache: Covalent
- MySQL: MySQL AB

PHP/Python/Perl: Zend, Python Software Foundation, the Perl Foundation[6]

2.2 History of L.A.M.P. Architecture

Michael Kunze coined the acronym L.A.M.P. in an article for the German computing magazine c't in 1998 (12/98, page 230). The article aimed to show that a bundle of free software could provide a viable alternative to commercial packages.

Knowing about the IT-world's love of acronyms, Kunze came up with L.A.M.P. as a marketing-like term to increase the popularity of free software [7]

The term was coined in defense of free software. Since then it has been widely proved that his statement is correct . The L.A.M.P. stack is in extremely wide use today all over the world

In Kunze's coining of the term, the "P" in L.A.M.P. stood for PHP. In practice, however, this letter has come to stand for any of the three scripting languages: PHP, Perl, or Python. Scripting languages are ideally suited for Web site development because they are easy to learn and highly oriented towards processing text. PHP, Perl, and Python have evolved into full-fledged languages over the past few years. [8]

For Web applications, L.A.M.P. has been proven faster, cheaper, more flexible, and easier than any alternative. There is a strong push to L.A.M.P. by vendors ranging from IBM to Oracle to numerous startups—and these vendors are adding enterprise-grade capabilities and management to L.A.M.P.. There is no question that L.A.M.P. is not a passing trend, but now entering the mainstream as a serious contender to J2EE and .NET.

2.3 How does L.A.M.P. work ?

The 4 technologies together refer to a server environment built to output to/for the internet. Linux is an Operating System, Apache is a Web Server, MySQL is a database engine and PHP, PERL and Python are all programming languages geared towards the internet.

L.A.M.P. is singularly focused towards Web applications. The architecture is very straightforward, as illustrated in Figure 1. Linux forwards HTTP connections to Apache, which serves static content directly from the Linux kernel. Dynamic pages are forwarded by Apache to PHP, which runs the PHP code to design the page. Database queries are sent to MySQL through PHP. Administration is commonly handled through phpMyAdmin, and every major enterprise management system can manage Apache and Linux.

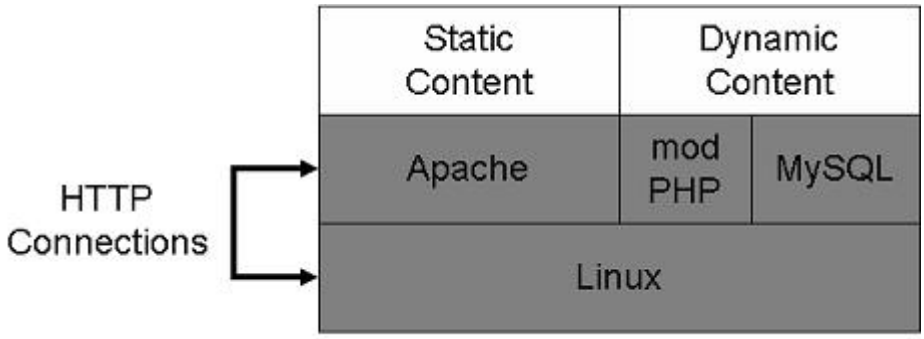


Figure 1: L.A.M.P. Architecture [9]

There are numerous resources on the Web explaining how to quickly build a L.A.M.P. application. Most developers are productive within hours of installing the L.A.M.P. stack.

Most of the Linux distributions have already contained all four L.A.M.P. components, That means, once your installation of the Linux operating system is finished, Your web server (Apache) is ready to use , the database MySQL is already up and running . You can start to develop your PHP scripts.

2.4 Other variations

On the other hand Linux isn't a necessary standard because development often occurs on a Windows platform running Apache, MySQL, and either PHP, Python, or Perl (this configuration is called "WAMP").

Following is a list of different acronym according different operating system:

- **AMP**, omitting the operating system
- **AMPS**, using SSL or Solaris
- **BAMP**, using BSD
- **FAMP**, using FreeBSD specifically
- **MAMP**, using Mac OS X
- **NAMP**, using NetBSD or Novell NetWare
- **NAPP**, using NetBSD and PostgreSQL

- **PAMP**, using PC-BSD
- **PUMA**, using Unix
- **SAMP**, using Solaris
- **WAMP**, using Microsoft Windows [10]

2.5 Who are Using L.A.M.P.?

Nowadays L.A.M.P. has been treated by tons of developers as „Golden Combination“. Many big companies (IBM,Intel,HP,Sun Microsystem ,SAP etc) are controbuting significantly to many L.A.M.P. based open-source projects.

In the mean time , millions of developers are adding values to these open-source projects .The characteristic of the open-source software makes it possible that developers can change the source code as they need .

Although those projects are each under distinct open source licenses, the code that a developer writes for his or her applications does not need to be released as open source. However, if the software contains modifications to some of the projects (including the Linux kernel and the MySQL database), those modifications need to be published under the GPL license [11].

L.A.M.P. components are used more and more widely on their own, as well as in L.A.M.P. installations. In just the past year, IBM and Oracle endorsed PHP; Sun announced integration of PHP and Java. Linux is perhaps the biggest success story among the L.A.M.P. components, but the usage of PHP has also grown dramatically over the past five years..[12]

Some of the L.A.M.P. components' usage and marketing share :

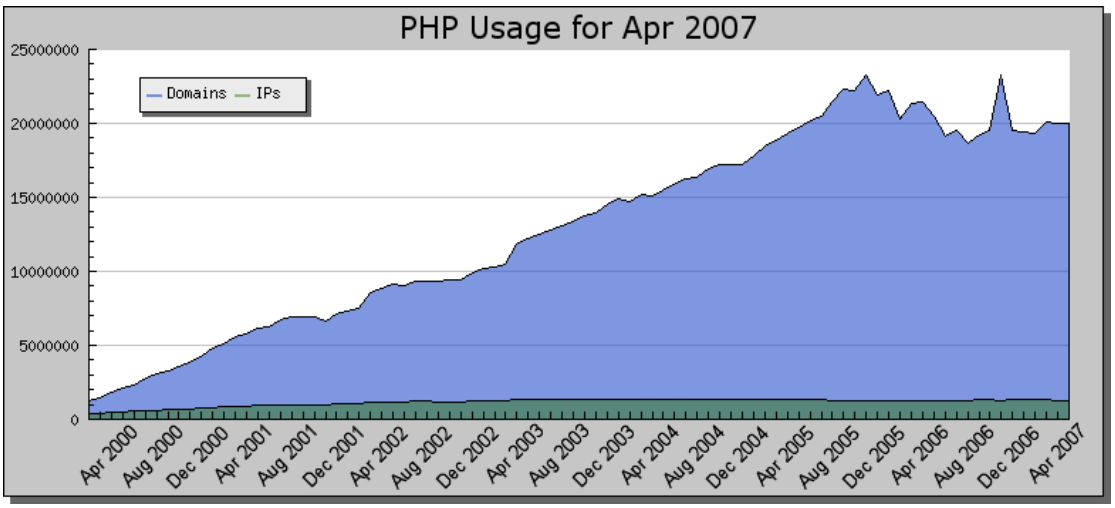


Figure 2: PHP Usage on the Rise 2000-2007 [13]

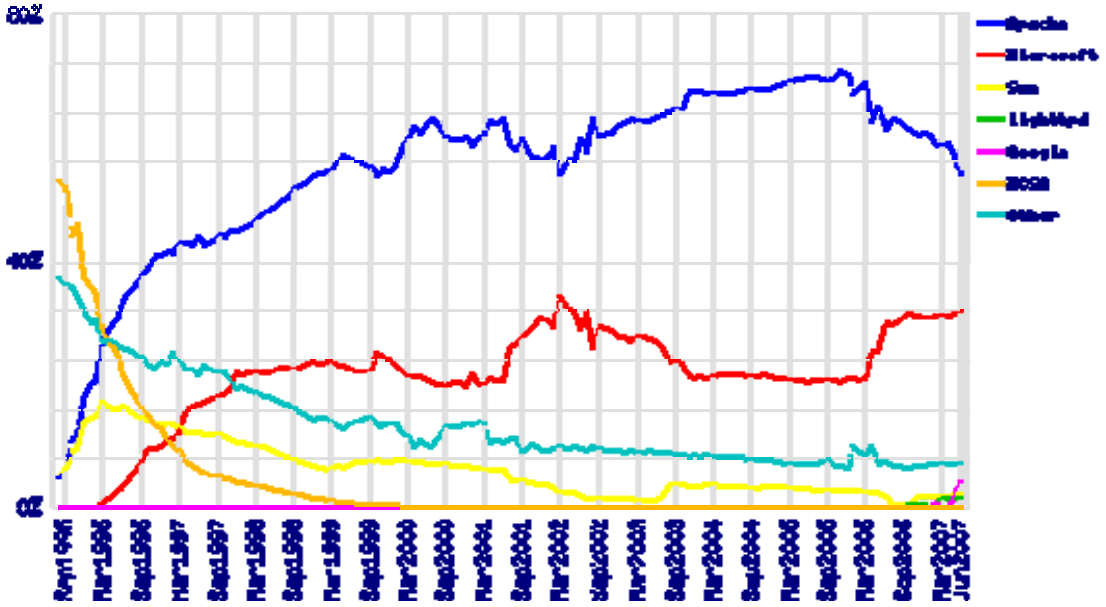


Figure 3: Apache Market Share [14]

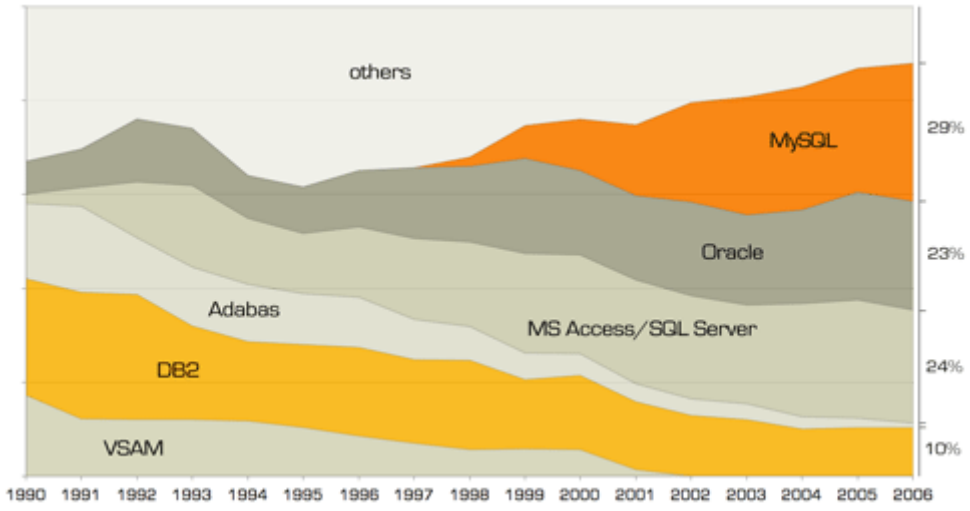


Figure 4: MySQL Market Share [15]

Chapter 3

RSS Syndication

3.1 What is RSS?

RSS is a Web content syndication format. Its name is an acronym for **R**eally **S**imple **S**yndication.

RSS is a dialect of XML. All RSS files must conform to the XML 1.0 specification, as published on the World Wide Web Consortium (W3C) website.[16]

These little orange RSS  and XML  buttons are getting more popular everywhere in the internet. In most of the cases, if you click on those buttons you will see some codes, they are XML files.

RSS has accumulated a number of meanings, from "RDF Site Summary," to "Rich Site Summary," to "Really Simple Syndication." I like the last term best as I think it best describes RSS as a service. RSS might just as easily be called XML syndication because it is based on the XML language.

RSS syndicates the content to a RSS reader. With RSS readers or feed readers, you can aggregate all of your information sources and other content into one program, creating a single view for this information. A good analogy is the capability to create your own custom newspaper that includes articles from different sources with the added ability to choose which subjects you read from each source.

RSS encapsulates metadata (information about data) about the content. This metadata allows an RSS reader to understand such things as the type of content (text versus multimedia), the date of publication, and so forth.

3.2 Benefits of RSS

In today's world, people are getting information though hundreds of e-mails and dozens of Web sites to find the information for which you are looking.

RSS aggregators simplify this task by bringing these many different data sources together into one view.

RSS feed readers allow you to read information sources or blogs in a single application or Web site. The reader will aggregate all of the feeds that you choose and list them in a simple-to-read fashion. The benefits of having a single location to turn to for your news and information content are numerous:

- You visit fewer Web sites
- Your news is on demand -- ready and waiting when you want it. If you can't get to it for a few days, all your news will still be there for you (unlike traditional sites where news expires off the front page each day).
- No e-mail newsletters clutter your mailbox.
- You customize the news and content that comes to you. you can have individual subjects 'delivered to your door' with less intrusion than an e-mail newsletter.
- You can ignore articles or channels that are not of interest to you .
- You stay up-to-date on any news by topic, industry, or subject area.
- You don't have to check back for new postings on the news site. The feed readers deliver content to you automatically.

Content delivery on the Internet now takes a new form. Most people turn to countless sources of information these days. Individuals might look to different portals for news, press releases, industry analysis, product reviews, and so on. Traditionally, this process was requiring a lot of time for users as they visited each site, poking around for new information before moving on to another

Feed readers aggregate all of this content into a simple, easy-to-view application, and do not intrude on your productivity tools, such as e-mail. Most feed readers have the same look and feel as e-mail applications or newsgroup readers, with folders on the left and content on the right. The folders on the left might represent different Web sites or different information channels. If you are an active blog reader, the folders represent each blog. Tt is inefficient to revisit a blog site multiple times a week to seek out when an author has posted new content; it's best to have that content delivered to you. This same principle applies for newsgroups and community forums. [17]

3.3 RSS History and specifications

Tuesday, April 6, 2004, there are a lot of folk legends about the evolution of RSS. Here's the scoop, the sequence of events in the life of RSS, as told by the designer of most of the formats.

- scriptingNews format, designed by DW at UserLand. 12/27/97.

- RSS 0.90, designed by Netscape, for use with my.netscape.com, which also supported scriptingNews format. The only thing about it that was RDF was the header, otherwise it was plain garden-variety XML. 3/15/99.
- scriptingNews 2.0b1, designed by DW at UserLand, enhanced to include all the features in RSS 0.90. Privately DW urged Netscape to adopt the features in this format that weren't present in RSS 0.90. 6/15/99.
- RSS 0.91, designed by Netscape, spec written by Dan Libby, includes most features from scriptingNews 2.0b1. "We're trying to move towards a more standard format, and to this end we have included several tags from the popular <scriptingNews> format." The RDF header is gone. 7/10/99.
- UserLand adopts RSS 0.91, deprecates scriptingNews formats. 7/28/99.
- The RSS team at Netscape evaporates.
- UserLand's RSS 0.91 specification. 6/4/00.
- RSS 1.0 published as a proposal, worked on in private by a group led by Rael Dornfest at O'Reilly. Based on RDF and uses namespaces. Most elements of previous formats moved into modules. Like 0.90 it has an RDF header, but otherwise is a brand-new format, not related to any previous format. 8/14/00.
- RSS 0.92, which is 0.91 with optional elements, designed by DW at UserLand. 12/25/00.
- RSS 0.93 discussed but never deployed. 4/20/01.
- MetaWeblog API merges RSS 0.92 with XML-RPC to provide a powerful blogging API. 3/14/02.
- RSS 2.0, which is 0.92 with optional elements, designed by DW, after leaving UserLand. MetaWeblog API updated for RSS 2.0. While in development, this format was called 0.94. 9/18/02.
- RSS 2.0 spec released through Harvard under a Creative Commons license. 7/15/03.[18]

The RSS language specifications have been created by different individuals and groups: David Winer (RSS 0.92, 2.0), Dan Libby of Netscape (RSS 0.9, 0.91), and the RSS-DEV, a working group continuing Libby's works (RSS 1.0). Many attribute the creation of XML syndication to David Winer from a pre-RSS format. RSS 2.0 is not simply RSS 1.0 with additional features. Most, if not all feed readers support each specification.

A similar specification, Atom is under development by the Internet Engineering Task Force (IETF) in the hopes of creating a universally adopted specification. For more information on Atom.[19]

As a content author, choosing a standard is not difficult. Depending on your content, you should use a spec that will support the metadata you wish to distribute (such as enclosures for podcasts). Additionally, you may choose to offer you content on multiple specifications, such as RSS 2.0.

Listing 1. A sample RSS 2.0 feed

```
<?xml version="1.0"?>
<rss version="2.0">

<channel>
    <title>Feed Title</title>
    <link>http://yourwebsite.com/</link>
    <description>Feed Description</description>
    <language>en-us</language>
    <pubDate>Mon, 03 Jan 2005 12:00:00 GMT</pubDate>

<item>
    <title>Article Title</title>
    <link>http://yourwebsite.com/articlelink.html</link>
    <description>Your content included here.</description>
</item>

</channel>
</rss>
```

3.4 feed readers

You can read RSS feeds in man ways: everything from stand-alone applications, to Web-based portals, to support directly within your Web browser. Each is discussed below. Some readers are free, while others offer advanced functionality at a cost. Many people get confused by the little RSS orange button. For instance, it takes you to a page of machine code and does not open in an appropriate RSS feed reader. This is the XML code; you just need to add the browser's address location to a RSS feed reader. Choices on how to add the feeds to your feed readers (and some Web browsers, such as Firefox®) include:

- When you end up on a RSS page with XML tags all over the place, copy the URL at the top of your screen and paste it into your feed reader. More on this in the feed reader section.
- Alternatively, you can right-mouse-click on the icon and select "Copy Link Location" (Firefox users) or "Copy Shortcut" (Microsoft® Internet Explorer® users.)

3.4.1 Web browser

If you run the Firefox browser, then your browser supports RSS feeds natively. Other browsers that support RSS aggregation out-of-the box include Opera and Apple's Safari®. The newest version of Microsoft Internet Explorer is slated to have RSS support.

In Firefox, you can subscribe to a feed easily -- just click on the **Live Bookmark** icon  in the location bar of your browser and select **Subscribe to <website channel name>**. A new window pops up to prompt you for a Bookmark name.

3.4.2 Desktop applications

Stand-alone applications are programs that you install on your computer just like an e-mail program such as Outlook®. The feed reader applications are very lightweight and usually run in the background. Most feed readers will have some sort of notification system through a sound or pop-up window in the bottom right corner.

- **SharpReader** [Free]
- **FeedDemon**.

3.4.3 Web Services

A few services exist that allow you to aggregate content on an external Web portal. A very popular free service with a large user base is **Bloglines.com**. Bloglines offers these features:

- Viewable on any platform, including Windows, Linux™, or Mac.
- Auto-discovery of RSS feeds for a given domain (when you don't know the RSS address).
- Access to your subscriptions from any computer. Reading history stays persistent across computers, ensuring that you only view unread content.
- Mobile version for Internet-enabled mobile devices.
- Strong Firefox support: To add a new RSS feed, simply right click on a RSS icon or Web page (with plug-in).
- Hides channels with no new content (optional).
- Allows you to save a post for later retrieval.

- Includes a blogging account.
- Subscriptions can be made public, so other users can read your subscriptions; this is called a blogroll.
- Shows the number of other users subscribed to the same channel.

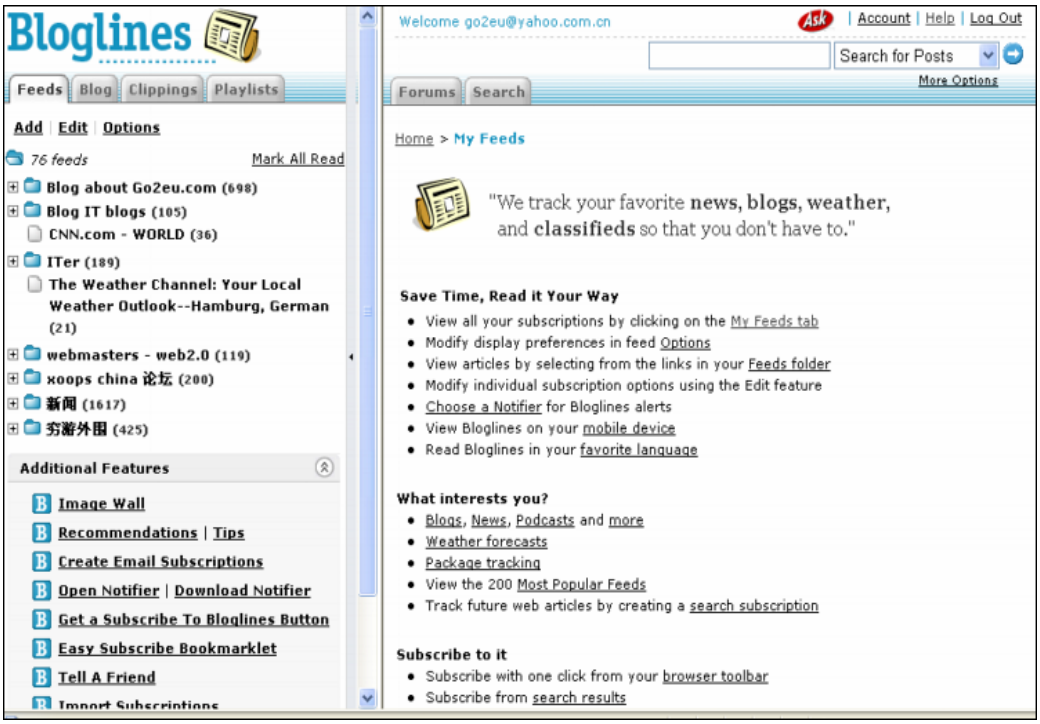


Figure 5. Screenshot of Bloglines.com

Some other web services which supports feed aggregation

3.4.4 Handhead Application

Figure 6. logo of Spb Inside

Spb Insight is a news reading Pocket PC program that gives a rich offline experience along with an easy interface and full RSS/Atom support. With Spb Insight you'll

always have all the latest information, right in your pocket, with images and whole articles, yet fully optimized for small screen viewing.

- Full website news
- Full RSS/Atom support (0.9x, 1.0, 2.0)
- Offline reading with pictures
- Easy to use interface
- One hand operation
- Article thumbnails
- No desktop component needed [20]

Figure 7. Screenshot of spb inside [21]

Chapter 4

A Glance to CMS

4.1 The Definition of "Content Management System" (CMS)

A web site's content is a combination of different components like texts, graphics, images, scripts, embedded files such as flash animations, audio/video streams or downloadable files.

Content management systems (CMS) are computer software systems for organizing, displaying and facilitating collaborative creation of this content.

This can be achieved by storing the content and the layout in different resources and dynamically merge them together to the final document.

4.2 Advantages of CMS

CMS facilitate the collaborative creation of websites. People can concentrate on the content while others care for the template to present the content. Also many CMS provide systems to enable users to add or modify content via their web browser

CMS make it easier to display the same content in different ways, like a normal view for web browsers and a printer friendly view

CMS make it easier to modify the layout of a website as one only has to modify the template at a single source instead of having to modify each single page to reflect the change

CMS often can automatically create additional content like menus, sitemaps etc.

CMS often provide methods to find content, for example by providing search functionality on the content [22]

4.3 Web content management systems

You need a web CMS most when several of these conditions apply...

- Site has tons of pages and it's very difficult to keep track of them all.
- There are a lot of people writing pages e.
- They don't want to learn how to write HTML.
- The alternative (software such as Dreamweaver) is too complex and expensive for the number of users you have.
- You don't trust them enough to publish their own pages without approval.

A web content management system is software for web site management.

4.4 Commercial CMS

Subdreamer CMS is a powerful CMS with a huge community backing it up. Monthly new skins, countless plug-ins, and a vast amount of features makes this a must have CMS.

PHPCow Article, News Publishing Content Management System. Offers script for creating magazine, newspaper, portal sites.

Bitrix Site Manager is a powerful, secure, full-featured content management solution for creating and managing corporate web sites. The product is delivered in source code.

4.5 L.A.M.P. based free / open-source web CMS

There are over thousands of different web CMSs available, following are just some of them.

- **Mambo**
- **PHP Nuke**
- **Typo3**
- **WordPress**
- **XOOPS**

Chapter 5

XOOPS Dynamic Web CMS

As we discussed in previous chapter ,the number of open source CMSs is remarkable . Finally we choose XOOPS Dynamic Web CMS.

Figure 8: logo of XOOPS dynamic web cms

5.1 XOOPS Introduction

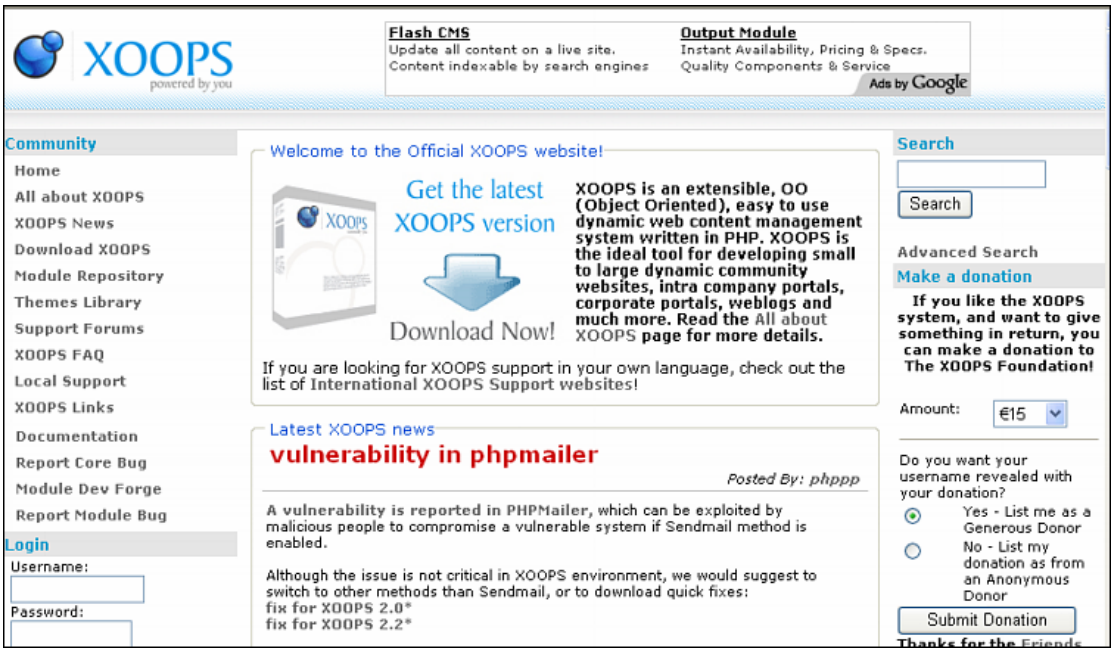


Figure 9: Homepage of XOOPS dynamic web CMS project

XOOPS is a program that allows web administrators and users to easily create dynamic websites with great content and many outstanding features. It is an ideal tool for developing small to large dynamic community websites, intra company portals, corporate portals, weblogs and much more. It can be installed on an Internet host with a PHP-capable web server (e.g., Apache) and a database (e.g., MySQL).

XOOPS is released under the terms of the GNU General Public License (GPL) and is free to use and modify. It is free to redistribute as long as you abide by the distribution terms of the GPL.

XOOPS is an acronym of e**X**tensible **O**bject **O**riented **P**ortal **S**ystem.

A lite XOOPS can be used as a personal weblog or journal. For this purpose, one can do a standard install, and use its News module only. For a medium site, you can use modules like News, Forum, Download, Web Links etc to form a community to interact with your members and visitors. For a large site as an enterprise one, you can develop your own modules such as eShop, and use XOOP's uniform user management system to seamlessly integrate your modules with the whole system.

The standard pronunciation of XOOPS should follow English rules, and you should pronounce it as **[zoo'ps].**

5.2 Key features of XOOPS

Database-driven
XOOPS uses a relational database (currently MySQL) to store data required for running a web-based content management system.

Fully Modularized
Modules can be installed/uninstalled/ativated/deactivated with a click using the XOOPS module administration system.

Personalization
Registered users can edit their profiles, select site themes, upload custom avatars, and much more!

User Management
The ability to search for users by various criteria, send email and private messages to users through a template-based messaging system.

Supported World-wide
XOOPS was created and is maintained by a team of several hard-working volunteers

working from all over the world. The XOOPS community has more than dozen official support sites around the world for support of non-English speaking users.

Multi-byte Language Support
Fully supports multi-byte languages, including Japanese, Simplified and Traditional Chinese, Korean, etc.

Versatile Group Permissions System
Powerful and user-friendly permissions system which enables administrators to set permissions by groups.

Theme-based sinkable interface
XOOPS is driven by a powerful theme system. Both admins and users can change the look of the entire web site with just a click of a mouse. There are also over 60 themes available for download!! [24]

5.3 XOOPS Installation

Installation of XOOPS is straightforward and easy to follow. The installation wizard walks through every step of the process. With proper server permissions and configurations, only a database username and password are needed to start using XOOPS.

Figure 10. XOOPS installation wizard

The whole installation wizard process will be included in **Appendix** section of this thesis

Chapter 6

XOOPS Module Design

This Chapter discuss how to build your own module under XOOPS CMS .

6.1 Official Default Modules

XOOPS currently ships with a number of modules developed and maintained by the XOOPS Module Development team. Members of the XOOPS community have also developed by a wide variety of third-party modules.

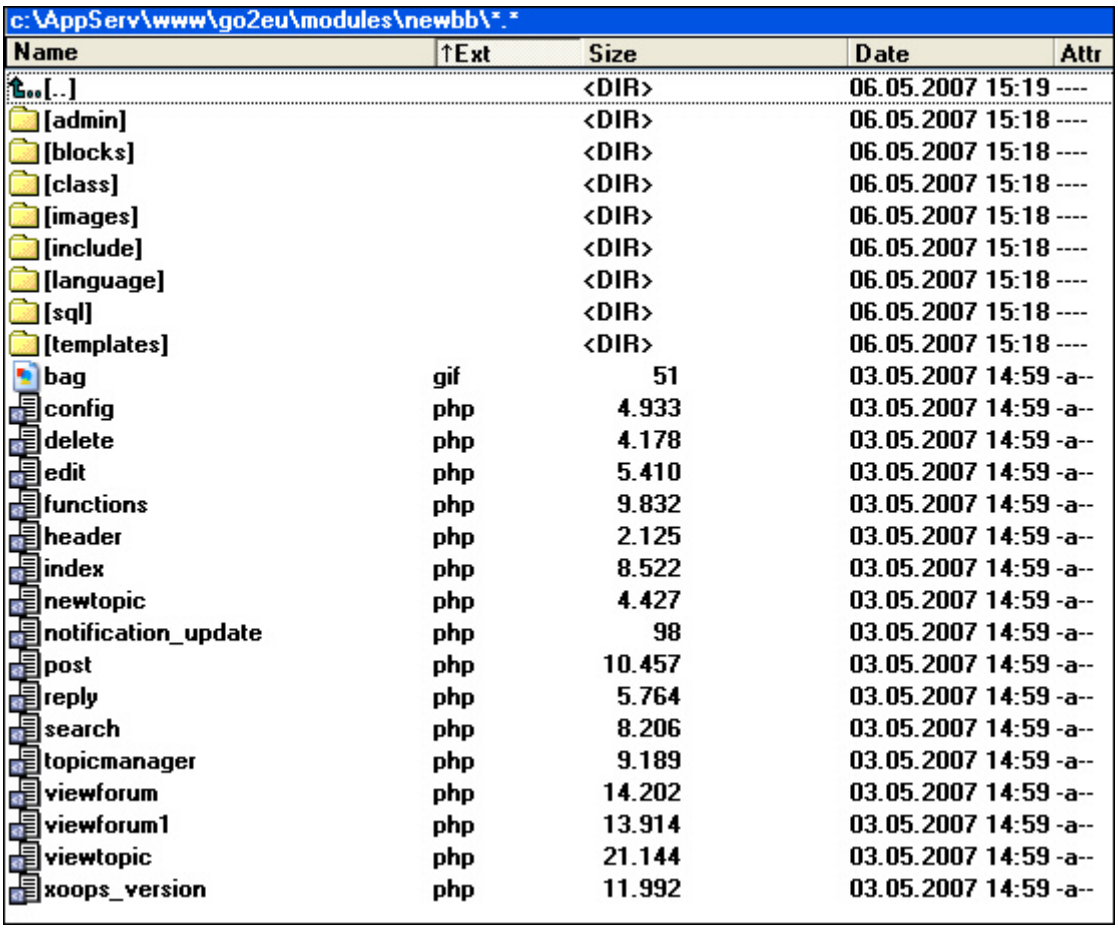
The following are the present official modules:

- News
- NewBB
- Polls
- Web Links
- Downloads
- Headlines
- FAQ
- Partners
- Members
- Contact Us

These default modules can fulfill most of the requirements of a dynamic website, if we want our own modules, we have to build it .As I mentioned in previous chapters, and XOOPS is easy to be extended by building our own modules to meet our specific requirements.

6.2 XOOPS general module standards

6.2.1 File Structure



c:\AppServ\www\go2eu\modules\newbb*.*

Name	↑Ext	Size	Date	Attr
[..]		<DIR>	06.05.2007 15:19	----
[admin]		<DIR>	06.05.2007 15:18	----
[blocks]		<DIR>	06.05.2007 15:18	----
[class]		<DIR>	06.05.2007 15:18	----
[images]		<DIR>	06.05.2007 15:18	----
[include]		<DIR>	06.05.2007 15:18	----
[language]		<DIR>	06.05.2007 15:18	----
[sql]		<DIR>	06.05.2007 15:18	----
[templates]		<DIR>	06.05.2007 15:18	----
bag	gif	51	03.05.2007 14:59	-a--
config	php	4.933	03.05.2007 14:59	-a--
delete	php	4.178	03.05.2007 14:59	-a--
edit	php	5.410	03.05.2007 14:59	-a--
functions	php	9.832	03.05.2007 14:59	-a--
header	php	2.125	03.05.2007 14:59	-a--
index	php	8.522	03.05.2007 14:59	-a--
newtopic	php	4.427	03.05.2007 14:59	-a--
notification_update	php	98	03.05.2007 14:59	-a--
post	php	10.457	03.05.2007 14:59	-a--
reply	php	5.764	03.05.2007 14:59	-a--
search	php	8.206	03.05.2007 14:59	-a--
topicmanager	php	9.189	03.05.2007 14:59	-a--
viewforum	php	14.202	03.05.2007 14:59	-a--
viewforum1	php	13.914	03.05.2007 14:59	-a--
viewtopic	php	21.144	03.05.2007 14:59	-a--
xoops_version	php	11.992	03.05.2007 14:59	-a--

Figure 11: XOOPS module file structure

Below is a basic structure of a module in XOOPS, not all the files are necessary for every module.

XOOPS_version.php

The most important file for the module, which contains module's configurations,a detailed description about this file is included in **Appendix** section of this thesis.

header.php

Inclusion of required XOOPS core files

index.php

It displays the main module content.

/admin

All the admin related files for the module should locate in this directory. There must be a restricted access to this directory.

Files inside this directory

- admin_header.php

  Permission checks, inclusion of mainfile.php and language files.

- index.php

  Module administration contents.

- menu.php

  This file includes variables used for creating popup admin menu for the module.

/language

This directory should at least contain an english directory.

Files inside this directory

- english/main.php

  Defines language constants used in module main contents.

- english/admin.php

  Defines language constants used in module admin sections.

- english/modinfo.php

  Defines language constants used in XOOPS_version.php

- english/blocks.php

  Defines language constants used in blocks/blocks.php

/blocks

This directory contains files for module blocks.

Files inside this directory

- blocks.php

/sql

SQL dump files used for module installation belong to this directory.

Files inside this directory

- mysql.sql

  mysql dump file

- postgresql.sql

  postgreSQL dump file

/images

Any image files used in the module.

/cache

Files that are created and updated dynamically belong to this directory. The permission of this directory should be set 666 or 777 for UNIX/LINUX servers.

/templates

all templates(HTML files) used for displaying module.

/templates/blocks

all templates(HTML files) used for displaying module blocks.

6.2.2 Root files to be included by a module

A module file, which should be accessed via a browser URL should be in the modules/modulename folder.

The first thing a module page should do is include the root file "mainfile.php". Mainfile.php includes "include/common.php" which will:

- Setup the connection to the database
- Login the user if not anonymous

- Include the XOOPS API functions
- Retrieve the current module ($XOOPSModule)
- Retrieve the module's configuration options ($XOOPSModuleConfig)
- Include relevant language files
- Checks that the site is not closed - or if it is, that the user can access it
- Checks if the user has access to the current module

The next step is to include the root file header.php, which will:

- Create the Smarty object
- Check the page cache settings
- either display the cached page or
- build the blocks to be displayed on the page

Now that the main parts of XOOPS have been included, the file can process the data to be shown on the page and assign variables to Smarty if applicable. This can be done by any php code

6.2.3 Module admin page

Admin pages should be in the modules/modulename/admin folder and have this structure:

```
include ('../../../include/cp_header.php');
XOOPS_cp_header();

XOOPS_cp_footer();
```

6.2.4 XOOPS module blocks

There are 2 main files necessary for a block - the php file that gathers the data for displaying in , and the html (template) file that controls the layout of the block.

Here is the folder/file layout necessary for adding blocks - modulename/

- blocks
- templates/blocks

Following the naming conventions, the filename for the block is modulename_block_blockname.php, and it should contain 2 functions - one to query the data, and one to edit the options for the block (if any). Here is one of the important tips -> the function to display the data must return a variable named $block in order for the data to be passed to the template.

Another important thing is the block may be displayed in several different modules, so any includes and links should be absolute links - relative links will vary, depending on which module is displaying.

6.2.5 Some very usefule XOOPS functions

There are some very usefule XOOPS functions available to be used by module design.

- XOOPS_confirm
- XOOPS_getLinkedUnameFromId, returns username within <a href> tags to userinfo.php or "anonymous"
- checkEmail($email, bool $antispam = false), checks if an email is a valid format (x@x.x[.x]) and returns true/false - if 2nd parameter is true, it will return an email address in an antispam format (someone at domain dot TLD)
- formatTimestamp($time, $format="l", $timeoffset=""), formats a UNIX timestamp into text, using 's', 'm', 'l' (short, medium, large - defined in language/yourlanguage/global.php), 'rss' or 'mysql' format as 2nd parameter and with an offset if supplied.
- XOOPS_refcheck, to check if the page request has come from external site

6.2.6 Manipulating database

Querying date

Generally, query the database to get a result and then operate on the returned data:

```
    $sql = 'SELECT * FROM'.$this->db->prefix('table').'WHERE ... ORDER
BY...';
    $result =& $this->db->query($sql);
    while ($array = $this->db->fetchArray($result)) {
        //do something useful here
    }
```

Useful Functions

- prefix($tablename) - adds XOOPS_DB_PREFIX and _ to $tablename
- &query($sql,$limit,$start) - returns a reference to the query results using the SQL statement $sql, limited to $limit rows, starting at row $start. This result may be read using functions such as fetchRow and fetchArray.
- fetchArray($result) - returns array (name => value) from $result and removes the row from $result.
- fetchRow($result) - returns enumerated array (X => value, where X starts at 0 and increments for each field) from $result and removes the row from $result.
- getRowsNum($result) - returns number of rows in $result
- getFieldsNum($result) - returns number of fields in $result
- getFieldName($result,$offset) - returns name of field based on $offset from first field in $result
- getFieldType($result,$offset) - returns type of field based on $offset from first field in $result
- quoteString($str) - Returns escaped string text with single quotes around it to be safely stored in database

Creating tables

XOOPS creates database tables for module when the module is first installed. It does this by executing instructions found in module .sql file. The location of module's sql file is specified in module's XOOPS_version.php file :

$modversion['sqlfile']['mysql'] = "sql/mysql.sql";

The sql file should contain a list of CREATE TABLE commands, needed module. like:

```
CREATE TABLE 'modulename_tablename' (

'id' int(12) NOT NULL auto_increment,
'varcharfield' varchar(120),
PRIMARY KEY ('id', 'varcharfield'),

);
```

A simple way of creating sql file is to create a 'dump' file of database, with only the tables necessary for module included. In phpMyAdmin this is done by clicking the 'export' tab.

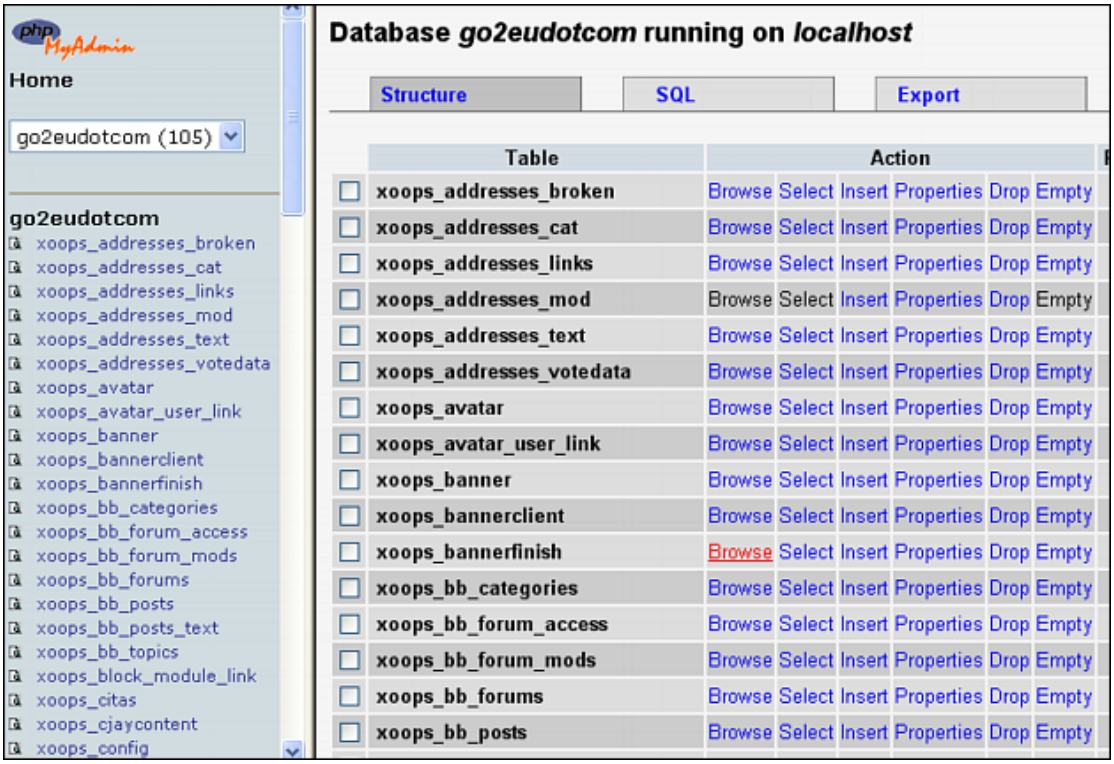


Figure 12: Screenshot of phpMyAdmin

More MySQL reference manual for CREATE TABLE syntax:
http://dev.mysql.com/doc/mysql/en/create-table.html

Adding records to the tables at install time

If some records need to be inserted to database table during installation of the module .the onInstall function will be used to achieve that. This function is called directly after the tables have been created from the sql file. The location of the file containing onInstall function is given in the XOOPS_version.php file:

$modversion['onInstall'] = "path/to/file/with/onInstall/function";

Finish the function with returning either true or false for success or failure, respectively.

Altering tables

There is also an onUpdate function available. This can be used for altering the database tables for the module when the structure has been changed in a new version. Module's XOOPS_version.php is used to specify the location of the file that contains the onUpdate function.

$modversion['onUpdate'] = "path/to/file/with/onUpdate/function";

The function takes two parameters and should be in the form of :

function XOOPS_module_update_[dirname](&$module, $old_version) {

...

}

The $module parameter is a reference to the (updated) XOOPSModule object.

The second parameter is the module's old version.

Finish the function with returning either true or false for success or failure.

Deleting tables

Tables for module should be deleted when your module is uninstalled. They must all be listed in the XOOPS_version.php file.

$modversion['tables'][0] = "modulename_category";
$modversion['tables'][1] = "modulename_article";

Using Index

Database indexes are used to find rows with specific column values. Without an index, MySQL has to start with the first record and then read through the whole table to find the relevant rows. This will slow down your dynamic site .The larger the table, the more this costs. If the table has an index for the columns, MySQL can quickly determine the position without having to look at all the data. This can cut the query time dramatically, which also means the module page will load faster.

Indexes improve the performance of SELECT queries to quickly find the rows that match WHERE or ORDER BY clauses.

We will always make the column which is used most often as the first index key.

MySQL stores row data and index data in separate files. So for every index we create, the size of the database will be increased. Therefore, we need to think carefully about

how often we would use the index before creating one and only create the indexes that we really need.

Creating a table with an index

we create an index at the same time we create a table in the mysql.sql file. MySQL accepts two commands for creating an index: ‘KEY’ and ‘INDEX’. Either one could be used as they perform exactly the same function.

Listing 2 : example SQL statements for creating tables

```
CREATE TABLE `example_table` (

`eg_id` int(12) unsigned NOT NULL auto_increment,
`eg_integer` int(10) unsigned NOT NULL default '0',
`eg_string` char(100) NOT NULL default '',
`eg_date` int(12) unsigned NOT NULL default '0',
PRIMARY KEY (`eg_id`),
KEY `eg_date` (`eg_date`,`eg_string`),
KEY `eg_integer` (`eg_integer`)

) TYPE=MyISAM;

CREATE TABLE `example_table` (

`eg_id` int(12) unsigned NOT NULL auto_increment,
`eg_integer` int(10) unsigned NOT NULL default '0',
`eg_string` char(100) NOT NULL default '',
`eg_date` int(12) unsigned NOT NULL default '0',
PRIMARY KEY (`eg_id`),
INDEX `eg_date` (`eg_date`,`eg_string`),
INDEX `eg_integer` (`eg_integer`)

) TYPE=MyISAM;
```

Adding an index to an existing table

ALTER TABLE example_table

ADD INDEX [index_name] (index_col_name,...)

Chapter 7

RSS Syndication Module

This chapter describes how to use PHP to provide RSS Syndication to XOOPS modules.

7.1 Use RSS to the Website

As discussed in Chapter 3 , RSS syndication technology has many advantages . There is no reason not to let XOOPS have this function.

RSS can be a huge benefit to website users, especially if they value opinions or news listed on website. Without having to return frequently to check if we have updates, they will know exactly when we update or add content, allowing them to save time and effort, and they won't miss anything either!

Content generation is not a problem at all, we have already XOOPS dynamic web CMS up and running, the only thing needs to be done is to pull date out of the database and produce RSS feeds on a time basis and let others know that we have this feature.

7.2 Let RSS feed to be found quickly by others

We need to put RSS standard icon  to the suitable position of our website. But that is not enough. In this case we can only let people know the RSS feed exists,
How to tell Firefox and Microsoft Internet Explorer, as well as other readers discussed in Chapter 3 about our RSS syndication ability?

We should add the following tag to the top of your home page to solve this problem:

Listing 3 : HTML code for feed readers

```
<link rel="alternate" type="application/rss+xml"
href="URL_OF_FEED" title="FEED_TITLE" />
```

7.3 A detailed look at RSS 2.0 standard format

The RSS standard defines and contains the content of a feed. These feeds can be from any data source, defining Internet documents and in a very basic sense, make up a list of links and their descriptions.
Look at the sample RSS 2.0 document

Listing 4: A sample RSS 2.0 document

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- generator="RSSxl/2.0" -->
<rss version="2.0">
   <channel>
      <title>HAW-Newsletter</title>
      <link>http://newsletter.haw-hamburg.de/</link>
      <description>Weekly News about HAW Hamburg</description>
      <language>en</language>
      <docs>http://www.wotzwot.com/rssxl.php</docs>
            <image>http://newsletter.haw-hamburg.de/images/HAWLogo1.gif</image>
      <item>
         <title>Fachtagung zum Thema Onlineberatung</title>
         <link>http://newsletter.haw-hamburg.de/</link>
         <description>Der Verband alleinerziehender Muetter und Vaeter feiert am 15. Juni sein
35-jaehriges Jubilaeum an der HAW Hamburg mit einer Fachtagung zum Thema
Onlineberatung</description>
      </item>
                  <item>
         <title> Studierende von heute - Kollegen von morgen </title>
         <link>http://newsletter.haw-hamburg.de/</link>
         <description>Am Dienstag, den 19. Juni, findet die Ausbildertagung des Departments
Information statt. Die Tagung steht dieses Jahr unter dem Motto "Studierende von heute -
Kollegen von morgen". </description>
      </item>
   </channel>
</rss>
```

The first child object of the XML formatted document is the definition of a <channel>. A channel is simply the feed itself and its associated information. In general most of RSS feeds have one channel object, but we could have several if needed. The objects: title, link and description are required by the channel object. They define the basic descriptive information about the feed. The optional objects are: language, copyright,

managingEditor, webMaster, pubDate, lastBuildDate, category, generator, docs, cloud, ttl, image, rating, textInput, skipHours, and skipDays.
A channel can contain an unlimited number of items. All elements of the ITEM element are optional, however, at least one title or description are needed to validate the element. The elements are: title, link, description, author, category, comments, enclosure, guide, pubDate, and source.

7.4 Use PHP to generate feed(XML files)

Now we know what the RSS data format is, we could look at the data we want to send out to our visitors, and put it in that format. PHP has some powerful RSS and XML handling features to speed our development. Like many of the common Web standards, PHP has a number of great functions ready for use in this application.

These functions that make short work of XML both in and out of an application.
First we want to take information from a locally stored data source, in our case the XOOPS content management system, and put that out as a feed to users. We will need to get this data, format it into an RSS object.

We will use XML_RSS to get and handle these feeds.

XML_RSS() function is a PEAR package to help you get through the more complex tasks of interpreting an XML RSS file more easily. PEAR is an open source library of PHP functions which is free for our use and under continual development.

XML_RSS() is simply a function, which gives the location of an RSS feed, will load the XML of the feed into an array, ready for use in our PHP application. The elements of the array will have named keys, associated with the elements and attributes of the RSS file read.[25]

Introduction to XML_RSS

Requirements for running XML_RSS
Example for the usage of XML_RSS

XML_RSS::XML_RSS Constructor
XML_RSS::getStructure Get complete structure of RSS file
XML_RSS::getChannelInfo Get information about current channel
XML_RSS::getItems Get items from RSS file
XML_RSS::getImages Get images from RSS document
XML_RSS::getTextinputs Get text input fields from RSS document [26]

7.5 Get the data out of the database

We pull data out of a database using "" and format it into an RSS feed. We will set it up so that it looks for the most recent additions to dataset whenever the RSS feed is called upon and returns a fresh RSS to the requester.

The feed can come from any data source, in our case, data are pulled out from MySQL database, but in the end we need to make sure that there is enough data that people receiving the RSS feed will be able to use the data. At a minimum the URL name and description is needed. Any data that is published inside CMS can be turned into a feed.

First we should use PHP to connect to MySQL database, pull updated information out, and format it into an XML RSS document. We will create a connection as normal, and generate a page displaying the XML laid out in a user readable fashion.

Sending the page to the requestor

Now that we have the data all well formatted in the code, we need to make sure we hand the data out properly so when someone inputs the RSS feed URL into their reader, they will get the XML RSS feed they expect (see Listing 2).

Listing 5. The complete php code for generating feed

```
<?php

$database =    "nameofthedatabase";
$dbconnect = mysql_pconnect(localhost, dbuser, dbpassword);
mysql_select_db($database, $dbconnect);
$query = "select link, news, description from `news` limit 15";
$result = mysql_query($query, $dbconnect);
while ($line = mysql_fetch_assoc($result))
          {
               $return[] = $line;
          }
$now = date("D, d M Y H:i:s T");
$output = "<?xml version=\"1.0\"?>
               <rss version=\"2.0\">
                    <channel>
                         <title>HAW News</title>
<link>http://www.go2eu-club.com/haw.htm</link>
                         <description>HAW News</description>
                         <language>en</language>
                         <pubDate>$now</pubDate>
```

```
                        <lastBuildDate>$now</lastBuildDate>
                        <docs>docsURL</docs>
<managingEditor>someone@haw-hamburg.de</managingEditor>
<webMaster>webmaster@haw-hamburg.de</webMaster>
                    ";
foreach ($return as $line)
{
      $output .= "<item><title>".htmlentities($line['news'])."</title>
                        <link>".htmlentities($line['link'])."</link>
<description>".htmlentities(strip_tags($line['description']))."</description>
                    </item>";
}
$output .= "</channel></rss>";
header("Content-Type: application/rss+xml");
echo $output;
?>
```

So let's go through this step by step. First, we set up a database connection object to a local database. In that database, we have a table with records containing news, link and description fields, which we will request to put into our XML response. An SQL query statement is executed against table with MYSQL_QUERY() function and with the result, we reformat using WHILE to walk through the resulting object, and reformat the data into a new simple array.

When the new array is ready, we start to build the XML file in the $output variable, appending new elements by walking through the $line array once for each returned response.

When this script is executed, we get a nice clean RSS file output similar to that in Listing 3.

Listing 6. RSS.php output

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- generator="RSSxl/2.0" -->
<rss version="2.0">
   <channel>
      <title>HAW-Newsletter</title>
      <link>http://newsletter.haw-hamburg.de/</link>
      <description>Weekly News about HAW Hamburg</description>
      <language>en</language>
      <docs>http://www.wotzwot.com/rssxl.php</docs>
           <image>
```

```
      <title>HAW-Newsletter</title>
      <url>http://newsletter.haw-hamburg.de/images/HAWLogo1.gif</url>
      <link>http://www.haw-hamburg.de</link>

   </image>
         <image>http://newsletter.haw-hamburg.de/images/HAWLogo1.gif</i
mage>
      <item>
        <title>Studierende von heute - Kollegen von morgen! </title>

<link>http://newsletter.haw-hamburg.de/modules.php?op=modload&name=Ne
ws&file=article&sid=823&mode=thread&order=0&thold=0</link>
        <description>Am Dienstag, den 19. Juni, findet die Ausbildertagung des
Departments Information statt. Die Tagung steht dieses Jahr unter dem Motto
"Studierende von heute - Kollegen von morgen".</description>
      </item>
              <item>
        <title> Recruiting-Workshop der Lufthansa Technik AG
 </title>
        <link>http://wwww.hanseaticconsulting.de/lufthansa-technik.html</link>
        <description>Die Lufthansa Technik AG veranstaltet am 13. Juli 2007
einen Recruiting-Workshop in Hamburg.
</description>
      </item>
   </channel>
</rss>
```

Once this php program is executed , we will get a new fresh RSS feed .We now put this RSS feed to some of the readers we mentioned in previous chapter.

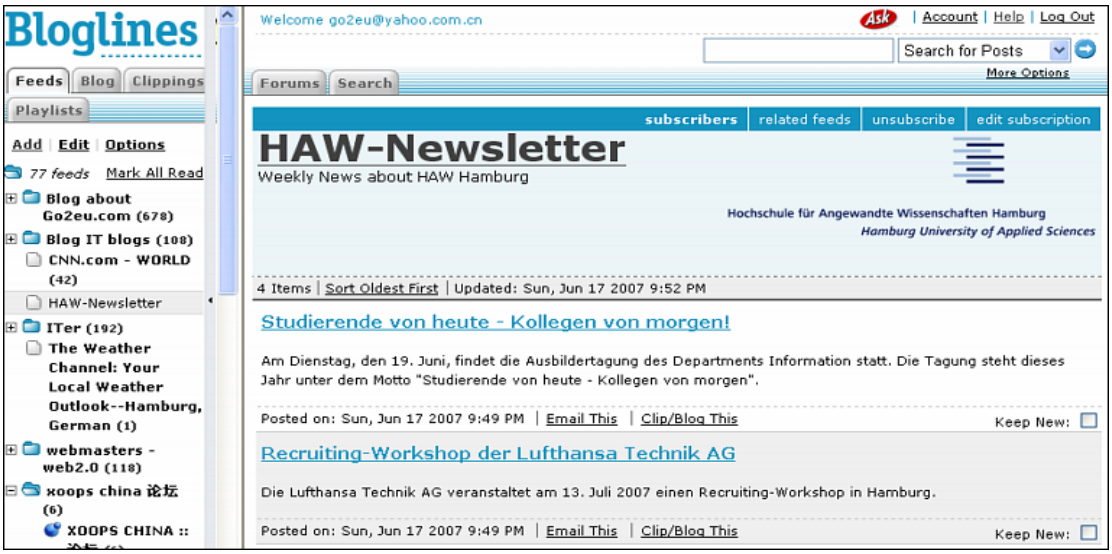


Figure 13. bloglines view of showing feed data

Figure 14: Spb Insight views of showing feed data

Chapter 8

Conclusion

There are many resources on the Web explaining how to quickly build a L.A.M.P. based application or website in a very efficient way. Inside this thesis, we discussed a maybe not the best, but probably the easiest method to achieve that goal: Using an open-source content management system.

XOOPS dynamic web CMS is very powerful, but it will become even better if we apply some new attractive technologies to it, like RSS syndication feature discussed in previous chapter. RSS syndication is just an example, by the means of XOOPS module design methods described in this thesis, we could extend XOOPS dynamic web CMS's function as good as we want it to be with very limited PHP programming knowledge.

New technologies will keep affecting internet itself, affecting the way people are using internet .

So this thesis is just a tip of the iceberg, a worm up ……

Chapter 9

References

[1]http://radar.oreilly.com/archives/2006/05/controversy_about_our_web_20_s.html Retrieved on June 16, 2007.

[2] Tim O'Reilly (2006-12-10). Web 2.0 Compact Definition: Trying Again. Retrieved on June 16, 2007.

[3] Web 2.0 http://en.wikipedia.org/wiki/Web_2.0 Retrieved on June 16, 2007.

[4] Dynamic Web Sites
http://www.woodst.com/downloads/newsletters/word/vol1/vol1_5.doc Retrieved on June 16, 2007.

[5] Jono Bacon Introduction to L.A.M.P. technology
https://www6.software.ibm.com/developerworks/education/wa-L.A.M.P./section2.html Retrieved on June 16, 2007.

[6] Peter Yared Shining a Light on L.A.M.P.
http://www.developer.com/db/print.php/3560296 Retrieved on June 16, 2007.

[7] L.A.M.P. (software bundle)
http://en.wikipedia.org/wiki/L.A.M.P._(software_bundle) Retrieved on June 16, 2007.

[8] Peter Yared Shining a Light on L.A.M.P.
http://www.developer.com/db/print.php/3560296 Retrieved on June 16, 2007.

[9] Peter Yared Shining a Light on L.A.M.P.
http://www.developer.com/db/print.php/3560296 Retrieved on June 16, 2007.

[10] L.A.M.P. (software bundle)
http://en.wikipedia.org/wiki/L.A.M.P._(software_bundle) Retrieved on June 16, 2007.

[11] GNU GENERAL PUBLIC LICENSE http://www.gnu.org/copyleft/gpl.html

[12] Peter Yared Shining a Light on L.A.M.P. www.developer.com/open/article.php/3560296 Retrieved on June 16, 2007.

[13] Usage Stats for April 2007 http://www.php.net/usage.php Retrieved on June 16, 2007.

[14] Netcraft June 2007 web servey http://news.netcraft.com/archives/web_server_survey.html Retrieved on June 16, 2007.

[15] Source: JoinVision E-Services GmbH, July 2006

[16] RSS 2.0 Specification http://cyber.law.harvard.edu/rss/rss.html Retrieved on June 16, 2007.

[17] Vincent Lauria Introduction to Syndication, (RSS) Really Simple Syndication http://www.ibm.com/developerworks/xml/library/x-rssintro/index.html Retrieved on June 16, 2007.

[18]RSS History http://cyber.law.harvard.edu/rss/rssVersionHistory.html Retrieved on June 16, 2007.

[19] Vincent Lauria Introduction to Syndication, (RSS) Really Simple Syndication http://www.ibm.com/developerworks/xml/library/x-rssintro/ Retrieved on June 16, 2007.

[20] SPB insight http://www.spbsoftwarehouse.com/products/insight/?en Retrieved on June 16, 2007.

[21] SPB insight Screenshots http://www.spbsoftwarehouse.com/products/insight/screenshots.html?en Retrieved on June 16, 2007.

[22] Introduction to Content Management Systems http://docs.phpcms.de/index.php/CMS_Introduction Retrieved on June 16, 2007.

[23] XOOPS dynamic web CMS http://www.XOOPS.org Retrieved on June 16, 2007.

[24] All About XOOPS http://www.XOOPS.org/modules/wfchannel/ Retrieved on June 16, 2007.

[25] Tracy Peterson http://www-128.ibm.com/developerworks/xml/library/x-phprss/?ca=dgr-lnxw01PHP-RSS Retrieved on June 16, 2007.

[26] XML_RSS http://pear.php.net/manual/en/package.xml.xml-rss.php Retrieved on June 16, 2007.
[27] dev XOOPS WIKI http://dev.xoops.org/modules/phpwiki/ Retrieved on June 16, 2007.

Appendix

A1 Get Dynamic Website up and running using XOOPS CMS

Following steps describes the installation process of XOOPS CMS, included in the downloaded package.

Predations and System requirements

To install XOOPS for the first time, I need to have the minimum following server software pre-installed L.A.M.P. or WAMP:
* HTTP Server (Apache) "Note, XOOPS only officially supports Apache"
* PHP 4.1.0 and higher (4.1.1 or higher recommended)
* MySQL Database 3.23.XX

Before starting the install, I should have:
* Setup the HTTP, PHP and database server properly.
* Create a database for your XOOPS 2 installation
* A user account with the proper database permissions.
* The ability to set the following directories and files world writeable: uploads/, cache/ and templates_c/ and the file mainfile.php
* Turn cookie and JavaScript support in your browser on.

Step by Step to get XOOPS CMS up and running

Start the installation by typing http://yoursite.com. This will start the install process.

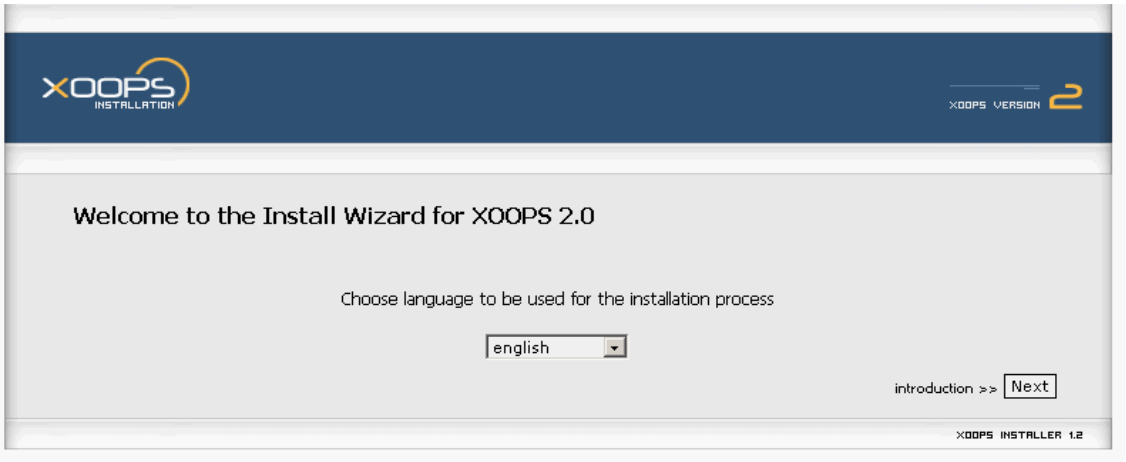


Figure 15: XOOPS Install Wizard

After performing the above procedures, your ready to continue installing XOOPS 2 with the Install Wizard.

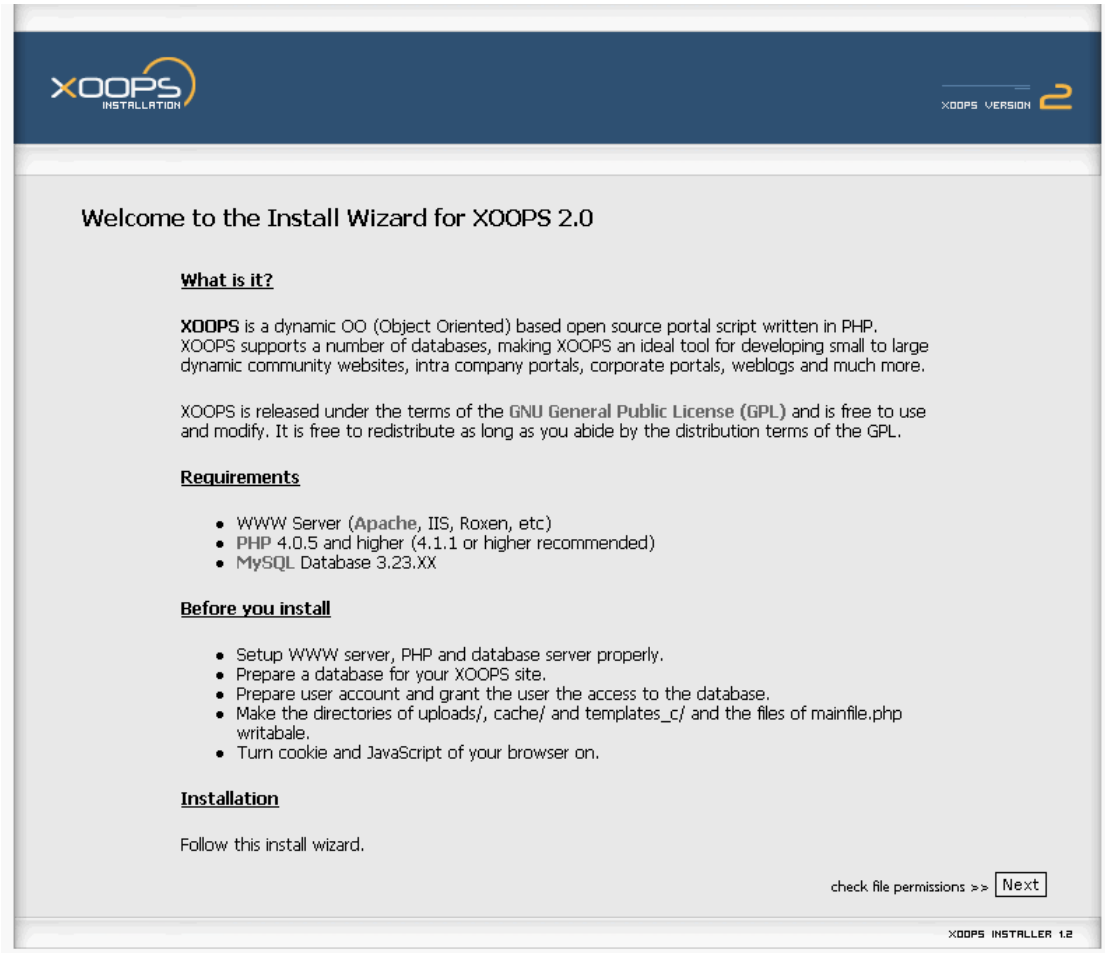


Figure 16: XOOPS Install Wizard Welcome Page

Click the Next button to continue on to the next screen.

The next part of the install Wizard is designed to check your file and directory permissions. If you're running in a Win32 environment, this should be a pretty painless install. If you running in a UNIX environment, the Wizard will display any problems and the corrective actions to take if there are problems.

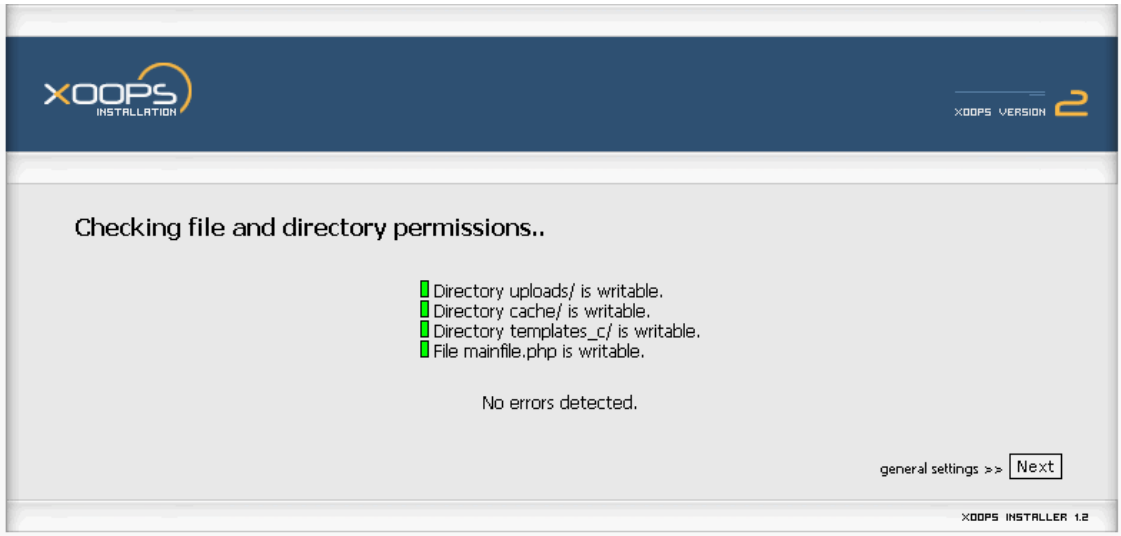


Figure 17: XOOPS Install Wizard: Checking permissions

If all lights are green on the Wizard, click Next to continue. If not, please read the screen and perform the necessary actions recommended by the Wizard.
The next part of the Wizard is for writing the settings to the mainfile.php file.

Figure 18. XOOPS Install Wizard: General Configuation

The General Settings screen is self explanatory, so input the required information into the files and click next.

The next four Wizard screens are informational displaying the settings from the General Settings screen for your confirmation and to show that the values were

written correctly.

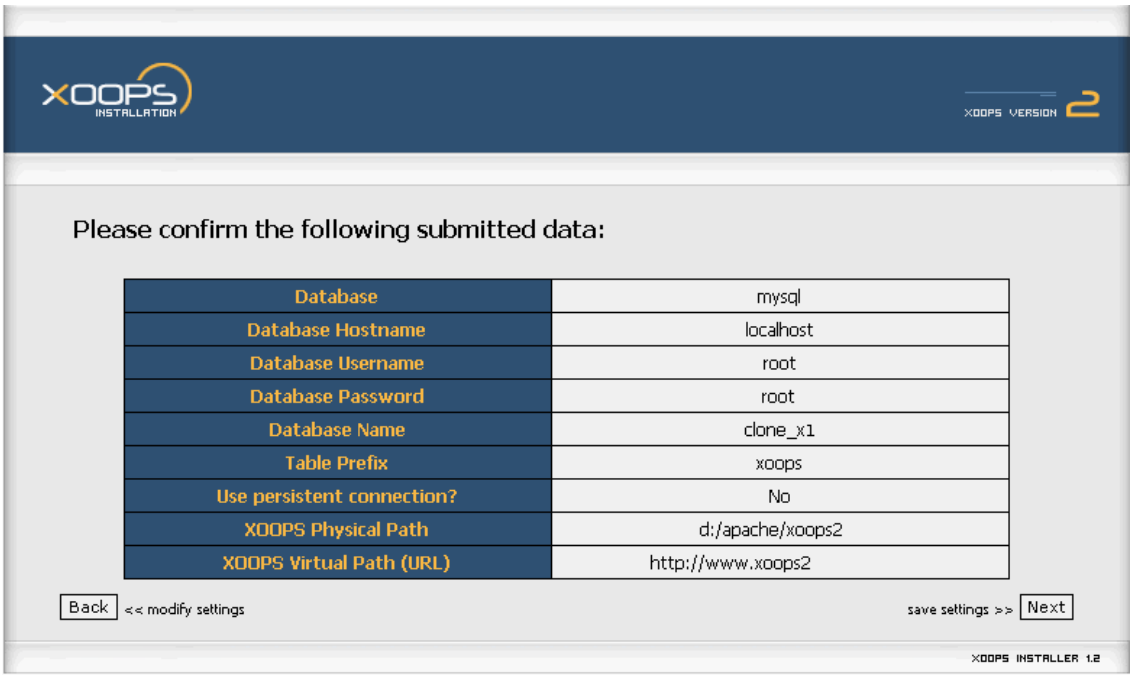


Figure 19: XOOPS Install Wizard: Confirm submitted date

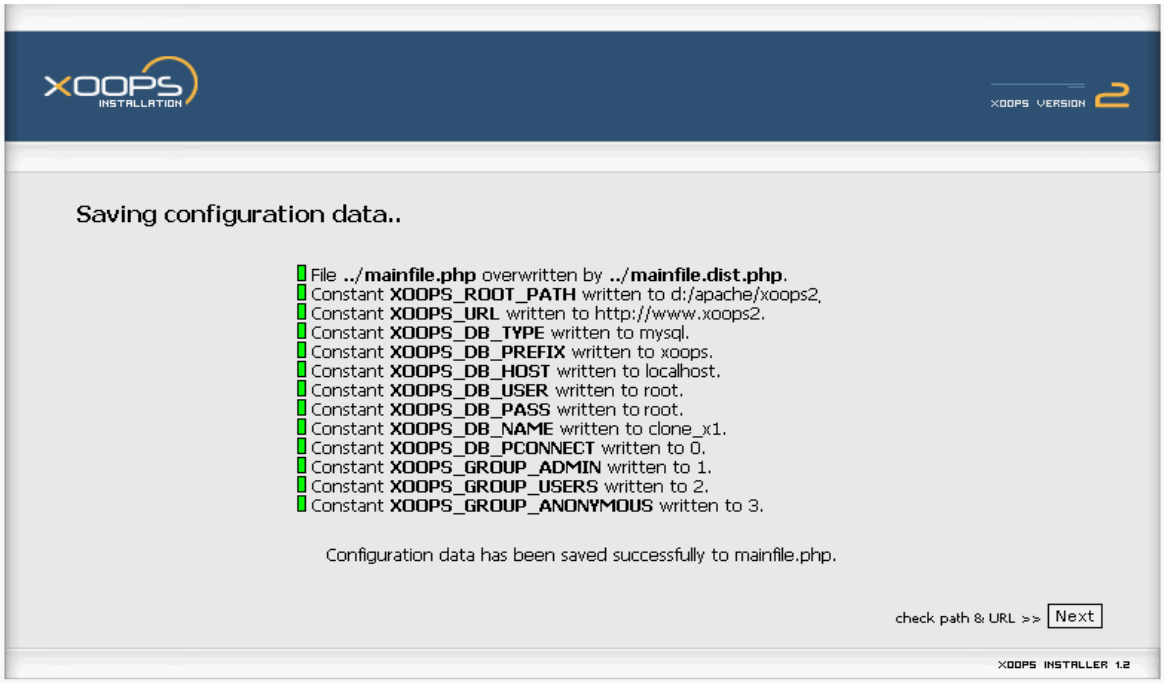


Figure 20: XOOPS Install Wizard: Saving configuration date

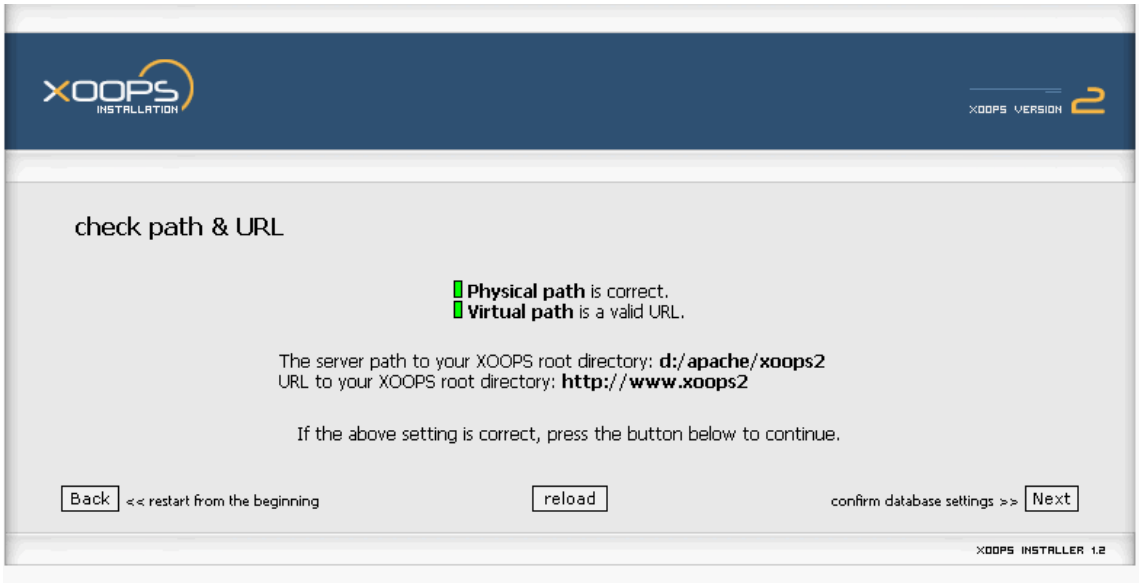


Figure 21: XOOPS Install Wizard: Check path & URL

Figure 22: XOOPS Install Wizard: Confirm database settings

If you seen any Red lights, please click the Back button to make the proper corrections.

The next Wizard screen is will be to show the progress for accessing the database.

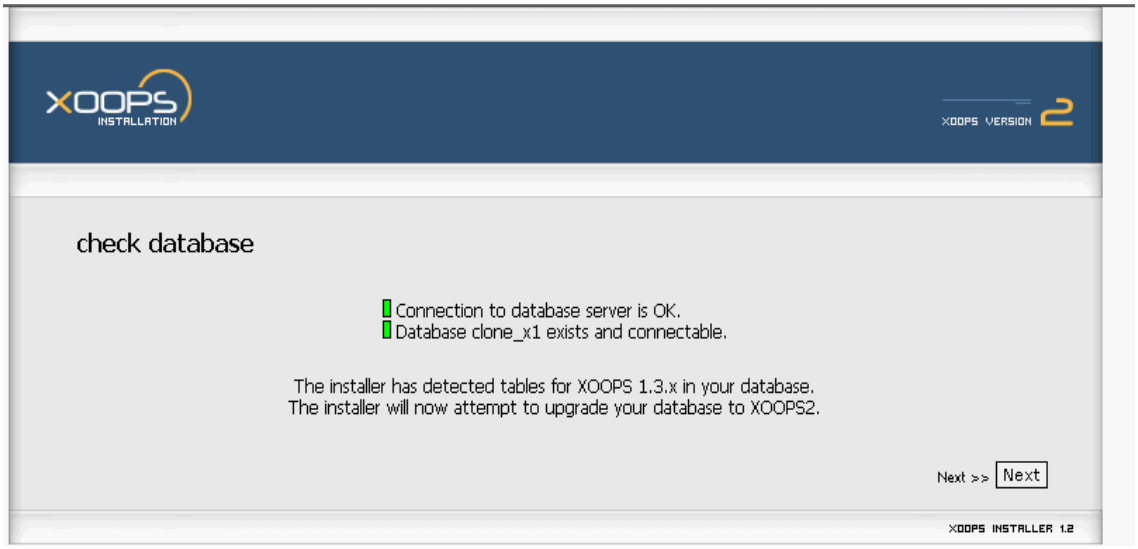


Figure 23: XOOPS Install Wizard: Check database

If you're in a hosted environment with out the proper access to create databases, please check with your provider for help in getting a database. If your provider (or you) created the database, all lights should be green. If you get a red light stating the DB does not exist and your user name has the ability to create databases, then click next and the install Wizard will attempt to create the database for you. If your user id does not have the rights to create a database, please correct this and continue the install. The next two screens are informational on trying to create and access the database. Click next or back depending on the Wizard screen.

After clicking next a couple of times, you will come to an informational screen showing the results of table creation.

Figure 24: XOOPS Install Wizard:Saving configuration data

If all lights are green, you're ready to proceed by clicking Next.

The next Wizard screen is for inputting site administrative information.

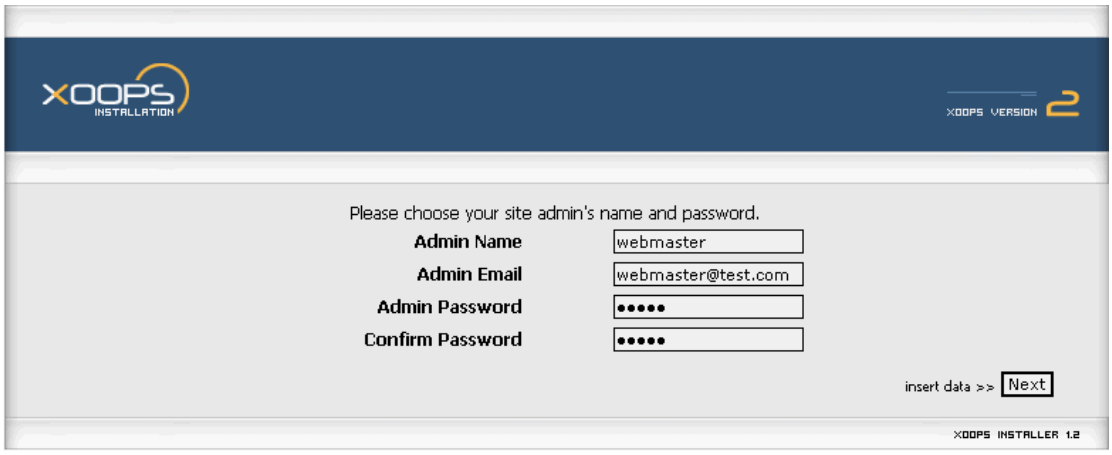


Figure 25: XOOPS Install Wizard: Administrator information

Please be careful here and write down or remember your administrative password. You'll need this after the install to continue setting up your XOOPS 2 site. Once you have completed inputting the correct information, click Next to continue. * Note, try

to refrain from using names with spaces for the Admin name. *

The next screen is informational. If all the lights are green, click Next to continue.

Congratulations!
Your installation should now be complete. You can check the site out by clicking the "HERE" text on the last screen. If all went well, then your new site should be up and running.

A2 Adding XOOPS global comments feature to module

Step 1

First of all, there are 2 variables that you must prepare before proceeding to subsequent steps. Those are:

- A.The name of unique ID for an item to which comments will be added. For example, this will be 'storyid' in News module, and 'poll_id' in XOOPSPoll module.
- **B**. The name of file which displays each item when the above unique item ID is passed as HTTP GET request. For example, the file name will be 'article.php' for News module, where an each article will be displayed by accessing this file as article.php?storyid=(unique id here). Similarly, it will be 'pollresults.php' for XOOPSPoll module.

Now open XOOPS_version.php and add the following lines:

```
$modversion['hasComments'] = 1;
$modversion['comments']['itemName'] = 'value obtained in A';
$modversion['comments']['pageName'] = 'value obtained in B';
```

For example, in News module:

```
$modversion['hasComments'] = 1;
$modversion['comments']['itemName'] = 'storyid';
$modversion['comments']['pageName'] = 'article.php';
```

Step 2

Copy the following files from the web links module and save them to your module directory

- comment_new.php
- comment_edit.php
- comment_delete.php
- comment_post.php
- comment_reply.php

Step 3

Open the file specified in **Step 1B** (i.e. article.php in News), and add the following line just before including footer.php

```
include XOOPS_ROOT_PATH.'/include/comment_view.php';
```

Step 4

Open the appropriate template file for your module (news_article.html for News module), and copy paste the following lines where comments should be displayed (You can of course customize the HTML tags as you prefer).

Listing 7. template file :HTML

```
<div style="text-align: center; padding: 3px; margin: 3px;">
   <{$commentsnav}>
   <{$lang_notice}>
</div>

<div style="margin: 3px; padding: 3px;">
<!-- start comments loop -->
<{if $comment_mode == "flat"}>
   <{include file="db:system_comments_flat.html"}>
<{elseif $comment_mode == "thread"}>
   <{include file="db:system_comments_thread.html"}>
<{elseif $comment_mode == "nest"}>
   <{include file="db:system_comments_nest.html"}>
<{/if}>
<!-- end comments loop -->
</div>
```

That's about all you would need to add on the user side. As for the admin side, **ALWAYS** call the following function whenever an item is deleted so that the comments attached to the deleted item will also be removed and number of user posts be updated accordingly.

function XOOPS_comment_delete(integer **module_id** , integer **item_id**)

For example in News module, the function is called as below whenever a news article is deleted:

XOOPS_comment_delete($XOOPSModule->getVar('mid'), $storyid)

Another useful function is XOOPS_comment_count(), which takes module ID and item ID as parameters and will return the total number of comments for the specified item.

function XOOPS_comment_count(integer **module_id** [, integer item_id*])

If **item_id** is not specified, then the total number of comments for the module specified by **module_id** will be returned.

Step 5 (Optional)

Setting up callback functions

You can specify callback functions by adding the following lines to XOOPS_version.php.

$modversion['comments']['callback']['approve'] = '**function**';

function will be executed upon successful post of an approved comment. This includes comment posts by administrators, and change of comment status from 'pending' to 'active' state. An XOOPSComment object that has been approved will be passed as the first and only parameter. This should be useful for example notifying the item submitter of a comment post.

$modversion['comments']['callback']['update'] = '**function**';

function will be executed whenever the total number of 'active' comments for an item is changed. Two parameters will be passed as parameters, the unique ID of an item as the first parameter and the total number of active comments for that item as the second.

$modversion['comments']['callbackFile'] = '**file name**';

The name of file in which callback functions are defined.

Example

modules/mylinks/XOOPS_version.php

```
$modversion['comments']['callbackFile'] = 'include/comment_functions.php';
$modversion['comments']['callback']['approve'] = 'mylinks_com_approve';
$modversion['comments']['callback']['update'] = 'mylinks_com_update';
```

modules/mylinks/include/comment_functions.php

```
function mylinks_com_update($link_id, $total_num){
        $db =& Database::getInstance();
        $sql = 'UPDATE '.$db->prefix('mylinks_links').' SET comments =
'.$total_num.' WHERE lid = '.$link_id;
        $db->query($sql);
}
```

```
function mylinks_com_approve(&$comment){
        / / send notification mail
}
```

A3. File : XOOPS_version.php

This file is used during module installation and removal, but otherwise not utilized during module usage. The <path2module>/language/<lang>/modinfo.php file is automatically included and expects the definitions for module installation included here. However, define statements and other defined variables in these files will NOT be available to your module except the Module Configuration data entered during installation.

Main Info

- $modversion['name'] = _MI_NEWBB_NAME;
  - Name of the module
- $modversion['version'] = 1.00;
  - Module version
- $modversion['description'] = _MI_NEWBB_DESC;
  - Module description
- $modversion['credits'] = "Kazumi Ono";
  - Module credits
- $modversion['author'] = "Me";
  - Module Author
- $modversion['help'] = "newbb.html";
  - Module help template
- $modversion['license'] = "GPL see LICENSE";
  - Module License
- $modversion['official'] = 1;
  - Is official XOOPS module (usually 0 for us)
- $modversion['image'] = "images/XOOPSbb_slogo.png";
  - Image in the modules panel and left nav menu (if any)
- $modversion['dirname'] = "newbb";
  - Directory of the module

Install and Uninstall

- $modversion['onInstall'] = "path/to/file/with/install/function";
  - Path to the file (from the module folder) containing a function called XOOPS_module_install_{dirname} that takes a XOOPSModule object as parameter. This function will be run AFTER module is installed
- $modversion['onUninstall'] = "path/to/file/with/uninstall/function";
  - Path to the file (from the module folder) containing a function called XOOPS_module_uninstall_{dirname} that takes a XOOPSModule object as parameter. This function will be run AFTER module is uninstalled

SQL

- $modversion['sqlfile']['mysql'] = "sql/mysql.sql";
    - SQL statements to create the database tables needed by this module
- $modversion['tables'][0] = "bb_categories";
    - List of tables created by the the sql dump; Used for uninstallation
    - Increase [0] by one for the next table (ex: ['tables'][1],['tables'][2],etc)

Admin

- $modversion['hasAdmin'] = 1;
    - Display in the admin nav bar
- $modversion['adminindex'] = "admin/index.php";
    - Admin index
- $modversion['adminmenu'] = "admin/menu.php";
    - Popup menu for admin menu icon

Menu

- $modversion['hasMain'] = 1;

Put in the Main Menu (0 for no). Please note that setting this to 0 also removes this module from the list of modules you can give a group access to from the groups admin screen. It's last permission value will remain in effect.

- $modversion['sub'][1]['name'] = _MI_NEWS_SMNAME1;
    - Name of the submenu in the Main Menu (when you click on the module)
- $modversion['sub'][1]['url'] = "submit.php";
    - File to run in the module directory

Templates

- $modversion['templates'][1]['file'] = 'newbb_index.html';
    - All the template files (for subdirs do 'subdir/newbb_sub.html')
    - Increase [1] by one for the next file (ex: ['templates'][2],['templates'][3],etc)
    - It will be saved in the database
    - Note: This item starts at [1], not [0].
- $modversion['templates'][1]['description'] = '';
    - A short description of the template file
    - Increase [1] by one for the next description (ex: ['description'][2],['description'][3],etc)
    - It will be saved in the database
    - Note: This item starts at [1], not [0].

Blocks

- a 'file' can have many names, descriptions, functions, options, edit_func, and templates --- (just create another set, ie: [2])
- $modversion['blocks'][1]['file'] = "newbb_new.php";
- $modversion['blocks'][1]['name'] = _MI_NEWBB_BNAME1;
- $modversion['blocks'][1]['description'] = "Shows recent topics in the forums";
- $modversion['blocks'][1]['show_func'] = "b_newbb_new_show";
- $modversion['blocks'][1]['options'] = "10|1|time";
- $modversion['blocks'][1]['edit_func'] = "b_newbb_new_edit";
- $modversion['blocks'][1]['template'] = 'newbb_block_new.html';

Search

- $modversion['hasSearch'] = 1;
- $modversion['search']['file'] = "include/search.inc.php";
- $modversion['search']['func'] = "newbb_search";

Notification

- $modversion['hasNotification'] = 1;
- $modversion['notification']['lookup_file'] = 'include/notification.inc.php';
- $modversion['notification']['lookup_func'] = 'newbb_notify_iteminfo';
- $modversion['notification']['category'][1]['name'] = 'thread';
- $modversion['notification']['category'][1]['title'] = _MI_NEWBB_THREAD_NOTIFY;
- $modversion['notification']['category'][1]['description'] = _MI_NEWBB_THREAD_NOTIFYDSC;
- $modversion['notification']['category'][1]['subscribe_from'] = 'viewtopic.php';
- $modversion['notification']['category'][1]['item_name'] = 'topic_id';
- $modversion['notification']['category'][1]['allow_bookmark'] = 1;
- $modversion['notification']['event'][1]['name'] = 'new_post';
- $modversion['notification']['event'][1]['category'] = 'thread';
- $modversion['notification']['event'][1]['title'] = _MI_NEWBB_THREAD_NEWPOST_NOTIFY;
- $modversion['notification']['event'][1]['caption'] = _MI_NEWBB_THREAD_NEWPOST_NOTIFYCAP;
- $modversion['notification']['event'][1]['description'] = _MI_NEWBB_THREAD_NEWPOST_NOTIFYDSC;
- $modversion['notification']['event'][1]['mail_template'] = 'thread_newpost_notify';
- $modversion['notification']['event'][1]['mail_subject'] = _MI_NEWBB_THREAD_NEWPOST_NOTIFYSBJ;

Configuration items

- $modversion['config'][1]['name'] = 'profile_search';
- $modversion['config'][1]['title'] = '_PROFILE_MI_PROFILE_SEARCH';
- $modversion['config'][1]['description'] = '_PROFILE_MI_PROFILE_SEARCH_DSC';
- $modversion['config'][1]['formtype'] = 'yesno';

can be 'yesno', 'select', 'select_multi', 'group', 'group_multi', 'textbox', 'textarea', 'user', 'user_multi', 'timezone' or 'language'

- $modversion['config'][1]['valuetype'] = 'int';

can be 'int', 'float', 'textarea' or 'array'. All items with formtype 'multi_xxx' must have the valuetype 'array'

- $modversion['config'][1]['default'] = 1;
- $modversion['config'][1]['category'] = 'settings'; //Implemented in XOOPS 2.2

Configuration Categories

- $modversion['configcat'][1]['nameid'] = 'settings';

the referenced name used in the configuration item's 'category'

- $modversion['configcat'][1]['name'] = '_PROFILE_MI_CAT_SETTINGS';

the displayed name/title

- $modversion['configcat'][1]['description'] = '_PROFILE_MI_CAT_SETTINGS_DSC';

Page Awareness

- $modversion['pages'][1]['name'] = _PROFILE_MI_PAGE_INFO;
- $modversion['pages'][1]['url'] = "filename.php";

Dynamic User Profile

- $modversion['hasProfile'] = 1;
- $modversion['profile']['field'][1]['name'] = 'profile_aim';

field name - can be referenced with $XOOPSUser->getVar('profile_aim') in module code

- $modversion['profile']['field'][1]['type'] = 'textbox';

type of form element for editing

- $modversion['profile']['field'][1]['valuetype'] = XOBJ_DTYPE_TXTBOX;

type of field - use XOOPSObject value types found in kernel/object.php

- $modversion['profile']['field'][1]['maxlength'] = 255;

maxlength of the field - Note: Mandatory when dealing with XOBJ_DTYPE_TXTBOX fields

- $modversion['profile']['field'][1]['default'] = '';

Default value

- $modversion['profile']['field'][1]['show'] = 1;

can this field be shown in user profiles (still subject to group permissions)

- $modversion['profile']['field'][1]['title'] = _PROFILE_AIM_TITLE;

Name of field, when displayed - such as in user profile or editing

- $modversion['profile']['field'][1]['edit'] = 1;

can this field be edited in user profile editing (still subject to group permissions)

- $modversion['profile']['field'][1]['description'] = _PROFILE_AIM_DESCRIPTION;

description - such as when editing the profile, this will show up

- $modversion['profile']['field'][1]['required'] = 0;

is field required when editing?

- $modversion['profile']['field'][1]['config'] = 1;

can this field be configured? Set this to zero if you rely on this field's information in your module code as configuration can alter the field completely - or DELETE it - if this is enabled

- $modversion['profile']['field'][1]['options'] = array(); [27]

Declaration

I declare within the meaning of section 25(4) of the Examination and Study Regulations of the International Degree Course Information Engineering that: this Bachelor Thesis has been completed by myself independently without outside help and only the defined sources and study aids were used. Sections that reflect the thoughts or works of others are made known through the definition of sources.

Hamburg , 18 June 2007 ,_____