PHP−Nuke: Management and Programming

Claudio Erba

Webmaster
www.spaghettibrain.com

Chris Karakas – Conversion from LyX to DocBook SGML, Index generation

Andre Purfield – Translation from italian and translation project coordination

Fortunato Matarazzo – Translation from italian: chapters 7, 8 and 9

Chris Karakas – Translation from italian: chapters 1−6, 10−11

Revision History

Revision 1.2	2003−05−29	Revised by: CK
New logo, CSS stylesheet, HTML validation, footer icons. Incorporated LDP reviewer's comments. Created Aknowledgements and Availability of sources sections.		
Revision 1.1	2003−02−13	Revised by: AP
Cleaned up the wording and a few typos.		
Revision 1.0	2003−01−09	Revised by: CK
First complete version.		

There has always been the necessity to have a definitive guide on PHP−Nuke. This tutorial describes the installation and structure of PHP−Nuke and the details of customizing the front end to suit the users' needs. The architecture of PHP−Nuke, with its modules, blocks, topics and themes is presented in detail, as well as the interplay of PHP and MySQL for the creation of a mighty content management system (CMS).It also delves into more advanced issues, like the programming of PHP−Nuke blocks and modules.

Table of Contents

Table of Contents

Chapter 1. Terms of distribution

1.1. Disclaimer

No liability for the contents of this documents can be accepted. Use the concepts, examples and other content at your own risk. As this is a new edition of this document, there may be errors and inaccuracies, that may of course be damaging to your system. Proceed with caution, and although this is highly unlikely, the author does not take any responsibility for that.

All copyrights are held by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark.

Naming of particular products or brands should not be seen as endorsements.

1.2. Formats

This document is available in the following formats:

- HTML (HyperText Markup Language), many HTML files (one for every section)
- HTML (HyperText Markup Language), one big HTML file
- TXT (ASCII Text)
- RTF (Rich Text Format)
- PDF (Portable Document Format)
- PS.GZ (Compressed Postscript)
- SGML (Standard Generalized Markup Language)
- LYX (LaTeX frontend LyX)

IMPORTANT: Downloads for offline reading!

If you want to download the HTML or RTF formats for offline reading, you will need to download the images as well – PNG for HTML and BMP for RTF, including the callouts! To save you the hassle, I have compiled the following zipped tar archives for offline reading:

- TAR.GZ (Compressed TAR Archive), many HTML files with images
- TAR.GZ (Compressed TAR Archive), one big HTML file with images
- TAR.GZ (Compressed TAR Archive), SGML file with images
- TAR.GZ (Compressed TAR Archive), RTF file with images

A tarball containing all the above formats, including images, is also available:

- TAR.GZ (Compressed TAR Archive), All files

1.3. Licence

1.4. Availability of sources

The modifiable sources of the original book (in italian), the images and example files are available in sxi format (OpenOffice Impress) on spaghettibrain. Openoffice is an office suite completely free, downloadable from www.openoffice.org.

See Section 1.2 for the modifiable sources of this document. These are the official versions. We (the translators and current maintainers) plan to continue work on this document and add new chapters and enhancements. If you want to see the version we are currently working on (the "bleeding edge" version), check http://www.karakas−online.de/EN−Book/ from time to time.

1.5. Aknowledgements

Since I am still at the very start of this work here, I cannot claim to have carried out this operation to its end, but, as the saying says, those who have started, are already halfways

The following people contribute, directly or indirectly to this project and their help is hereby kindly aknowledged:

* Francisco Burzi with all the introductory scripts, found in the files of installation of PHP−Nuke
* Vasco Clergy and his daughter Valentina of Lug Rieti for the translations of the modules of the handbook
* Micaela bechini, for the daily translations of phpnuke.org from English.
* The communities of Splatt.it, Nukeitalia.com, PHPnuke.it (and its Mailing List), Postnuke.it, envolution.it, xoops.it and obviously the 1500 registered users of spaghettibrain.
* Aemmenet, in the person of Mark Atzori that has granted us free use of the server on which spaghettibrain is accommodated.
* Roberto Scano of IWA Italy and Patrizia Bertini of Webaccessibile.org for the contributions on usability, accessibility and W3C validation.
* Marcello Tansini of webmasterpoint.org for the support given to the project in terms of visibility.
* Andrea Birgahi, the best PHP−Nuke Theme Maker of the world for the diagram of spaghettibrain, for the logo of the book and a lot more.

* My girl Sara for all...
* My mother Lella, my sister Cora, my dog Grey and the newly arrived baby dog Maya.

This book I dedicate it to my Papà Antonio. Hello Pà.

Some quotes have been taken from the installation files of phpnuke.org.

The translators would like to thank the Linux Documentation Project (TLDP) and its reviewers, especially Tabatha Marshall and Greg Ferguson for their comments and efforts to make this document available from TLDP's plattform to a wider audience.

We are in search of available volunteers to translate this script in as many languages as possible. In the case you are interested in translating this book, write to webmaster@spaghettibrain.com, chris@karakas−online.de or visit the site that will have a classified section dedicated to this argument.

Chapter 2. Introduction to PHP−Nuke

2.1. Purpose

This book is born as a "thank you" to all the users who visit spaghettibrain. There has always been the necessity to have a definitive guide on PHP−Nuke, possibly in Italian language. Due to time constraints, nobody has ever had the will to carry out this operation.

2.2. What Is PHP−Nuke

PHP−Nuke is free software, released under the GNU License.

It is a CMS (Content Managment System) that integrates in its inside all the instruments that are used to create a site/portal of information (meant in broad sense). Given the immense number of present functions in the installation and in an even greater quantity of modules developed from third parties, the system is also adept to the management of

- Intranet business,
- e−commerce systems,
- corporate portals ,
- public agencies,
- news agencies,
- online companies,
- information sites,
- e−learning systems
- and so on...

PHP−Nuke utilizes as hinge of its own structure the duo PHP+ MySQL, very often being accompanied by the Apache web server. Many modules have integrated many other languages, such as Javascript, Java, Flash and also even systems that serve, through the portal, sounds and films in streaming mode(Online Radio, TV Online, Images, Files...)

PHP−Nuke is developed with a particular eye to the suggestions of the W3C, in its origin, the code is in fact W3C compliant and one has validated both the code and the style sheets. It is then up to the user who intends to create a portal to adhere to these standards during the modification of the graphics or the intrinsic characteristics of the system.

The personalisation either of the graphical, or of the programming part has only a single limit, the fantasy and capability of the programer and web designer.The presence of many PHP−Nuke sites similar to each other is due mainly to the lack of time of those who created them or the fear that the phase of personalisation is too difficult on a technical level. In fact, it suffices to let oneself be inspired by the available themes, in order to realize how easy it is to sew a new dress to onIt is s portal.

Francisco Burzi, father and mother of PHP−Nuke, describes his creation as follows:

> PHP−Nuke is a Web Portal System, storytelling software, News system, online community or whatever you want to call it. The goal of PHP−Nuke is to have an automated web site to distribute news and articles with users system. Each user can submit comments to discuss the

articles, just similar to Slashdot and many others.

Main features include: web based admin, surveys, top page, access stats page with counter, user customizable box, themes manager for registered users, friendly administration GUI with graphic topic manager, option to edit or delete stories, option to delete comments, moderation system, Referers page to know who link us, sections manager, customizable HTML blocks, user and authors edit, an integrated Banners Ads system, search engine, backend/headlines generation (RSS/RDF format), and many, many more friendly functions.

PHP−Nuke is written 100% in PHP and requires Apache Web server, PHP and a SQL (MySQL, mSQL, PostgreSQL, ODBC, ODBC_Adabas, Sybase or Interbase). Support for 25 languages, Yahoo like search engine, Comments option in Polls, lot of themes, Ephemerids manager, File Manager, Headlines, download manager, faq manager, advanced blocks systems, reviews system, newsletter, categorized articles, multilanguage content management and a lot more.

2.3. Short history of PHP−Nuke

Francisco Burzi, describes the history of PHP−Nuke as follows:

> PHP−Nuke is a free software, released under the GNU GPL License, version 2.0. PHP−Nuke is the result of many years administrating a news site called Linux Preview. First, around August 1998, I wrote my own code in Perl called NUKE and used it for about 1 year, then my site grew big, so I needed a more powerfull system and decided to use Slash, the same used in the Slashdot site. It's good, but you realy need to know Perl to modify it, need too many modules, need to load a damn daemon that sucks all your CPU power. My Pentium III just appears to be a 386 each minute the daemon make its work.
>
> Well, then I discovered Thatware, a good project to have a news site under PHP. I learned PHP in less than a week and began modifying it. There are too many mods to mention, it was practicaly a rewrite. I added some cool stuff, deleted some others and after more than 380 hours of hard work in 3 weeks! PHP−Nuke was born.
>
> On August 17, 2000 I sold LinuxPreview.org to LinuxAlianza.com and now I have all the time to dedicate to the development of PHP−Nuke.
>
> From January 2001 to January 2002, PHP−Nuke has been financially supported by MandrakeSoft, the folks that made Mandrake Linux. This gave me and PHP−Nuke a lot of oxygen and made possible a lot of stuff.
>
> Now, I'm alone with this killer project. There is a lot of help from the people that use and develop modules and themes. Now, phpnuke.org is a big site with a lot of users and helpful information for any user around the world. There are also strong users community sites in almost any language you can imagine. Just go to phpnuke.org and enjoy this great community!

2.4. The Nuke Communities

A careful look is due to the true value of PHP−Nuke, that is the communities that you will find all around. Thanks to the voluntary job of these persons, of these sites, PHP−Nuke has become a well−known system and it is always thanks to them that PHP−Nuke is a multilanguage system that supports more than 25 languages.

Even the modules have been created mostly from developers in external communities and have, in second round, been included in new distributions of PHP−Nuke.

There are communities out there who are solely devoted to the creation of new graphical themes of PHP−Nuke, to technical support, file mirroring as well as a real lot of multilingual communities that take care of their members informing them in their local language, thus creating new personal ties and more focused projects.

- Nukeforums.com: Technical support to PHP−Nuke
- Nukedownloads.com: File mirror for downloads
- Somara.com: Themes and graphics
- Nukethemes.com: Themes and graphics
- Ecomjunk.com: Addons and modules
- Nukeaddn.com: Addons and modules

Communities in Italian language:

- Spaghettibrain.com
- PHPnuke.it
- Splatt.it
- Nukeitalia.com

Thanks to the work of these portals and single persons we have more than 500 different modules that may be used to personalize our portal, from the weather one to the e−commerce, the gallery of images to the chat realized in flash to the videogames in Java included in the layout of PHP−Nuke. Projects of particular interest are: Splatt.it (Forum for PHP−Nuke), PHP ProximaPHP Proxima (visual management of the layout of PHP−Nuke).

2.5. Why use PHP−Nuke and not static HTML pages

- Because managing large sites with only static HTML pages is dangerous for your health.
- Because through the dynamic pages, users can interact (Forum, chat)
- Because through the dynamic pages we can offer value added services (restricted areas, various services based on user classification...)
- Because the information is more easily catalogued.
- Because with a few PHP pages we recall a lot of information.
- Because keeping the contents up−to−date does not demand particular technical expertise and can be managed by anyone (by Davis Batistes).
- It is the simplest way to to pull over a complete portal, thanks to its open source engine, it allows anyone to implement new modules or to modify and to personalize existing modules. (by Micione, www.vizzani.net)
- It is very intuitive and easy to learn (by Anonymous)
- It is easy to modify by those who intend to personalize the program (By Arus)

- It is easy to use by the lesser experts among us.

Chapter 3. Front end structure: user view

In this chapter we will occupy ourselves, in detail, with all the functionalities implemented in PHP−Nuke, that is what our portal system can do and how. We will do this from the part of the visitor, imagining that we are the one who visits our site and uses its functionality.

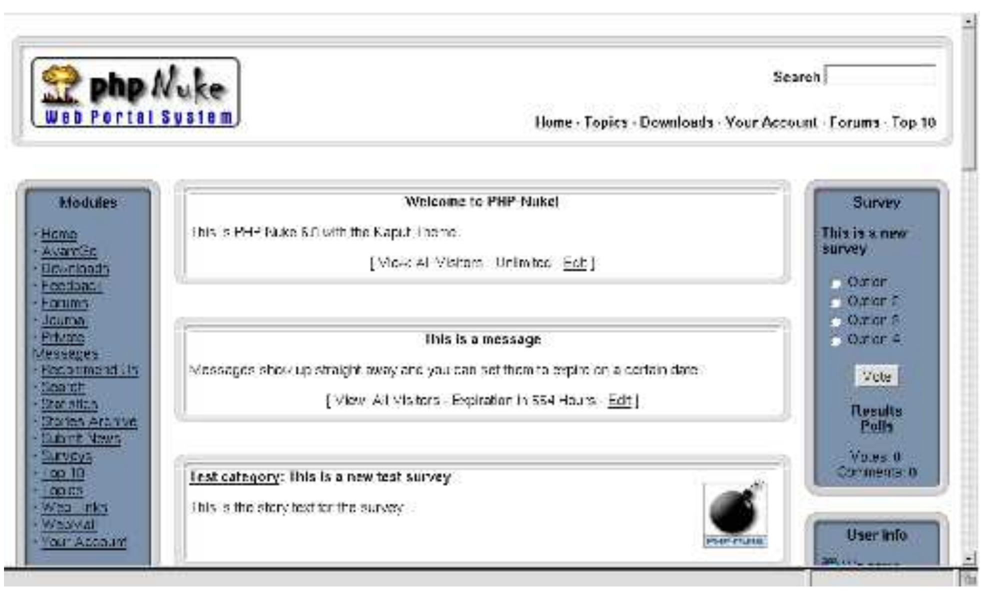
We will analyze all the preinstalled modules in the PHP−Nuke distribution and will give a look also at some very interesting modules that still have not been included in the official distribution.

Before starting, we should spend two words on how PHP−Nuke is structured; this system is structured as a 3 column portal, the two lateral ones including the blocks, the central one displaying the function modules. This does not mean to say that the structure of our site cannot be modified completely. The initial skeleton is, favorably, the one to start from in order to create a super personalized portal. Beyond the 3 columns mentioned we have also a header (top of page) and a footer (bottom of page).

Blocks:
: they are present in the left/right columns of our portal[1] and deliver functions that are repeated in all pages of the site (e.g. the menu, banner and login blocks).

Modules:
: They are the heart of the page, they appear in the center column and each one has its own function. For example the news module delivers the articles, the search module makes an internal search of our site − they should be imagined as independent pages. They are the "heart" of the page that we visit (see Figure 3−1).

Figure 3−1. PHP−Nuke Homepage



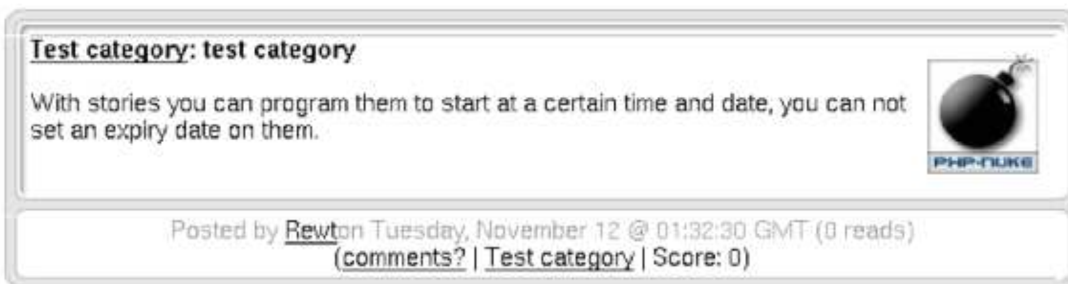
PHP−Nuke Homepage

3.1. The preinstalled modules

news:

Born as the heart of PHP−Nuke, it was the obligatory Home Page in previous versions. In the last versions it is instead possible to define which modules should appear as the default page. The News module expands its branches on more pages. The first one we see is a collection of the latest News published (it is possible , from the configuration panel [admin−>preferences] to choose the number of latest news to be displayed, say 5, 10, 15, 20, 25, 30). In the main page only a small initial text of the news article is published. If the text is too large, it will be possible to read it whole by pressing the link "Read more". The article module has many elements that distinguish it from other ones. In the first place, the title, the topic, that is the main category and is usually characterized by an image that, if clicked, brings up a selection of the articles that belong to this topic. We have a second way to classify the articles, which is to assign them a category they are supposed to belong (see Figure 3−2)

Figure 3−2. Classifying articles



Classifying articles

IMPORTANT

This category is NOT a subcategory of topics, but rather a cross−sectional category that is completely independent from topics. Probably its most important function is to distinguish between "articles" and other admin−defined classes of news for which it will be possible to NOT be automatically published on the start page. "Articles" will always appear on the start news page.

For example, imagine a portal that talks about soccer and it has 3 topics:

- League A
- League B
- League C

We could think of cross−sectional categories like:

- Championship
- Champions League
- Soccer player market

We can have an article that talks about League A/Championship, or of the Soccer player market League B. Clicking o topic, say League A, we will have a selection of all the articles that talk about League A, clicking on the category, Soccer player market, will have a selection of the articles that independently talk about soccer market that independently of the League being A, B or C.

At the bottom of the article we find more information about the article: who inserted it, when & how many times it has been read etc.

The counter is incremented only if the "Read more" link has been clicked. The counter does not increment unless the user clicks "Read more".

It displays how many more bytes are still to be read, if there have been any comments on the article and what score has been given to the article by the readers. It is also possible to print the article in a printer−friendly format or you can send the link via e−mail to a friend.

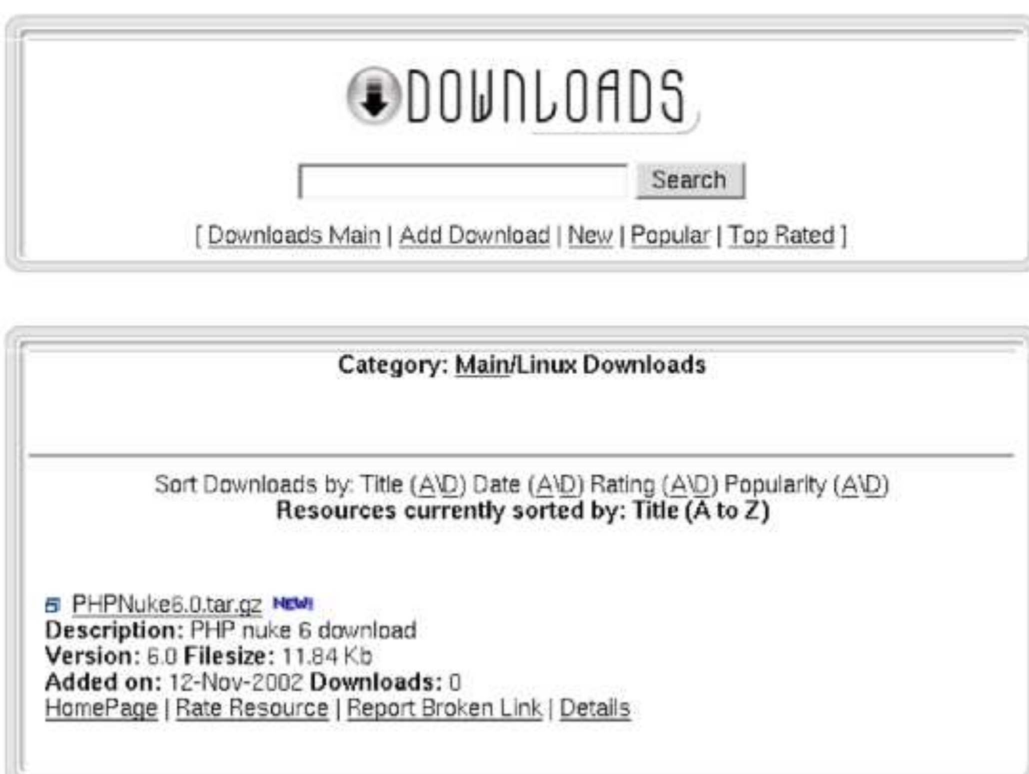
Clicking on " Read all " brings us to the page that contains the entire article and the comments pertaining to it. In this page the user can read the entire article and interact with it through a multitude of operations.

He/She can cast a vote for the article, thus expressing a judgement on its validity, can comment on the article or answer to comments inserted from other users, can follow the links associated with this article, display it in a printer−friendly format and send the link via e−mail to a friend. You can also attach a survey to the article.

AvantGO:
: It is a very simplified version of the news archive, cretated mainly from the need to visit the page via a Palm Pilot. AvantGO is a system for archiving and visualizing the pages on palmtop screens due to the fact the Palm Pilot has a very small screen and has a low resolution (and even a low bandwidth connection) so they require simplified pages.

Downloads Module:
: This module is deeply branched and manages a file archive (present on our own or a third party site) offering the user various modes of interaction (see Figure 3−3).

Figure 3−3. Downloads module



Downloads module

In the main page it is possible to use an internal search engine that searches for keywords among all the files cataloged. There is also the possibility to add external links to a file (these files will not be added immediately, but put in a waiting list until an administrator will approve them and they become visible). We can also base our selection on which files were downloaded most, or which ones obtained the highest score. On this page we can have a list of categories that accompany the files (there may exist subcategories but in Figure 3−3 there is only 1 category, "Linux Downloads"), the user is recognized when he/she views the downloads section after their first visit so if new downloads have been added since the last visit, the corresponding category will be have a "new" icon beside it.

Once we have entered the desired section, we can download the files that interest us, cast a judgment vote, report a nonexistent link to the administrator or see more information regarding the author of the file.

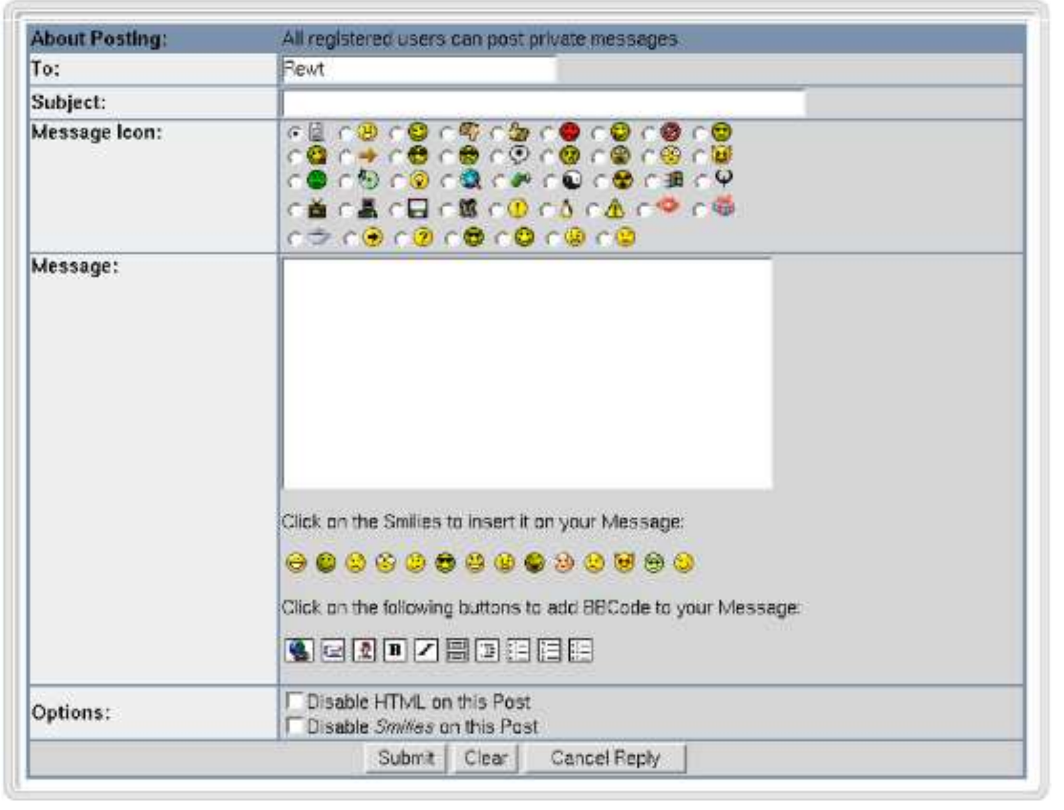
The file list may be ordered by insertion date, judgment or popularityy (files downloaded most).

Feedback:
: Allows the user to send feedback and contact the webmaster of the site. The user just fills in the appropriate fields which are "Name", "E−Mail" and then the "Message Text". The system will then format an e−mail message that will arrive to the webmaster of the site.

Member List:
: It displays all the users registered on the site. It is possible to select the users on the basis of their basic information fields (name, nickname, personal homepage and e−mail address). It is also possible to obtain a complete list of all the users and to order it by real name, e−mail address or homepage.

Private messages:
: All the registered users have access to an internal messaging system, thus being able to exchange messages with each other. In the login box of each user the number of messages that are archived for this user will be displayed, and there is a management functionality allowing for replies or deletions (see Figure 3−4).

Figure 3−4. Private messages



Private messages

The message that we compose has various parts:

- The Recipient
- The Subject
- The animated icon that will accompany the subject of the message
- The text that can be equipped with emoticons (emotional icons) and an aid for formatting the message in HTML adding Hyperlinks, emphasized words, bullet lists etc...

Recommend Us:
This module is so you can send an e−mail to a friend recommending visiting our PHP−Nuke site. The message that will be sent to the friend must be configured by the administrator.

Book Reviews:
This module serves as an archive of product/services/site reviews. The book review must be inserted by the administrator but also from a user (a book review will need , in this case, be accepted by the administrator) who, after inserting a short description of the product then may express his judgment assigning a score to it. It is also possible to insert a descriptive image. The book reviews are cataloged in alphabetical order and the selection can be made based on starting letter.

Search Module:
It is the main search engine for PHP−Nuke, it does full text searches on articles, comments, sections, users and book reviews (see Figure 3−5). It is possible to make multiple searches (e.g.. an article of a certain category written by a certain author).

Figure 3−5. Search module

Search module

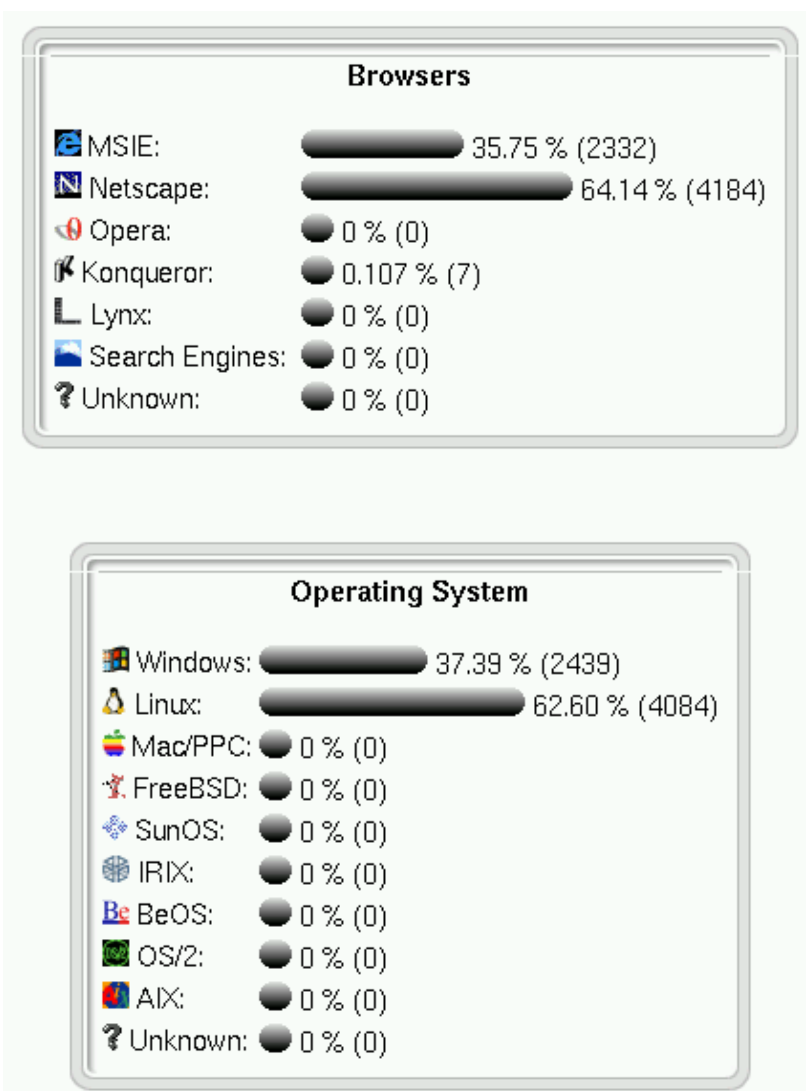
Sections:

This module is a classification system like the topics. The articles inserted in this module do not appear in the news module, they can be displayed on more than one page and for this reason they are able to accommodate articles which are big in size. Every section can be associated with a different image. The article even provides a display system for printable pages.

Statistics:

The statistics module provides basic statistical information regarding use of the site. The information varies from the total number of displayed pages , to the type of browser and operating system used, up to the number of users that are registered, the version of PHP−Nuke used etc...(see Figure 3−6)

Figure 3−6. Statistics module



Statistics module

Stories archive:

Archives all the articles by month enabling a chronological consultation. After having chosen the month, the corresponding articles are displayed with small flags besides them, visualizing the language they were written. Also in this plane it is possible to see the article in printable format and to send it to a friend. An inner search engine is also comprised as well as the display of details regarding the article, such as:

- number of comments
- number of readings
- score

Submit news:

The users or the simple visitors of the site can propose to the administrator an article that will then be examined and, if approved, published. The users do not have all the possibility of classification that the administrator does, in fact they can only decide the article's title, the topic, the language and the text. They cannot classify it or choose if it should appear in the Home Page or not. And they can neither decide to publish it on a temporary basis.

Surveys:

Enables the administrator to create a survey that will later appear in a block or in the survey list. The users can vote on this survey (not more than once in 24 hours), and eventually comment. Moreover it is possible to display the list of the previous surveys and to consult their final results.

Top10:

It lists the top 10 active ones of all our portal.

- 10 most read articles
- 10 most commented articles
- 10 most active categories
- 10 most read articles in the special sections
- 10 most voted surveys
- 10 most active authors
- 10 most read book reviews
- 10most downloaded files
- 10 most read pages

Topics:

Lists the main categories of PHP−Nuke. Once we have entered this module, we will be able, by clicking on the corresponding icon of the topic we are interested in, to carry out a selection of articles and in automatic mode, to see all the articles corresponding to this topic. We are also presented with a small search interface to finish our search in the chosen context.

WebLinks:

It is a collection of web links. This module has the exact same functionality as the Download module so there is no need to explain it any more.

Your Account:

It's the administration console for your "User Profile" (It only works for registered users), the implemented functions are (see Figure 3−7):

- It changes your info: enables management of your profile by changing your E−mail, Where you're from, AIM, ICQ, Avatar & Fake E−mail etc...
- It changes your Home: It creates a personalized menu (as a block) for navigation, the user can put in there whatever he wants (tests, links, images).

- Setup comments: Configures the display of comments, assigning display criteria .
- Theme selection: Changes the theme of the site, allowing you to choose between all available themes.
- Logout/Exit: It lets us exit from our current user profile, cancelling the cookie.

Figure 3−7. Your account

Your account

We then have a main menu that informs us of how many and which comments we have inserted and how many stories we have published.

Content:
: It is a module that lists all the categories (cross−sectional arguments to the topics, do you remember?), that lists, in the first instance all the present categories and once the category is selected, it lists all enclosed articles emphasizing the publication language.

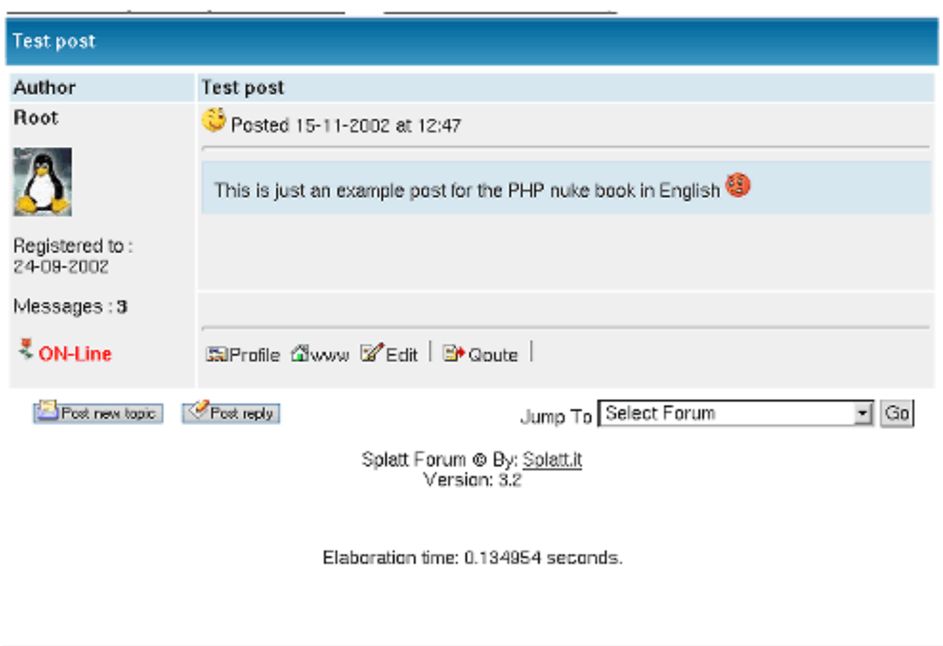
Encyclopedia:
: It is a system for creating one or more word dictionaries. In the first selection scheme it requires you to enter the dictionary (displaying even a small flag that indicates the language), after the click we are invited to choose the letter that corresponds to the searched word or to use the inner search engine of the encyclopedia, once found it's enough to click on the word to discover its meaning.

FAQ:
: It is an archive of Questions/Answers divided by category that can be consulted by the user as a first solution to his problems. He can divide the Questions/Answers by category in order to facilitate the consultation.

FORUM:
: In version 5,6 – 6.0 of PHP−Nuke the Splatt forum is present. We still do not know if it will be integrated in successive versions to Nuke. I suggest to use it anyway, since it is a mature application and has an italian support comunity. The functionality implemented in this forum (on the user side) are many (see Figure 3−8), the forums are divided by category, have a dedicated inner search engine, users can associate to every post (participation in the forum) icons relating to the argument, cast a vote on a discussion, see various icons according to the degree of attention degree that a specific

discussion has generated, see how many questions and answers a certain thread has received, see the profile of that user and many other functions...

Figure 3−8. Forum



Forum

3.2. Other non−installed modules

Indy News:

very interesting module that manages the "Attach−>File or Image" " function in every article. In fact, during the phase of an article insertion, it is possible to enclose the following:

- Image: In this case, inserting a GIF or a JPEG will result in a preview in the Homepage (the module will display it on the left, on the right there will be the topics icon), then, by clicking on "read all" (only if other text is present), or the image itself, it will appear in its original size.
- Other files: besides images it is possible to enclose also other files (for well−known filetypes, the corresponding icon wil appear in "read all", for others there will be a default icon). It is important to remember, for the attached files, to add a text in the extende text section, otherwise it will be impossible to display the link link "read all" that displays in the footer of the article the icon with the file and download info. The system was originally an adaptation of the IndyNews module for PHP−Nuke, has been created by the webmaster of bergamoblog.it for version 5,5, for the version 5.6 of PHP−Nuke the adaptation was done by Spaghettibrain.com.

Guestbook:

Allows the users to insert greeting messages (like a real guestbook), archiving their history in a way that, besides inserting new messages, it is possible to read all the messages of the other users too. Do not confuse it with the forum!

Chat:

There is a flash chat module that manages user's chatting in real time without the page refreshing. It is very interesting in that it comes with skins of various colors, thus giving the administrator the possibility to choose the one that is most suitable for the site.

Dmoz/ODP:
: Integrates the Open Source search engine Dmoz/Open Directory Project to the inside the PHP−Nuke portal . It is like having an integrated powerful motor like yahoo in your own graphics and pages.

3.3. The preinstalled blocks

Advertising Block:
: From this release on we have the possibility to insert our banners also in the blocks (buttons various of dimensions) being managed as if it were our own circuiting banner, counting clicks, impressions etc...

Content:
: Displays the most active content.

Encyclopedia:
: lists all active encyclopedias, clicking the link will bring us directly inside the list of terms of the chosen encyclopedia.

Forums:
: The forums block lists the last 10 posted messages and a search engine that executes a query on all the posts of the forum.

Last 5 articles:
: It lists the last 5 published articles offering information on how many readings and how many comments have been made.

Last 10 referers:
: It lists the sites from which the last 10 visits have arrived.

Ephemerids:
: It is a block that manages the recurrent events. It lists the events happened on the same date but in the previous years.

Reviews:
: It lists in a block the book reviews of the day.

Sections Articles:
: lists the active sections. Clicking one we will get to the corresponding article list.

Top 10 Downloads:
: It lists the 10 most downloaded files.

Top 10 Links:
: It lists the 10 most clicked links in the archives.

Chapter 4. Back end structure: administrator view

The administration page is reached by calling the page admin.php (www.yoursite.com/admin.php) and carrying out the login procedure inserting your user and password. (Remember that the normal users should not login from the page admin.php but from the appropriate module).

Once logged in, the administrator finds an interface that lists all the areas which can be acted upon (see Figure 4−1). If the administrator is a superadmin, he may work on all the areas of the site, if instead he is an administrator with limited powers he will see the links relative to the areas on which he is allowed to work. Through the preferences configuration we will be able to decide whether to display icons or just a textual interface. According to our choice either a textual or an icon administration interface will appear. Figure 4−1 shows the interface with the icons, as you can see.

Figure 4−1. Administration panel

Administration panel

Remember that when you write new administration modules you must also create the corresponding icon, otherwise, when in visual administration mode, only the textual link corresponding to your module will appear and you won't be able to click it.

In order to set up the graphical administration mode you must go to the preferences section and set up in "graphical options" the "graphical menu in administration" option to "yes".

4.1. The administration functions

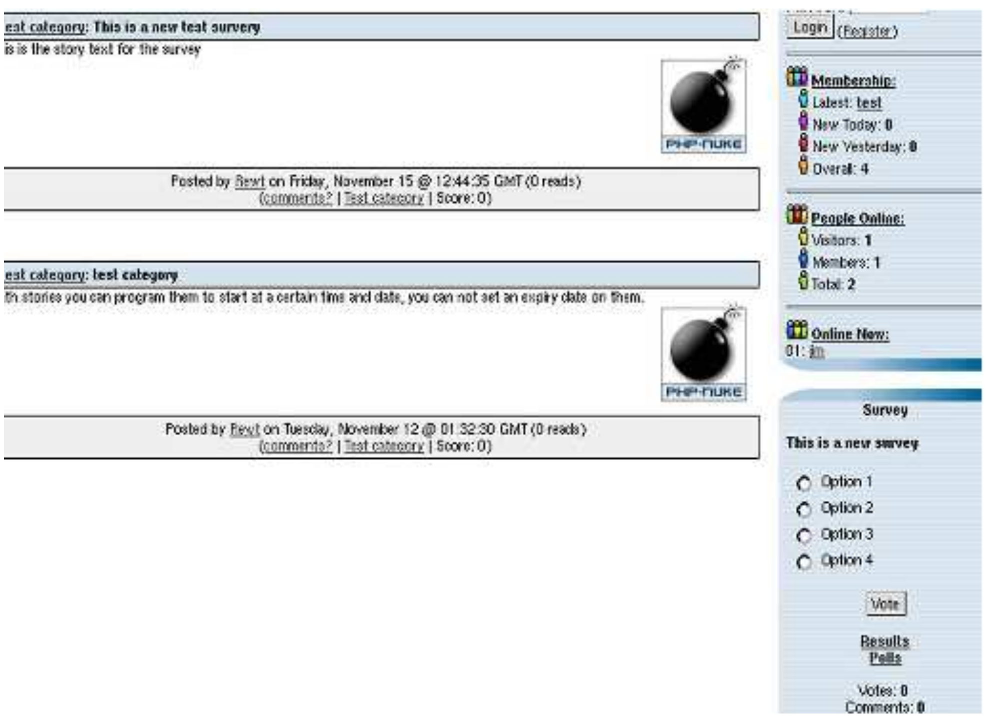
Function "add article":

It is the function that adds a new article to the News module. The options offered are many and will be analyzed here for one (see Figure 4−2):

- Title: inserts the news title.
- Topic: Determines which Topic will be associated with the article.
- Category:Determines which Category will be associated with the article.

- Publish in the Homepage: if this option is not selected, then the article will be displayed only in the topics or the relative categories and not in the main page of the news module .
- Activate comments: If it's not activated the users cannot comment on the article.
- Language: If in the preferences we have activated the multilingual option, we will be asked in which language we will publish the article. (e.g. if I publish an article in english, I will see it displayed only if I click on the small english flag in the languages block and so on...).
- Story Text: It is the text that appears in the preview.
- Extended Text: It is the text that appears when we click on "read all".
- Programmed article: The administrator is given the ability to choose when the article should be published, deciding on the publishing date and hour. It is not a neccesary function but it is very useful.
- Preview or send: Depending on the choice made here determines whether the article will be displayed in preview mode or directly published.
- Survey: It is possible to attach a survey to a specific article. In the case that this option is activated, when the reader clicks on "read all", a survey block like the one shown in the screenshot will appear.

Figure 4−2. Articles



Articles

Function "Backup DB":
: It is the function that allows us to create a backup file that contains both structure and content of the PHP−Nuke database. This is very useful in case our data gets lost. Once we click on "Backup DB", we will have to wait for the server to create the file. Waiting time varies from a few seconds to some minutes in the case of a large database. Once created, we will be asked to download the file. Remember to keep your backup in a safe place!

Function "Blocks":
: It's a very important function because it allows us to control the left and right columns of our portal. The scheme is presented with a list of the blocks that we have created, we can then activate, deactivate or edit them changing their position and order and assigning them permissions. We can in fact decide if a block should be visible by all, only by the registered users, or only the administrator.

We can also make the block visible only in a specific language.

Please note

This info is also present in Chapter 8

The PHP−Nuke blocks can be of 3 different types:

- RSS/RDF: They are blocks that capture news from other sites that are put at our disposal, the files are in a standard format suitable for reading the text contained in them. (For example the site Spaghettibrain.com gives news to other sites).
- Blocks of content: They are blocks which we insert simple HTML or text that will be displayed inside the block (see the following example)
- Blocks of files: They are PHP scripts that execute predetermined commands (see the following paragraph)

In order to create a new block that will be added to the list of available blocks, we must scroll down the page and position ourselves on "add block".

The title field is a common element for all and will be compiled in.

- If we want to create a RSS/RDF block we must choose the news source from the available list or add one by clicking on "setup". In this case we will supply the address of the file to read (this info generally will be supplied by the webmaster of the site from which we capture the news, or, if it is a site created with PHP−Nuke, simply by asking for the file backend.php of that site). The other fields will all be compiled in with the exception of "filename" and "Content".
- If we want to create a block of simple text instead, we will omit the field "RSS/RDF file URL" and will complete "Content" instead (Omitting "filename").
- If instead we want to include the PHP files that interface with a database or perform particular functions, then we will omit "Content" and "RSS/RDF" and will choose between the available files the one that will create our block. (If you want more info on how to create blocks, see Chapter 8).

Remember that before publishing a block a preview will be shown to us.

Content Manager:
: This function allowes us to add new categories and new content inside the content section. It is very similar to the articles but with less functions. A noteworthy feature is the possibility to add the tag

```
<! -- pagebreak -- >
```

in order to create a multipage article.

Downloads:
: It creates categories, subcategories and adds files to the download area. For security reasons, the system does not allow file uploads via HTTP, only their linking through their HTTP address. If for example the file *files.zip* is found in the directory *files* of our site, we would have to link it as *www.oursite.com/files/file.zip*. This allows us to link external resources also.

Edit Administrators:
: Enables us to add new administrators, defining their access levels. Besides having a super administrator it is in fact possible to activate only partial functions for the various administrators.

Edit Users:

From here it is possible to manually add new users and to modify existing ones, choosing their profile by typing the nickname in the appropriate form.

Encyclopedia:
: Allows the creation of multiple word lists (choosing also the language), after having created an encyclopedia we can proceed to the insertion of terms.

Ephemerids:
: Allows the insertion of recurrent events choosing the date and inserting a description.

FAQ:
: Allows the creation of the main FAQ categories and all related questions/answers.

Splatt Forum:
: The management of the forum is divided in 4 areas:

- Preferences: It manages the characteristics of the forum (For security reasons it's advisable to deactivate the option to mail in HTML).
- Categories and forum: defines the categories, the forums included in them, the moderators of every forum, levels of access etc... For a forum to be visible, you MUST activate at least one moderator otherwise the forum won't show up!
- Ranks: defines the attention thresholds for the forum. Upon reception of the nth post, aproppriate images will be attached to attract proper attention of the visitors.
- Users: moderator management through a complete list of the registered users.

HTTP Referrers:
: It displays the origin of the last accesses to the site.

Messages:
: It creates a central block in the Home Page in order to send selective messages to the users. The messages can be sent to only registered users, to non−registered users, to the administrator or one may carry out a selection by language.

Modules:
: Allows the management of the modules installed. The modules can be activated, deactivated or be assigned read permissions. A module can be world−readable, readable only by the registered users, or only the administrator.

Newsletter:
: the PHP−Nuke administrator can send a newsletter to the registered users who have consented to receive them or send them to all the registered users. Attention to spam!

Optimize DB:
: Optimizes the data increasing database speed.

Preferences:
: This subject will be treated in Section 4.2.

Book Reviews:
: It allows us to insert book reviews. In every book review it is possible to cast a vote, a link relative to the subject and finally an image that represents the content.

Section Manager:
: it manages the sections and contents thereof. It is possible to associate an image to the section subject, just as it is in the topics. It is possible to add articles to the sections selecting the aproppriate category through a radio button, to divide long texts using the <! −− pagebreak −− > tag and to edit or cancel already added sections.

Articles:
: Manages articles inserted by third parties. It is the moderation area of the News module that we have already analyzed in this section.

Survey/Polls:

Creates a new survey for the site, edits or cancels old ones. It is possible to insert up to 10 different answers to every survey. In the context of the creation of the survey it is possible, on the same page, to publish a news article that announces its creation.

Topics:
: Allows you to create new topics and the association, through a pop−up menu, of corresponding images.

Links:
: Allows us to edit links published by others, create categories for archiving the links, eliminate links, see user messages informing us of any broken links through an interface very similar to the downloads section and add new links.

Logout/Exit:
: Exits from the administration area rendering the cookie invalid. It is good practice to login logout after having finished working with PHP−Nuke. For security reasons.

4.2. The Preferences Page

Here are the parameters needed for the configuration of the config.php file through the admin/preferences area:

- Site name: It corresponds to the title tag, it is what appears up in the right bar of the browser. It is very important for the search engines.
- Site URL: It is the internet address of your site.
- Logo: It is the logo of your site. In standard themes and themes not modified much it is the Logo that appears up on the left.
- Slogan: It is equivalent to the description tag, it's also very important for the search engines.
- Beginning date of the site: It's the date that appears in the statistics module.
- Administrator e−mail : it is the e−mail which will receive the notifications for article insertions by third parties and the mails from the "contact us" module.
- Articles in Top Page: Specifies the number of news articles that can be displayed in the main page of the news module.
- Articles in Home: Specifies the number of news articles that can be displayed in the home page of the site (if the news module is the main one).
- Stories in old articles box: Specifies the number of news articles that can be displayed in the old articles box.
- Activate ultramode: specifies if other sites can take news titles from our site.
- Allow anonymous to post : Specifies whether anonymous users can write comments.
- Default theme: defines the default theme of the site.
- Select language: defines the default language of the site.
- local time format: defines the format of the local time (Depends on the server, if it is running linux, this is controlled in */usr/share/locale*)
- activate the multilanguage characteristics: choose whether the site should support a multilingual functionality or not
- display the small flags in place of the list: if the multilingual option is activated then this decides whether the block should display the small state flags in place of a list of language names.
- activate banners : sets up the option to use banner rotation on the site.
- For the footer of the page, imagine we have to insert 3 texts in a table that is 100% wide and positioned centrally:
    - Page Footer Line 1: first text to insert
    - Page Footer Line 2: second text to insert

    - Page Footer Line 3: third text to insert
- Backend title: The title that will appear in the file from which other sites get our newsfeed.
- Backend language: language for the newsfeed file.
- Notify new insertions via e−mail: It defines whether the administrator gets an e−mail when a user inserts a news article.
- E−mail address to send the message: e−mail to which the notification message will be sent.
- E−mail subject:
- E−mail message: notification text (e.g. "you have received a new article").
- E−mail account (From): who has sent the message.
- Moderation type: chooses whether a moderation should be set up for the comments or not.
- Comment limits in Bytes: sets up the maximum size for the comments.
- Default anonymous user name: assigns a name to the persons who chose to remain unregistered.
- Graphical administration menu: sets up whether to have icons or text in the administration area.
- Minimal length for the user's password: for security reasons it is advisable to set up a rather long password.
- Activate referring HTTP: whether statistics regarding the origin of the visits should be gathered.
- How many referrers you want maximally: the maximum number of statistics pertaining to the origin of the visits (max 2000).
- Activate comments in surveys: Whether comments are allowed in the surveys or not.
- Activate comments in articles: Whether comments are allowed in the articles or not.

Chapter 5. How to install PHP−Nuke

The screenshots regarding the installation procedure, contrary to those of the other chapters, refer to the Windows platform. This is so in an effort to reduce the number of misunderstandings and ulterior help requests from the Windows community. (Judging from the feedback we receive, the Linux and FreeBSD community seem to be more able to deal with installation problems in this respect).

For the installation of PHP+ MySQL+ Apache, PHPMyAdmin etc. refer to Chapter 11 where you will find notes and links to useful tools in order to emulate PHP−Nuke on your client.

These instructions are valid for all PHP−Nuke versions up to 5,6, from 6 onwards the installation procedure will be much simpler.

5.1. Installation process

5.1.1. Download

Well... there is little to say here, it suffices to go to a site that holds the files and download them. There is only a small remark to make: if you use Windows and download a version comprised of a file with the ending tar.gz, do not worry, your Winzip supports it without problems. Once downloaded, extract it and "throw" all its contents in a folder you created for this purpose and you call "Nuke5" or whatever (sites from which you can download PHP−Nuke are: phpnuke.org, www.spaghettibrain.com etc.).

5.1.2. Upload through FTP

Well, now what remains is just to upload the files to the interior of our main server directory that resides on our provider.

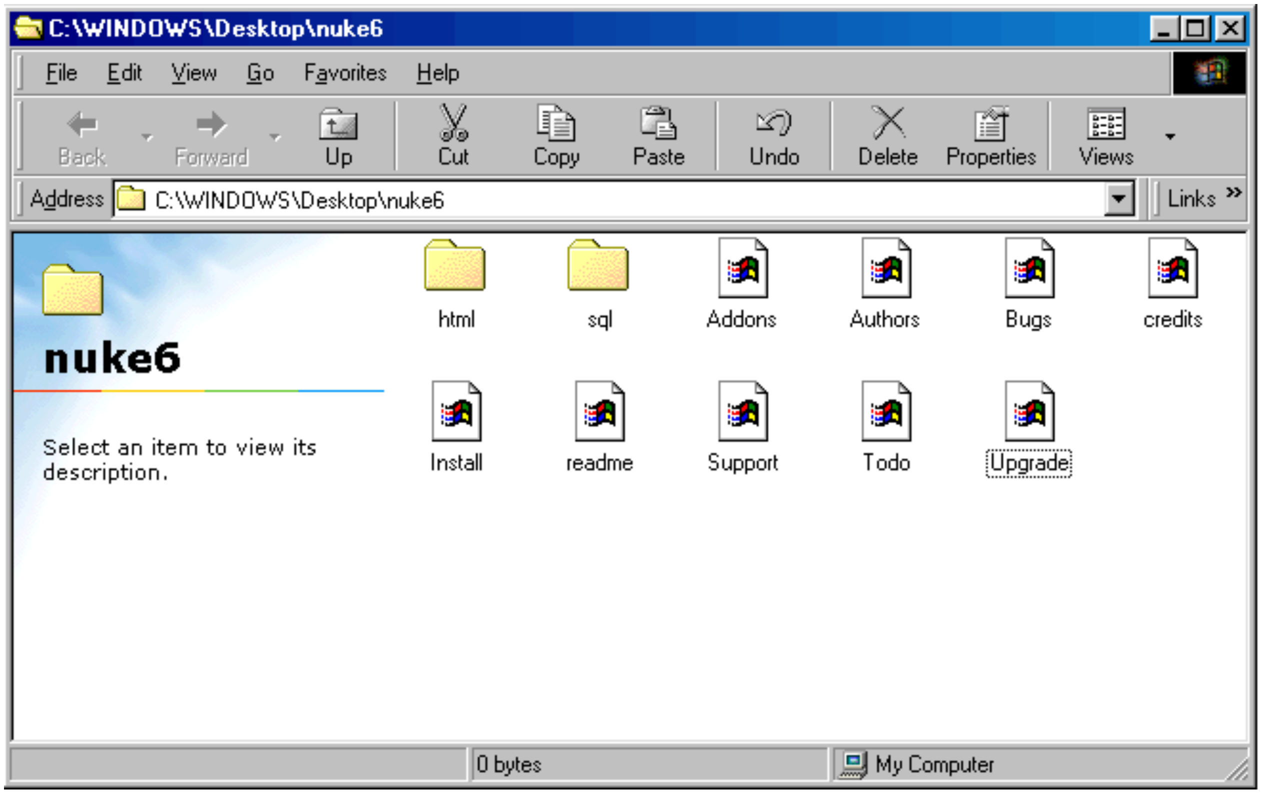
Attention!

It is highly recommended, prior to any installation steps, to verify that your provider supports PHP and MySQL.

Tip

Don't upload all extracted files. After extraction, you will find a structure similar to the one depicted in Figure 5−1.

Figure 5−1. PHP−Nuke 6.0 file structure



PHP−Nuke 6.0 file structure

You don't need to upload all the files from the folder, in the main directory of your web presence you will only need to upload the contents of the html folder (so just do a doubleclick on the html folder and upload everything that is inside it).

5.1.3. Formulation of the file permissions

Important

This process only really applies if your PHP−Nuke will be installed on a Linux/Unix server, if instead you will install it on Windows operating systems you don't have to do anything.

Setting up permissions on files serves the purpose of having them execute only certain operations (write, execute etc.) when called. Setting them up correctly is important for PHP−Nuke to operate in its full functionality.

The permissions to give are the following (for the base permissions, see Section 10.1 in the context of security):

- Files: 666
- Directories: 777

With WS_FTP you must select the files or folders on which you want to impose the permissions and, with the right mouse key, to select the option CHMOD.

Once you set up the permissions on all folders and all files, this procedure will cost you some time, but it is very important to carry out. Moreover, you will have to do it every time you insert a new file or module to your PHP−Nuke.

5.2. How to install PHP−Nuke through PHPMyadmin

5.2.1. What Is PHPMyadmin

PHPMyAdmin is an visual system for the management of a MySQL database (see Figure 5−2). It is written in PHP and serves to display the contents of the databases on the server (or client) on which MySQL is installed. Through this interface you can create new databases, modify existing ones and modify the contents of single fields.

Figure 5−2. PHPMyAdmin start screen

PHPMyAdmin start screen

5.2.2. How to install the DB of PHP−Nuke with PHPMyadmin

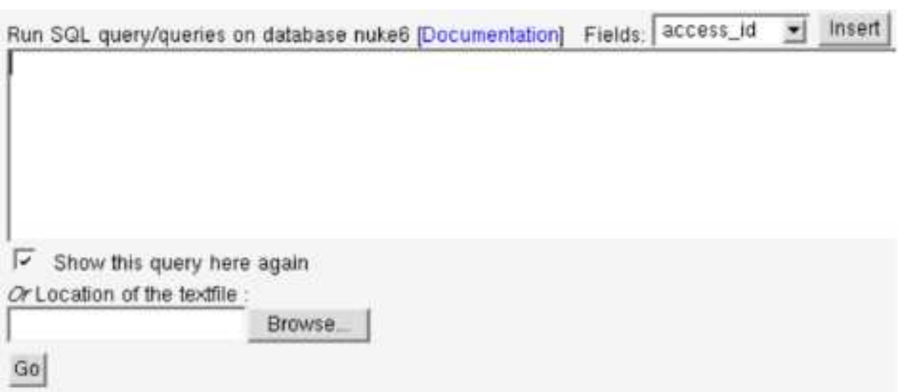
Clicking on the left bar, depending on the database you selected, you will see a list menu coming up, showing the structure of the database (and, on the same time, the central page will show the enlarged structure of the database), with a series of options, all of them in the bottom of the page (see Figure 5−3). It is these options we are interested in when installing PHP−Nuke.

Figure 5−3. PHPMyAdmin table selection

PHPMyAdmin table selection

What you have to do now, is to click on "browse" and go search for the .sql file that contains the instructions that build the structure of the PHP−Nuke database (see Figure 5−4). Once found, it suffices to click on "Go" and the database will be installed. Of course, if there are errors, they will be reported at the end of the installation procedure. And of course, the same holds for the message "operation succeeded".

Figure 5−4. PHPMyAdmin: SQL query

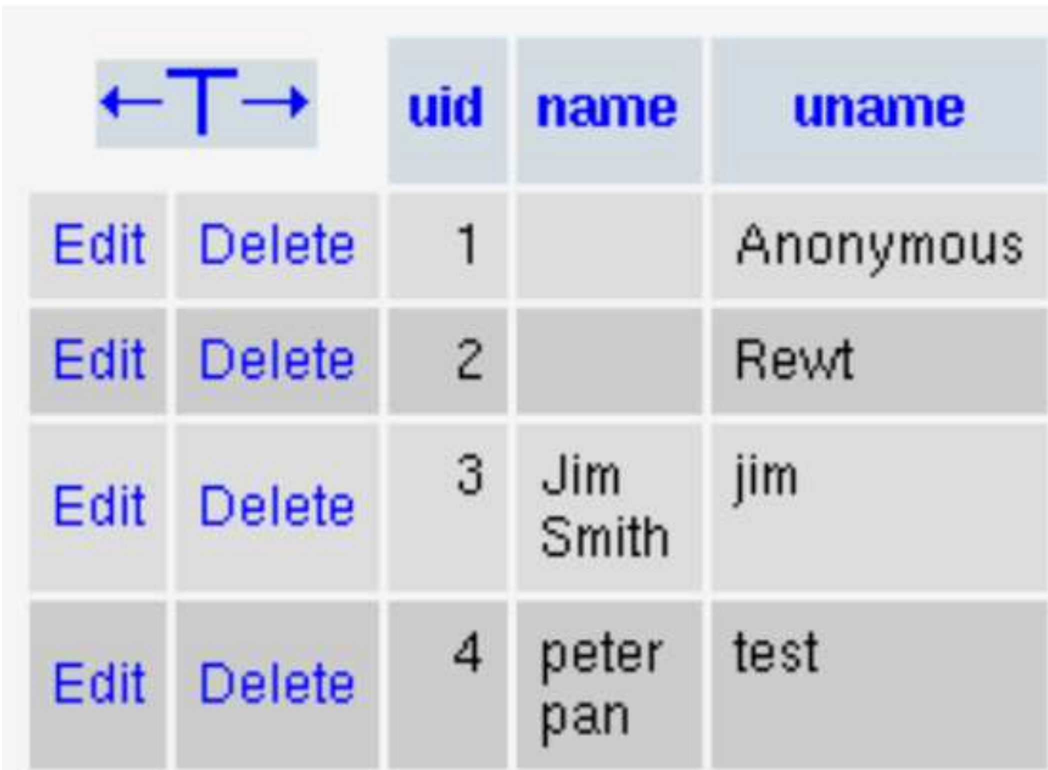


PHPMyAdmin: SQL query

Tip

Other options of PHPMyAdmin that are not relative to the installation of PHP−Nuke can be found in the tutorial published on www.spaghettibrain.com and in Section 11.3.

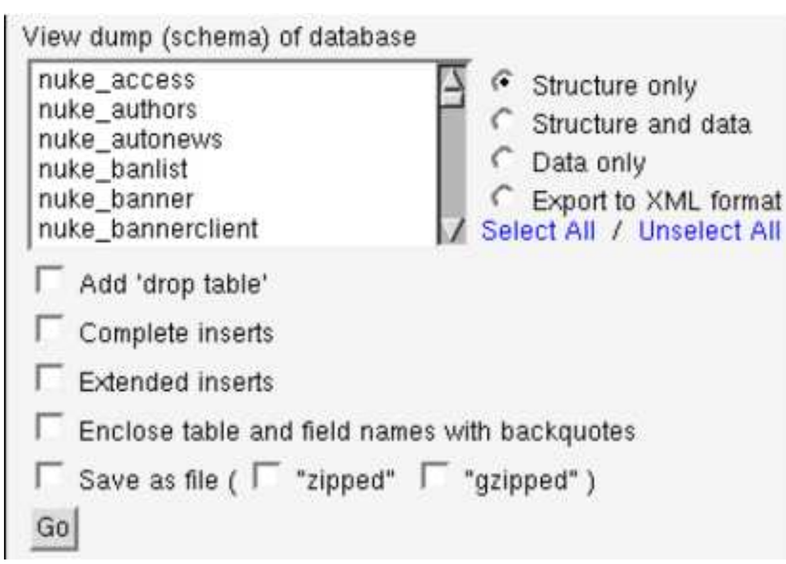
Figure 5−5. PHPMyAdmin: table data



←T→		uid	name	uname
Edit	Delete	1		Anonymous
Edit	Delete	2		Rewt
Edit	Delete	3	Jim Smith	jim
Edit	Delete	4	peter pan	test

PHPMyAdmin: table data

Figure 5−6. PHPMyAdmin: database dump



PHPMyAdmin: database dump

5.2.3. The config.php file

Ok, the last thing that remains to do before starting with the management of your site, is to configure the file config.php This is important because it sets up a connection between the PHP files of PHP−Nuke and the MySQL database that manages it.

There is only a few parameters to configure. When you open the file "config.php" you will see the following near the top:

```
$dbhost = "localhost"; ❶
$dbuname = "root"; ❷
$dbpass = ""; ❸
$dbname = "nuke"; ❹
$system = "1"; ❺
$prefix = "nuke"; ❻
$user_prefix = "nuke";
$dbtype = "MySQL";
```

❶ In place of "localhost" you will have to put the host/server that the database is installed on.

❷ In place of "root" you must put your username.

❸ You will have to insert your password here.

❹ In place of "nuke" you will have to insert the name of your database here.

❺ Leave it to 0 if on Linux/Unix systems, put 1 if on a Windows server)

❻ I recommend this to be left to its default value, "nuke", is the prefix that goes in front of the name of every database table.

Let's do an example:

- Host DB: 212.110.12.297
- User DB: Pippo
- Password DB: Topolino
- Database Name: Orazio
- Operating System Used: Linux (what else!!! :−))

The file config.php should then look like:

```
$dbhost = "212.110.12.297";
$dbuname = "Pippo";
$dbpass = "Topolino";
$dbname = "Orazio";
$system = 0;
$prefix = "nuke";
```

Attention!

This is case sensitive! Remember to use The Capital Letters!!! On Linux systems, if you write a user name or a password without taking care of letter case, the system will not allow you to log in.

Ok, we are done, the only thing that remains to do is to enter the administration section (www.yoursite.com/admin.php). The very first time you will log in using "God" as username and "Password" as password. I recommend you to change these as soon as possible.

5.2.4. Resources

If you are looking for PHP−Nuke hosting, Spaghettibrain.com has custom offers for you... give us a visit! Spaghettibrain.com is the italian support site for PHP−Nuke, there you will find modules, security patches, support forums and a lot more. Come for a tour soon.

Chapter 6. Architecture and structure

The structure of PHP−Nuke is organized in to modules, all the files are managed by other files that are located in the PHP−Nuke root directory and include, according to the parameters passed to them, the intended module.

These tasks are carried out from only 3 pages:

- index.php : In order to display the main page
- modules.php : In order to include the internal modules.
- admin.php : In order to include the administration interface.

It's not possible to call a module by calling a direct path to it. This is so in order to make installation easier, render the graphics management more independent (otherwise we would have to change the path of the images each time we position ourselves in an internal directory), have only a few files in the root directory and render the system more secure.

Everything is being called, as I said, through parameters (strings) passed to the "modules.php" file which specify which files are to be included. If for example we want to call the Topics module, the string to be passed should be http://www.yoursite.com/modules.php?name=Topics

The command that is sent this way is "includes in the page created by modules.php the output of the file index.php that is found in the folder modules/Topics/".

The other files present in the PHP−Nuke root directory are:

- auth.php: Manages authentication through the cookies.
- mainfile.php: Contains all the necessary functions for the management of PHP−Nuke
- header.php: manages the variables that are related to the header (inclusion of metatags, Javascript...)
- footer.php: variables related to the footer.
- backend.php: manages the output of the news that can be captured from other sites
- ultramode.txt: ditto
- robots.txt: contains instructions for the search engines informing them which folders not to index

6.1. Directory structure

Admin:
: Contains 4 subdirectories (links, language, case, modules) that manage the various administration modules. The folder that accommodates the operating files is modules/admin/ .

Blocks:
: Contains all the block files for all of our available blocks.

Images:
: Contains all the images relating to PHP−Nuke, for example in the folder "topics" we will find archived the images of the topics that will appear in the news, in "banners" all banners in rotation etc...

In includes
: are all the files that are necessary to particular management situations, these files do not work independently but are included in other files, mainly in mainfile.php and header.php. The files are:

- counter.php : serves to identify the users based on the operating system used, the browser, the page of origin, date of the visit...
- javascript.php : includes all necessary Javascript (if you need particular Javascript code, include it in this file).
- meta.php : manages the keywords to pass to the search engines and other parameters of the header. It is an optimal system for learning how to create keywords and position the site with a good ranking in the search engines.
- my_header.php : manages the disclaimer message in the homepage.
- sql_layer.php : serves to manage the database abstraction layer. Transforms SQL instructions to the language of the chosen database. Remember that PHP−Nuke can manage various databases.

Language:
: contains translation files for the basic PHP−Nuke module. The language file naming convention is lang−english.php.

Attention!

The translation of the modules must be inserted in the appropriate folders (modules/languange) and not appended to these files, as it was done till now.

Modules:
: The modules of PHP−Nuke comprise all the functionality that one can add to it.. In the Modules folder we insert the folders of every new module.

Themes:
: Here we add the graphical interfaces known as "Themes", every folder has the name of the corresponding theme and contains a main file called theme.php and all other support files.

Upgrades:
: contains only the files that serve to upgrade the system from a previous version to a newer one..

6.2. Main page management

The file "index.php" is very simple one, it has the task to load the main PHP−Nuke page the module that was chosen as the default one to appear in the main page of our site.

Here in detail is what happens when the page "index.php" gets loaded:

- the mainfile.php is included,
- a database query is made in order to see which module was set up as the default one,
- the origin of the visitor is checked (if he comes from a site that links us, this fact will be inserted in a table in the database).

Various checks are also made and error messages are defined in case the connection to the database fails. This avoids (in part) the error messages from the PHP preprocessor. Even if problems come up, the page will be presented in a standard design and the error message will be definable from the inside.

6.3. Module management

For reasons of order, the modules are managed through the files present in the subdirectories that contain them, every module has its own folder in the interior of the folder "modules".

In order to be loaded, the module files get included in the modules.php file by passing it the aproppriate parameters. The main page of each module must be called index.php, the other possibly pages possibly present in the module will have an additional variable in the inside of parameter strings by which they are called..

For example in the AvantGo module (see) in order to load the index.php file, it is enough to pass the module name to the parameter string (by default, the file that will be searched for is index.php):

```
modules.php?name=AvantGo
```

If we instead wanted to call a page other than the default index.php (say, print.php), the string we will have to pass is :

```
modules.php?name=AvantGo&file=print
```

that is the file variable with a value (print) that corresponds to the name of the file we want to load without the .php extension.

Inside the folder modules/nameofmodule there is also a subfolder called "language". In this fashion we manage in a simple and immediate way the multilanguage functionality inside the modules.

The modules.php file works this way:

- Includes mainfile.php
- Verifies whether the module is active or not
- Verifies whether the string passes a file name different from index.php
- Verifies the permissions of the module (whether everybody can see it, or only registered users, or only the administrator).

6.4. Administration management

Admin:
: contains 4 subfolders (links, language, case, modules) that manage the various administration modules. The folder that contains the operating files is admin/modules, here we have the files that execute the different admin operations.

The folder admin/links instead says which admin module has to be called and puts a link in the admin area for that module.

Example (administration module for the FAQ):

```
if (($radminsuper==1) OR ($radminfaq==1)) {
adminmenu("admin.php?op=FaqAdmin ", "" _FAQ." ", "faq.gif");
}
```

This module:

- Verifies that administration rights are present (this module may be managed by either the superadmin or an administrator that has been qualified to do so on the FAQ level),
- passes a case (op=FaqAdmin) that says to the admin.php file (that includes all the admin modules)

which module to load, associates a value in order to translate the term "faq" and associates an icon for the visual administration (faq.gif).

The folder admin/case instead serves to define which module to use in certain cases. This is important when, using the same admin file, one needs to carry out different operations according to the case passed:

```
Case1 = insert
Case2 = cancel
etc...
```

In fact it says which module to load on verification of a case. For example, in the module faq the cases are many, let's consider only the last 2:

```
case "FaqAdmin":
case "FaqCatGo":
include ("admin/modules/adminfaq.php");
break;
```

Both cases load the file adminfaq.php but they make it carry out different operations. The first one loads the file in the default scheme, the second one instead gives the O.K. for the insertion of a new category. This happens through a string like

```
admin.php?op=FaqAdmin
```

in the first case and

```
admin.php?op=FaqCatGo
```

in the second.

Chapter 7. Customising PHP−Nuke themes

7.1. Structure of a PHP−Nuke theme

Making your own personal graphical theme for your site is very important so that you don't have another PHP−Nuke clone, if your site looks the same as other sites it dosn't make you, the Webmaster look very professional. Personalising the portal starts from the graphical side of things. Knowing how to put your hands on a PHP−Nuke theme means being able to play with all of the graphical elements that we can use. The example theme we will use in this chapter is the NukeNews theme, made by Francisco Burzi for PHP−Nuke. It's a theme composed of alot of HTML files included in theme.php. This is a very good solution that permits you to manage the graphical part of the theme through an editor like DreamWeaver using the least amount of PHP code.

The NukeNews theme is structured in this way:

- theme.php: It manages the main functions of the variables for the background colors.
- tables.php: it manages the functions opentable(); and closetable();
- header.html: It manages the header for your site
- footer.html It manages the footer for your site
- blocks.html It manages the blocks
- center_right.html Manages the outlay of the page.
- center_left.htmlManages the layout of the page.
- story_home.htmlManages the layout of the page.
- story_page.htmlManages the layout of the page.

These files are included in the functions specified in theme.php We then have a style sheet called style.css (style/style.css) included in the header.html file in our theme folder. For convention, the style sheet must always be called style.css and must always be contained in one folder called "style" inside of our theme folder. The images generally are grouped in a folder called "images" that is always found in our themes folder.

The folder structure of the NukeNews topic will be :

- themes/NukeNews
- themes/NukeNews/style/
- themes/NukeNews/images/

Always remember that case is important, you must respect the difference between UPPERCASE and lowercase for compatibility with any Unix systems.

The theme.php file is the heart of all PHP−Nuke's graphical management.

The HTML file inclusion does not happen in all kind's of themes, some programmers include all the HTML in the theme.php file, but including it separately, solves many problems such as HTML formatting, that would otherwise be included in the PHP code. It also gives us the possibility of editing all with a visual editor (WYSIWYG).

The theme.php is the file that creates the managing functions of all of PHP−Nuke's components (header, footer, central parts, block...).

The themeheader() function manages the site header. It is composed of various tables that form the heading, and sometimes also defines some elements of the body tag that are not included in the style sheet and the variables that are placed inside the included html files.

Example:

> The variable $theuser is defined inside of the themeheader() function and is then recalled in the header.html file in a table:

Code in theme.php (that defines the $theuser variable)

```
if ($username == "Anonymous") {
$theuser = "   <a href=\"modules.php?name=Your_Account &op=new_user\">Create an account";
} else {
$theuser = "  Welcome $username!";
}
```

Code in header.html (that visualises the $theuser variable)

```
<td width="15% "nowrap >< font class="content" color="#363636 " >
<b> $theuser </b></font></td>
```

The themefooter(); function manages the footer of our site.

It has some interesting elements we have to analyse:

First of all, it identifies if the visualised page has got the $index variable set equal to 1, in this case we will also insert the right blocks on our page, but if instead $index==0 then the right blocks will not appear on our page.

It then defines the footer messages (which are captured from config.php) and inserts all them in a variable that is recalled from the footer.html file.

The function themeindex() manages the news in Home Page and formats them adding elements according to various cases using the function "if". It also includes the story_home.htm file.

The function themearticle() instead manages the internal news page (that we can see by pushing on "Read more..."; remember that the layout part in this case is managed including the story_page.htm file, but the blocks that must be included (i.e. the article's survey, correlated links etc.) are defined by the news module.

The function themesidebox() instead manages the layout of the box that we create or that we find already made (see Chapter 8), it too includes a file called blocks.htm that defines the style and the layout.

We have ignored an element of the file theme.php. These are the variables that format the text, some of them are inserted in css (the style sheet) but others are instead defined at the beginning of the theme.php file.

Let's see the variables from the NukeNews theme:

```
$bgcolor1 = "# efefef";
$bgcolor2 = "# cfcfbb";
$bgcolor3 = "# efefef";
$bgcolor4 = "# cfcfbb";
$textcolor1 = "# 000000";
$textcolor2 = "# 000000";
```

As you see the expression values of these variables are in decimal format.

Define your site colours – $bgcolor2 is generally used for table edges as you can see in the function opentable(), $bgcolor1 is used for table background. The others two background variables use the same criteria. $textcolor1 and $textcolor2 are used to format the text colour.

Now we have to examine what is included in the tables.php file. This file creates 4 functions (opentable(); closetable(); opentable2(); closetable2();) that include HTML tags that open and close tables in a predefined way.

It is very easy to use when creating modules (see Chapter 9) , you don't have to rewrite the HTML every time you want to create a table but it's enough with the following syntax:

```
opentable();
echo "Content of the table";
closetable();
```

In this way you've created a table in a fast and effective way. But how is this function structured? We will examine first opentable(); then closetable();

Please note

These are php functions so you have to respect the HTML syntax inside php adding / before every " (ie align="left" must be written as align=\"left\")

```
function OpenTable() {
global $bgcolor1, $bgcolor2;
echo "<table width=\"100% \" border=\"0 \ "cellspacing=\"1 \" cellpadding=\"0 \ "bgcolor=\"$bgcolor2 \" >< tr >< td > \n ";
echo "< table width=\"100% \" border=\"0 \ "cellspacing=\"1 \" cellpadding=\"8 \ "bgcolor=\"$bgcolor1 \" >< tr >< td > \n ";
}
```

The syntax is very simple, isn't it?

- The function is opened
- Necessary variables are called ($bgcolor1, $bgcolor2)
- We open a table 100% wide and we define the background colours for it
- Open Line, Open Column
- We insert a new table 100% wide (for the edges)
- The width and height characteristics are defined.
- line column

We stop on the column because it's here we will insert the table content (In fact opentable is where we start from to close this table!)

```
function CloseTable() {
echo "</td ></tr ></table ></td ></tr ></table > \n";
}
```

In fact...

- The function is opened
- You close Column, You close Line

* You close Inner Table
* You close Column, You close Line
* You close External Table

Its easy to construct HTML functions with PHP, isnt it?

7.2. Modifying the HTML template

7.2.1. Example creation of HTML file to include in the theme

We will not analyse the HTML syntax of all the files, I think it's better that you understand the principles that are used and that you learn a few tricks that allow you to use a visual editor such as DreamWeaver.

Example:

The block is created in this way:

```
<table border="0 "cellpadding="1" cellspacing="0 "bgcolor="#000000" width="150 "><tr><td>
<table border="0 "cellpadding="3" cellspacing="0 "bgcolor="#dedebb" width="100% "><tr>< td align="left" >
<font class="content "color="#363636" ><b> $$title </b ></font>
</td></tr></table></td></tr></table>
<table border="0 "cellpadding="0" cellspacing="0 "bgcolor="#ffffff" width="150 " >
<tr valign="top "><td bgcolor="#ffffff">
$content
</td></tr></table>
< br >
```

As you see, we create a table of fixed width (in our case 150) and we assign it some colours (for the background etc...). We also assign two variables ($title and $content) that will, once included in theme.php, load the title and the content of the block. It would have been useful, for a code cleaning reason, to define background values in the css tables instead.

To have all the necessary cases to get the conclusions of this chapter and to write a pair of rules, we have to analyse a very simple module that includes a case we have still not mentioned, the images management:

```
</td><td><img src="themes/NukeNews/images/pixel.gif "width="15" height="1 "border="0" stop = "">
</td><td valign="top" width="100%">
```

The analysed code is a spacer that adds a space of 15 pixels, but where do we go to recover the image? Which path do we have to assign it? Remember that the theme.php file is included in root, so the image path must start from root to the indicated theme. The path to the image "pixel.gif" is "themes/NukeNews/images/pixel.gif"

Attention!

When you add images in an automatic way by using a visual editor, the path will be only images/image.gif, and you will have to correct it by hand adding the correct path.

Another device is to assign, in theme.php, is a variable to the name of the theme to make it independent from eventual changes of the folder name. So for variable $nameoftheme = " the NukeNews " the image route syntax will be:

```
<img src="themes/$nameoftheme/images/pixel.gif">
```

7.3. Theme construction: the rules to follow

Considering that the example is always on NukeNews, I suggest you always use this theme as an example, by using the HTML template without directly inserting too many tags in the PHP code, you save time and increase the sites stylistic effect by easily making changes across your site.

1. When you add images in an automatic way by using a visual editor, the path will be only images/image.gif, and you have to correct it by hand with the right path.
2. You can insert variables in the HTML that will then be called by PHP − it is important that they are inserted in the global part of the function that will include the file.
3. The visual editor has the defect to open and close the tables, correcting what it thinks is an error. Pay attention because sometimes, in the html files we use, a table is not closed because it will be closed by the successive html file. For example, as often happened, DreamWeaver would close a header by making a mistake in the tables. The header must be closed in the following way:
   ```
   <td bgcolor="#ffffff "width="150" valign="top ">
   ```
   Where this table is that one that includes the right blocks.
4. Try to validate the code as much as possible and to use the style sheets as much you can. In this way you will save a lot of time when you will modify colors, fonts etc... In order to validate the code go to www.w3c.org

Chapter 8. Creating blocks

8.1. The characteristics of the various types of blocks

There are 3 different types of PHP−Nuke blocks:

- RSS/RDF: They capture news that's available from other sites in standard reading format, i.e. text (For example the site spaghettibrain has a lot of news that are at other sites' disposal).
- Blocks of contents: blocks in which we insert simple text or HTML text that will be then displayed inside the block (See following example)
- Blocks of files: They are PHP scripts that execute fixed commands (see Section 8.2)

In this paragraph we will see a simple example of how to insert the links and the text in a text block. If you already know a little HTML there is no point in following this example.

We suppose you want to insert a block with text and a link to 3 different sites:

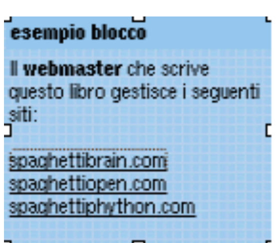
The Webmaster who wrote this book manages the following sites:

- spaghettibrain.com
- spaghettiopen.com
- spaghettipython.com

The text will be formatted in this way in order to be inserted in the block (see Figure 8−1):

```
<B> webmaster </b> who writes  this book manages the following sites: <b ><br>
<a href="http://www.spaghettibrain.com">spaghettibrain.com</a>
<a href ="http://www.claudioerba.com">claudioerba.com</a>
```

Figure 8−1. Block example



Block example

Some Small HTML lessons:

<b>
: It is for bold text, it opens a tag. All words that we write after this tag will be bold until </B> (which closes the tag).

*
*
: It is for a page break, it does not need to be closed.

<ahref="http://siteyouwant.com">SiteName

is used to open the http://siteyouwant.com.

8.2. How to create a new block

To create a block of the third type, i.e. a php script that interfaces with the database and extrapolates the data, first of all we have to understand how these blocks are structured. They are contained in a folder called blocks, the name of the block must be block−blockname.php. It is very important all blocks start with "block−" . Every block in which the name will start with block− will be included in the screen of the blocks that can be activated. We will find in the blocks administration menu all the available blocks in the file_name drop−down list. If not assigned by the administrator, the name will be the same that follows block− We can't use break spaces in a block name, they must be replaced by using underscore _ . All the blocks that will respect these rules will be inserted in the blocks admin menu.

8.2.1. How to create a block, theoretical approach:

You have to follow these rules when creating a block:

- In every block you create you have to insert the following code at the beginning:
  ```
  if (eregi("block-Name_of_Block.php", $PHP_SELF)) {
  Header("Location: index.php");
  die();
  }
  ```
  By using this code you protect the file avoiding users approaching it directly from the blocks folder, and the block will be displayed only when selected from your site.
- In the blocks you can include everything you want, Perl, java, php, flash etc...
- All the block output must have a value that can be obtained from the variable $content.

Remember that you have a limited amount of space in the block, pay special attention to the layout!

Warning

In order to have W3C standard compatible blocks, Francisco Burzi says:

> "To respect the W3c standards for HTML 4.01 Transitional, it is very important that you replace all "instances of &" in the URL by the tag "& "

For example the URL:

```
<a href="modules.php?op=modload&name=FAQ&file=index">
```

must be written:

```
<a href="modules.php?op=modload&amp;name=FAQ&amp;file=index">
```

and don't use, for example, the tag "li" (used to create a list), but leave that in the style sheet (CSS) which will make it for you.

The background for the tables and font etc., are better left to the style sheet (CSS).

We will now see how to construct a block starting from the beginning.

8.2.2. How to create a block, a practical example

We will make a very simple block that shows the pages visited in our site the day before. We'll have a single query and a single value, in order to make things easier. Our block is called "hits", so the complete name of the block will be block−hits.php

First of all, we open the php tag

```
<?php
```

Then we insert the protection script we've seen before:

```
if (eregi("block-hits.php", $PHP_SELF)) {
Header("Location: index.php");
die();
}
```

And now we insert the variables that we want to call (in this case the parameter $prefix and $dbi, which handles the database abstraction):

```
global $prefix, $dbi;
```

Now we continue inserting the query that reads from the database how many pages were seen in our site: (the instruction would be "read the first line value of the table nuke_counter in the cell count")

```
$result = sql_query("select count from "$prefix."_counter order by type desc limit 0.1", $dbi);
list($count) = sql_fetch_row($result, $dbi);
```

Finally, we pass the "$content" variable that will be echoed by the block and close the PHP tag:

```
$content. = $count
?>
```

Our complete script will be :

```
<?php
if (eregi("block-hits,php", $PHP_SELF)) {
Header("Location: index.php");
die();
}
global $prefix, $dbi;
$result = sql_query("select count from "$prefix."_counter order by type desc limit 0.1", $dbi);
list($count) = sql_fetch_row($result, $dbi);
$content. = $count
?>
```

Chapter 9. Creating modules

9.1. Module structure

The PHP−Nuke modules are PHP−written applications that manage the central part of the site. For example, the "News", "Forum", "Members List" etc. are all modules. Each module, based on it's complexity, is structured using one part for the user's and another for the administrator, in this case there are some contents we have to modify. It's all managed from the modules.php file which alone, carries out the job of authentication and management of the access rights on that module. The modules.php file checks and verifies if the module was activated or not, and verifies the access rights. This saves us a lot of work because we don't have to insert these controls in every module we create.

We refer you to what is written in Section 6.3 to be more exhaustive.

For example in the avantgo module, in order to load the index.php file, it's enough to pass the module name to the parameter string, the file that will be searched for is index.php.

```
modules.php?name=AvantGo
```

If instead we wanted to call a page different from the default index.php (say, print.php), the string we will have to pass is :

```
modules.php?name=AvantGo & file=print
```

That is the file variable with a value (print) that corresponds to the name of the file we want to load without the .php extension.

Inside of the folder modules/nameofmodule there is also a subfolder called "language". In this fashion we manage in a simple and immediate way the multilanguage functionality inside the modules.

The rows modules.php work in this way:

- It includes the mainfile.php
- Verifies if the module is active
- Verifies whether the string passes a file name different from index.php
- Verifies the permissions of the module (whether everybody can see it, or only registered users, or only the administrator).

9.2. Creating fully compatible modules: the rules to follow

For people who have a base knowledge of the PHP language it is very simple to construct a module. Generally to create a PHP−Nuke module means:

- To create PHP files for the users, ie the public part of the site
- To create an administrator interface
- To verify that everything we have created is in keeping with the PHP−Nuke development rules.

But what about the development rules?

It is a good idea to stop on this point before continuing on to the programming part.

1. Rule: the modules must be included in the folder modules/namemodule in the public part and the folder admin/modules in the administration part
2. Rule: the main file of the module included in modules/nameofmodule must be called index.php
3. Rule: the tables in the php syntax are indicated by prefix. For example Nuke pages will be indicated with "$prefix."_pages, where $prefix takes the value from the config.php file which is nuke by default.
4. Rule: the location of the images or links must start from the root of your html directory and not from the folder modules/nameofmodule because the files contained in it are included in a file placed in html's root directory that's called modules.php
5. Rule: to manage the multilanguage function in an optimal way we have to create some text abstractions that we will insert in the files by making a folder called "language" inside the folder of the module. Everything will then be automatically recalled. For example, if we need to create a module that we call Topolino (the Italian name of Mickey Mouse) we must give the possibility to those who use the Italian interface to read "Topolino" and to those who use the English one to read "Mickey Mouse" ; −).

How do we do it?

First of all we create the folder "language" inside the folder modules/topolino We insert in this folder two php files that we will call lang−italian.php and lang−english.php We create an abstraction for topolino, in the lang−italian.php it will be:

```
define("_TOPO", "Topolino");
```

And in English it will be:

```
define("_TOPO", "Mickey Mouse");
```

In this way inserting in the module the abstraction "_TOPO" this will be automatically replaced by Topolino in the Italian interface and by Mickey Mouse in the English interface.

9.3. Module creation, the public part

We continue using the example of Topolino from Section 9.2 and create a very simple module that displays a GIF of Topolino with a list of 3 predefined names that are editable by the users. This is a nonsense module but it's simple enough to be understood by everybody. The DB we will use is MySQL, but the example, by changing some detail, works with all DB's. First of all let's see the skeleton of every module that we will construct:

```
<?php
if (!eregi("modules.php", $PHP_SELF)) {
die ("You can't access this file directly...");
}
$index = 1;
require_once("mainfile.php");
$module_name = basename(dirname(__FILE__));
get_lang($module_name);
include("header.php");
include("footer.php");
?>
```

Before making anything it's necessary to create a folder called "modules/Topolino", the file we have just created (with the other contents) must be called index.php and must stay in that folder.

We create a table in the database called nuke_topolino that is structured in this way:

- idperson: It is a cell that contains the id of person (int 11, primary)
- nameperson: It is a cell that contains the names of the people (varchar 60)

And we manually insert (by using PHPMyAdmin − see Section 11.3 − or an equivalent interface) the names of the 3 people we are interested in, see Figure 9−1 (the module, for simplicity reasons doesn't allow you to add or to cancel people but only editing those that already exist).

- Id 1: Topolino
- Id 2: Minnie
- Id 3: Pluto

Figure 9−1. PHPMyAdmin: inserting values

		idpersonaggi	nomepersonaggi
Edit	Delete	1	Topolino
Edit	Delete	2	Minnie
Edit	Delete	3	Pluto

PHPMyAdmin: inserting values

It's possible to include the footer at the end of every function. It's a solution that is a bit more complex because we must write more lines, but I have to stress how much alot of modules use it.

Once that the table of the DB is ready we can begin to enjoy creating the code that will give us back our output. Our output will be one simple query with a cycle that will give back the values inserted in the database (the simplest thing in the world, Gosh!).

Attention!!!

To maintain the abstraction of the DB so that it can work on various database's in an independent way, we cannot use classic PHP syntax that is generally used by the MySQL addicted ; −), instead we must use the syntax illustrated in the file include/sql_layer.php

The query that we will compile will be structured in the following way:

```
$resultpersons = sql_query("SELECT idperson, nameperson FROM "$prefix."_topolino", $dbi);
for ($m=0; $m < sql_num_rows($resultpersons, $dbi); $m++)
{
list($idperson, $nameperson) = sql_fetch_row($resultpersons, $dbi);
echo "$idperson - $nameperson < br >";
}
```

Very simple, isn't it? OK!, before passing to the administrator's interface for this module, we will start to modify it with the intention of giving it a minimum style dignity.

I would propose:

- to insert the results in a table
- to put a title to it and a description for the module.

We can do this by rendering all of the module's compatible with the multilanguage system of PHP−Nuke: We define the abstractions that will compose the two phrases we need, in the file lang−english.php we have to insert:

```
<?php
define("_BENVETOPOMOD", "Topolino Module, Welcome!");
define("_DESCRITOPOMOD", "This is an example module that serves to illustrate how a PHP-Nuke module is created");
>
```

Remember to insert a file called index.htm in our language folder! We need it to protect the inside of that folder from undesired navigations. We are nearly at the end of the frontend part, now we have to insert the stylistic part in the code that we have created and to assemble everything. We take two code pieces previously constructed (The initial one and the one created by us) and we join the stylistic modification:

```
<?php if (!eregi("modules.php", $PHP_SELF))
{ die ("You can't access this rows directly..."); }
$index = 1; require_once("mainfile.php");
$module_name = basename(dirname(__FILE__));
get_lang($module_name);
include("header.php");
echo "<br>";
echo""._BENVETOPOMOD."";
echo "<br><br>";
opentable();
echo "<br>";
echo""._DESCRITOPOMOD."";
echo "<br><br>";
$resultpersons = sql_query("SELECT idperson, nameperson FROM ".$prefix."_topolino", $dbi);
for ($m=0; $m < sql_num_rows($resultpersons, $dbi);
$m++) { list($idperson, $nameperson) = sql_fetch_row($resultpersons, $dbi); echo "$idperson - $nameperson <br>";
}
closetable();
include("footer.php");
?>
```

We have only added text, some breaks for the headers and an "Opentable();" and a "Closetable();" to include the text. The result can be seen in Figure 9−2

Figure 9−2. Example module

Example module

9.4. Module creation, administrator part

It's time to create the administration part of the module. In this very simple module the only function that will work on the DB will be the one in which we can modify the text of one of the three languages that we have created. First of all, we have to create the files to insert in the folders:

- admin/case
- admin/links
- admin/modules

It's important to remember how much we have just said in Section 6.1:

Admin:
: contains 4 subdirectories (links, language, case, modules) that manage the various administration modules. The folder that holds the files is modules/admin/ .

The folder admin/links instead says that the admin module must recall and position one language in admin for that determined module.

Example (Administration for the FAQ module):

```
if (($radminsuper==1) OR ($radminfaq==1)) {
adminmenu("admin.php?op=FaqAdmin ", "" _FAQ." ", "faq.gif");
}
```

This module:

- Verifies the administration rights (This module can be set up so it can be viewed by the superadmin or an admin)
- It passes a "CASE" (op=FaqAdmin) that indicates to the admin.php file (that includes all the admin modules) the module to call, associates a value in order to translate the term "FAQ" and associates an image for the visual administration (faq.gif).

The folder admin/case instead serves to define the module to be used in each specified case . This is important when, using the same admin file, we need to perform more operations using a CASE statement:

```
case1 = insert
case2 = cancel
```

```
etc...
```

In fact it says which module to call in order to verify the CASE condition. For example, in the FAQ module there are lots of cases, here are the last 2:

```
case "FaqAdmin":
case "FaqCatGo":
include ("admin/modules/adminfaq.php");
break;
```

Both of the CASE statements call the adminfaq.php file, but they are used for different operations. The first one calls the file with the default scheme, the second, on the other hand, gives its O:K. to insert a new category.

This happens through a string like "admin.php?op=FaqAdmin" in the first case and "admin.php?op=FaqCatGo" in the second.

We will now create, in the following order, the files:

- admin/modules/topolino.php
- admin/case/case.topolino.php
- admin/links/links.topolino.php

In order to create the file inside the modules folder (modules/topolino.php) we must have a structure of this type:

- Starting part of the file (Similar for all the modules)
- Definition of the necessary functions.
- Definition of the necessary cases in order to call the various functions for the admin module.
- Final part of the file.

The syntax for the starting part of the file is the following:

```
<?php
if (!eregi("admin.php ", { die ("Access Denied"); }
$result = sql_query("select radminsuper, admlanguage from "$prefix."_authors to where aid=' $aid '", $dbi);
list($radminsuper, $admlanguage) = sql_fetch_row($result, $dbi);
if ($radminsuper==1) {
```

This part of the file controls the administration rights for whoever calls it, a control on which language to use and (not in this module) a control on the administrator's rights. An administrator could have only partial rights of administration on modules or some modules can be managed only by a superadmin. In our specific case the module can be managed only by the superadmin because the control is only:

```
if ($radminsuper==1)
```

In case there are some specific rights (for example in the reviews module) the rights to control would have been:

```
if (($radminreviews==1) OR ($radminsuper==1))
```

Activating the rights on two levels on new modules indeed isn't simple, you must specify in the nuke_authors table a new field that designates the rights, then modify admin/modules/authors.php adding the checkbox for the rights of the new module and modify the corresponding UPDATE queries.

Let's go back to our module, the initial part of the syntax is obligatorily

```
<?php
if (!eregi("admin.php ", { die ("Access Denied"); }
$result = sql_query("select radminsuper from "$prefix."_authors to where aid=' $aid '", $dbi);
list($radminsuper) = sql_fetch_row($result, $dbi);
if ($radminsuper==1) {
and that end is instead:
} else {
echo "Access Denied";
}
>
```

All that we find in the middle, are the management functions and the cases that must be checked, which we will now go on to construct. There are a couple of rules to follow in order to construct the admin functions:

- We must include a header at the beginning of the function and a footer at the end:

```
include("header.php");
include("footer.php");
```

We have to include the GraphicAdmin(); function immediately after the header. It will display the navigation panel that leads to all the other admin links. The functions we'll now go to create are:

- Display of the database records
- Selection of the record and its insertion in a modifiable text field
- Modification of the record through insertion in the database of the value modified in the text field.

Here is the code of the function which accomplishes the record selection:

```
function mousedisplay() {
global $admin, $bgcolor2, $prefix, $dbi, $multilingual;
include ("header.php");
GraphicAdmin();
Opentable();
$resultpersons = sql_query("SELECT idperson, nameperson FROM ".$prefix."_topolino", $dbi);
for ($m=0; $m < sql_num_rows($resultpersons, $dbi); $m++)
{
list($idperson, $nameperson) = sql_fetch_row($resultpersons, $dbi);
echo "$idperson - $nameperson < to href=\"admin.php=mouseselect & idtopo=$idperson \" > Select mouse </to > < br > ";
}
closetable();
include("footer.php");
}
```

The next function to implement is the one that corresponds to the selection of one of the three records:

```
function mouseselect() {
global $admin, $bgcolor2, $prefix, $dbi, $multilingual, $idtopo;
include ("header.php");
GraphicAdmin();
Opentable();
$resultpersons = sql_query("SELECT idperson, nameperson FROM "$prefix."_topolino to where idperson=' $idtopo '", $dbi);
```

```
for ($m=0; $m < sql_num_rows($resultpersons, $dbi); $m++)
{
list($idperson, $nameperson) = sql_fetch_row($resultpersons, $dbi);
echo "< form action=\"admin.php\" method=\"post\">";
echo "< input type=\"text\" name=\"nameperson \ "size=\"20\" maxlength=\"20 \ "value=\"$personname \" >< br >< br > ";
echo "< input type=\"hidden\" name=\"idperson\"value=\"$idtopo\" > ";
echo "< input type=\"hidden\" name=\"op \ "value=\"mousemodify\" > ";
echo "< input type=\"submit\" value=\""._ADDTOPO."\" > ";
echo "</form >";
}
closetable();
include("footer.php");
}
```

It is very important to take note of some things:

1. The variables that we insert will be checked, in fact you can see the variable "$idtopo" was inserted between the used variables.
2. The value checked in the CASE statements is passed to us through a hidden form field

```
(< input type=\"hidden\"name=\"op\"value=\"mousemodify\">).
```

The last function we consider is the one that corresponds to the update of the values in the database (Here too we added the two variables that we were interested in ($nameperson, $idperson):

```
function mousemodify() {
global $admin, $bgcolor2, $prefix, $dbi, $multilingual, $idtopo, $nameperson, $idperson;
include ("header.php");
GraphicAdmin();
Opentable();
sql_query("update "$prefix."_topolino set nameperson=' $nameperson' where idperson=$idperson", $dbi);
echo"OK ";
die(mysql_error());
closetable();
include("footer.php");
}
```

The last two elements to insert are the definitions for the CASE statements (that is, which function to call according to the variable passed to the module) and the closing of the file.

Definition of the cases:

```
switch($op) {
case "":
mousedisplay();
break;
case "topolino":
mousedisplay();
break;
case "mouseselect":
mouseselect();
break;
case "mousemodify":
mousemodification();
break;
}
Closing of the file:
} else {
echo "Access Denied";
```

```
}
>
```

The cases definition page is very easy to construct, it gathers the cases that are included in the file admin/modules/topolino.php and puts them in the file admin/case/case.topolino.php

This is the syntax:

```
<?php
if (!eregi("admin.php ", { die ("Access Denied"); }
switch($op) {
case "topolino":
include("admin/modules/topolino.php");
break;
case "mouseselect":
include("admin/modules/topolino.php");
break;
case "mousemodify":
include("admin/modules/topolino.php");
break;
}
>
```

The last two things we have to make are the compilation of the file admin/links/link.topolino.php and the creation of a language module: Compilation of the file link.topolino.php

```
<?php
if ($radminsuper==1) {
adminmenu("admin.php?op=topolino ", "" _EDITTOPOLINO." ", "topolino.gif");
}
>
```

Where: admin.php?op=topolino defines which module must be called, " _EDITTOPOLINO" is the term to translate (it must be compiled in admin/language). For the modification of the language module I refer you to the previous paragraph with one single note. The language file of the admin section is common for all (admin/language), the relative languages must be added to the end of those that already exist. Just another thing, the syntax of this example is not perfect, its beyond the scope of this to make it work perfectly but it does illustrate the operation of the module (Which you will find available for download at www.spaghettibrain.com so you will be able to study it).

Chapter 10. Some security precautions

10.1. The permissions on the folders and files

This section is of importance only to those who use PHP−Nuke under Linux/Unix (this is true for the greater part of PHP−Nuke sites that are hosted by providers, and often also those who test locally use Linux).

Burzi says that the directories should be assigned a mode of 777, the files a mode of 666, but we may calmly let our PHP−Nuke do its work under more restrictive permissions, as illustrated below:

- config.php (666)
- backend.php (666)
- ultramode.txt (666)
- All directories (755)
- Other files (644)

The files config.php, backend.php, ultramode.txt must have the write permissions because :

- For config.php editing the preferences we will write this file modifying the text.
- For the backend and ultramode on the other side, we will write them (in an automatic way) modifying the titles and abstracts of the news.

There is however something particular we have to take into account: if we use modules that upload files in some directories, their permissions wil have to be raised. As an example, consider the IndyNews module, a non standard module that makes it possible to enclose files and images in articles. The structure of the module is the following:

- modules/indynews/media

In the inside of the "indynews" folder the permissions of the folder "media" would have to be 777, due to an override problem, the 777 permissions will have to be imposed on everything that is below "modules". For this reason, everything that resides in "modules" will be in 777 mode and this could cause a vulnerability. A solution is to move the folder that will have to accommodate the uploaded files to the outside of the modules folder, even to the document root, changing inside the module all the references to it.

Doing so will leave one single folder in root with permissions set to 777.

10.2. Cookies − timeout and configuration

PHP−Nuke makes heavy use of cookies, be it for user authentication, or admin authentication. The cookies are text files that are saved on our computers and contain various information that is read when we enter a certain site. In the case of PHP−Nuke the information saved there pertains to the user, the chosen theme and the language used.

The cookie is also the instrument that enables us not to have to retype the password each time we log in. This way, each time we access a PHP−Nuke site, the cookie works for us by managing the login operation.

The problem is that if the cookie does not have an expiry date low enough, someone can to steal it from us and be able to access the site as a user or administrator. This is possible for a series of reasons:

1. the cookie of PHP−Nuke has a life duration close to infinite (31536000 seconds)
2. Explorer (most used browser, unfortunately) has vulnerabilities that allow the execution of malicious scripts on the client that "steal" the cookie from the user and send it to the "burglar".
3. PHP−Nuke does not succeed in filtering all the malicious scripts (or, to put it better, Internet Explorer is so stupid that corrects inserted scripts with the wrong syntax in order not to be recognized).

Let's show a concrete example of how a script kiddie (those who hold themselves for hackers, but they are not...) can try to obtain administrator rights on our site:

1. The script kiddie inserts a script that supposedly contains news:
   ```
   < vb script give the cookie to me and send it to the server xyz>
   ```
   that is not filtered by the function check_words() of PHP−Nuke.
2. The administrator of PHP−Nuke opens the page up with Internet Explorer!!! (This hack does not work if you're using Mozilla, or better yet any Linux browser). The list of the news waiting to be approved for publishing is seen by the administrator. When he goes to look at the Submissions, Internet Explorer (stupidly) corrects the vbscript in this way:
   ```
   <vbscript>(script kiddies commands go here)
   ```
   succeeding to interpret the wrong syntax in the correct way (!!!), taking the cookie and sending it to the script kiddie.
3. The script kiddie puts the cookie among the other ones of his own, connects to the site and... is recognized as being the administartor!!!

But how is it possible to protect ourselves from this type of hack?

There are some solutions that should increase much of the security for our administration area:

1. First of all STOP using Internet Explorer as a browser and pass the seat to Mozilla. Mozilla is a browser that supports all sites in an optimal way and is not plagued by all the vulnerabilities of Microsoft. If you use Linux instead you won't encounter any problems of this sort...

   a. In case you want to continue to use the Explorer, you should at least download the patches from Microsoft.
2. Disable, where possible, the insertion of HTML tags (for example in the forum of Splatt.it)
3. Narrow down the life of cookies. If for example we set up the life of the cookie to two hours, the script kiddie will be forced to use the cookie within that period, this limits much of their ability to act in time.

   If instead we leave the life of the cookie to its preset value, the script kiddie may use our cookie even for 1 month after it was stolen.

   How to set up the duration of the administartor cookie? The cookie is set up in the file includes/auth.php and the function to modify it is the following:

   ```
   if ((isset($aid)) && (isset($pwd)) && ($op == "login")) {
       if($aid! = "" AND $pwd!="") {
           $pwd = md5($pwd);
           $result=sql_query("select pwd, admlanguage from "$prefix."_authors  where aid='$aid'", $dbi);
           list($pass, $admlanguage)=sql_fetch_row($result, $dbi);
           if($pass == $pwd) {
               $admin = base64_encode("$aid:$pwd:$admlanguage");
               setcookie("admin", "$admin",time()+7200);
               unset($op);
   ```

```
        }
    }
}
```

As you see we have modified the life duration of the second cookie from 2592000 (a month) to 7200 seconds (two hours). As you can easily see, we have reduced the action radius of the script kiddie down from one month to two hours.

4. A much more effective tag filter is in the study phase, although for the moment, the proposed solutions did give a definitive answer to the problem. The admissible tags are defined in in the file config.php in the variable $AllowableHTML, these are valid for the comments and the insertion of news in the function check_html() which for the moment lets some tags pass through.

All these actions and a correct configuration of the permissions as illustrated in Section 10.1 should guarantee us a good security for our site. It is also important to closely follow the security warnings for PHP−Nuke that are brought up on http://neworder.box.sk/.

Chapter 11. Programmer's tools

11.1. The database tables

Here are the tables that comprise the PHP–Nuke database and what they serve for:

nuke_access:
: Defines various users profiles.

nuke_authors:
: Defines the administrators and their access levels.

nuke_autonews:
: Manages automatic news.

nuke_banlist:
: Manages banned users (those excluded from viewing the site).

nuke_banner:
: Manages banner campaigns, impressions and clicks.

nuke_bannerclient:
: Manages the banner campaign clients.

nuke_bbtopics:
: Manages the BBCode.

nuke_blocks:
: List of created blocks.

nuke_catagories:
: List of the categories.

nuke_comments:
: Manages comments and answers.

nuke_config:
: Manages some configurations like the possibility to mail in HTML etc.

nuke_counter:
: Manages statistics.

nuke_disallow:
: Blocks a user.

nuke_downloads_categories:
: Manages categories and subcategories for the download area.

nuke_downloads_downloads:
: Manages the files present in the download area.

nuke_downloads_editorials:
: Manages the comments on the files.

nuke_downloads_modrequest:
: Manages the reporting of broken links.

nuke_downloads_newdownload:
: Manages the insertion of files from third parties.

nuke_downloads_moddata:
: Manages voting on files.

nuke_encyclopedya:
: Lists the various encyclopedias.

nuke_encyclopedia_text:
: Lists encyclopedia entries.

nuke_ephem

Manages recurrent events.

nuke_faqanswer:
Archives the FAQ answers.

nuke_faqcategories:
Manages the categories in which the FAQ are subdivided.

nuke_forum_access:
Records the last access of the users.

nuke_forum_mods:
Defines moderators of the forums.

nuke_forums:
Defines the active forums.

nuke_forumtopics:
Defines forum topics.

nuke_headlines:
Defines the sources from which to take news with the blocks.

nuke_links_categories:
Defines the categories in which the links are subdivided.

nuke_links_editorials:
Manages the judgments on the links.

nuke_links_links:
Archives the links.

nuke_links_modrequest:
Archives the messages related to the link (broken link etc.).

nuke_links_newlink:
Archives links suggested for insertion.

nuke_links_votedata:
Archives link votes.

nuke_main:
Defines which is the main module that must be included in index.php.

nuke_message:
Manages the home page messages.

nuke_modules:
Lists and manages the installed modules.

nuke_pages:

nuke_pages_categories:

nuke_poll_check:
List of the IP addresses that have voted in last 24 hours.

nuke_poll_data:
List of survey answers.

nuke_poll_desc:
List of present and past surveys.

nuke_pollcomments:
Comment to the survey.

nuke_posts:
Main titles of forum posts

nuke_posts_text:

nuke_priv_msgs:
Manages the module private messages.

nuke_queue:
: Lists the texts waiting for publication

nuke_quotes:

nuke_ranks:
: Registers the votes for the news

nuke_referer:
: Lists where the last x rows come from

nuke_related:
: Associates eventual links to the topics.

nuke_reviews:

nuke_reviews_add:

nuke_reviews_comments:
: Comments to the reviews.

nuke_reviews_main:
: main table of the reviews.

nuke_seccont:
: Lists the section texts.

nuke_sections:
: Lists the active sections.

nuke_session:
: Lists the active sessions.

nuke_smiles:
: Lists the supported emoticons.

nuke_stats_date:
: Statistics module.

nuke_stats_hour:
: Statistics module...

nuke_stats_month:
: Statistics module.

nuke_stats_year:
: Statistics module.

nuke_stories:
: Texts of the news.

nuke_stories_cat:
: Categories of the news.

nuke_topics:
: List of topics.

nuke_users:
: List of users.

nuke_words:
: Words to censure.

11.2. The syntax of SQL code

Aiming at making PHP−Nuke compatible with more databases, the SQL syntax has been transformed to functions, in order to achieve a standard syntax that is independent of the database used. For convenience, let us recall the file sql_layer.php in a somewhat cleaned−up version:

```
sql_connect($host, $user, $password, $db) ❶
sql_logout($id) ❷
sql_query($query, $id) ❸
sql_num_rows($res) ❹
sql_fetch_row(&$res, $nr) ❺
sql_fetch_array(&$res, $nr) ❻
sql_fetch_object(&$res, $nr) ❼
sql_free_result($res) ❽
```

❶ Log into the DB.

❷ Disconnect from the DB.

❸ Query.

❹ Number of Rows.

❺ Fetch Rows.

❻ Fetch Array.

❼ Fetch Object.

❽ Free Result.

With this syntax you will be able to render all the modifications, blocks or modules you create compatible to all the databases supported by PHP−Nuke, which are:

- MySQL
- mSQL
- PostgreSQL
- PostgreSQL_local
- ODBC
- ODBC_Adabas
- Interbase
- Sybase

11.3. PHPMyadmin, administering MySQL via web

11.3.1. What is PHPMyadmin

PHPMyAdmin is n visual system for the management of a MySQL database. It is written in PHP and serves to display the contents of the databases on the server (or client) on which MySQL is installed. Through this interface you can create new databases, modify existing ones and modify the contents of single fields (see Figure 11−1).

Figure 11−1. PHPMyAdmin start screen

PHPMyAdmin start screen

11.3.2. How to install the PHP−Nuke DB with PHPMyadmin

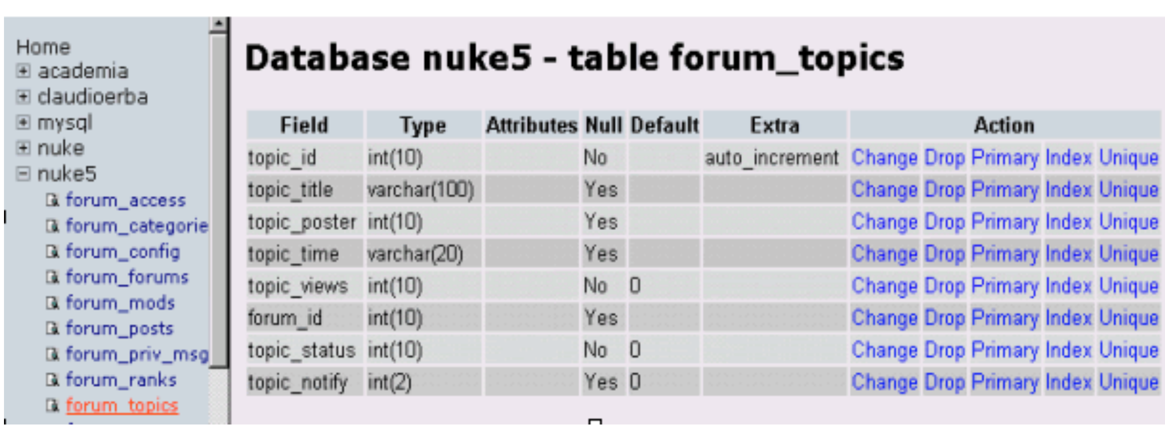
Please read first Section 5.2.2 for the basic installation procedure of the PHP−Nuke MySQL database through PHPMyAdmin. In this section we will discuss some more advanced PHPMyAdmin topics:

PHPMyAdmin gives you also the possibility to do a backup of the data in your database (or only of the structure). If you choose "Structure Only and the "Send" option, you will save on your PC the database structure without the data. Choosing "Structure and Data" instead, you will be sent a real whole backup of your DB.

11.3.2.1. Other options of PHPMyAdmin

Yet another couple of instructions: In order to see the structure of a table, you only need to click on the one that is marked in the left bar and you will see all its fields and modification options appear in the central part (Figure 11−2).

Figure 11−2. PHPMyAdmin: table forum_topics



PHPMyAdmin: table forum_topics

ATTENTION!

The DROP command eliminates all the contents of the DB, the table or the single field, use it with caution.

Generally, the management interface of PHPMyAdmin is supplied by the provider who sells you the hosting or, if you install it on Windows in order to work locally, does not need any particular tweaking in its configuration.

In case you want to install it on Windows in order to read databases that are online or to install it on your webspace because your hosting supplier does not include it in your tools, you can configure everything by editing the config.inc.php file as follows:

Supposing that:

- IP DB Server: 156.123.22.34
- User: Pippo
- Password: Topolino
- Database Name: Minnie

Then:

```
$cfgServers[1]["host"] = "156.123.22.34"; // MySQL hostname
$cfgServers[1]["port"] = ""; // MySQL port - leave blank for default port
$cfgServers[1]["adv_auth"] = false; // Use advanced authentication?
$cfgServers[1]["stduser"] = ""; // MySQL standard user (only needed with advanced auth)
$cfgServers[1]["stdpass"] = ""; // MySQL standard password (only needed with advanced auth)
$cfgServers[1]["user"] = "Pippo"; // MySQL user (only needed with basic auth)
$cfgServers[1]["password"] = "Topolino"; // MySQL password (only needed with basic auth)
$cfgServers[1]["only_db"] = "Minnie"; // If set to a db-name, only this db is accessible
$cfgServers[1]["verbose"] = ""; // Verbose name for this host - leave blank to show the hostname
$cfgServers[1]["bookmarkdb"] = ""; // Bookmark db - leave blank for no bookmark support
$cfgServers[1]["bookmarktable"] = ""; // Bookmark table - leave blank for no bookmark support
```

In the config.inc.php you will find more configuration parameters that are repeated. They serve to manage DBs in various hosts with the same interface.

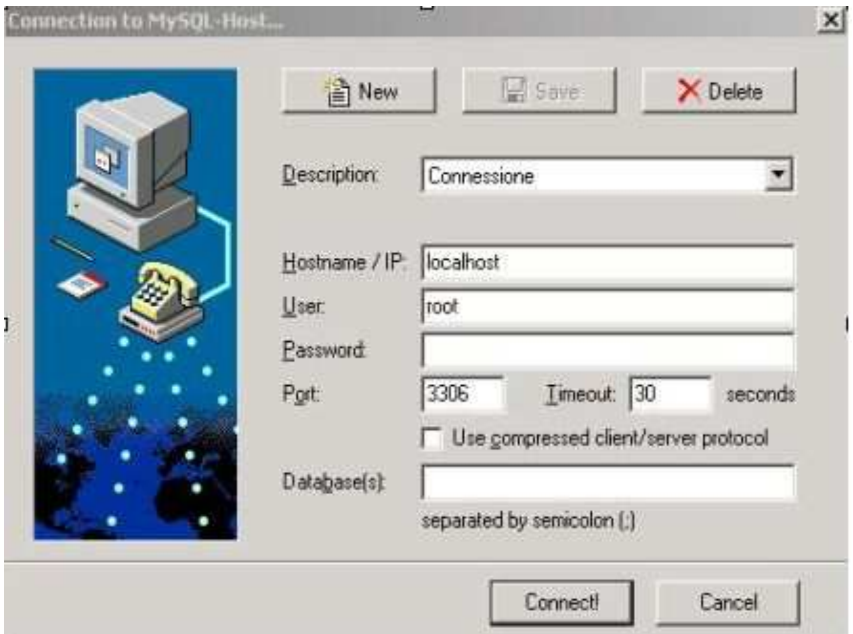
11.4. MySQL Front, how to administer a MySQL DB from Windows

Mysql Front is a client for Windows that allows for the management to manage a local or remote DB through a client interface. Since we consider PHP−Nuke to be more aproppriate and flexible tool for this task, we would like to focus our attention a little on a functionality of MySQLFront that on PHPMyAdmin has shown itself to be little reliable. That is the ability to load, import and export DBs of great dimensions without losing data or getting errors.

In this section we will limit ourselves in analyzing the connection phase, the import and upload.

Connection (Figure 11−3): Clicking on File−>Connection, we will get an interface in which we are asked to enter the host, user and password. If we have to use MySQLFront locally, we will leave the localhost entry untouched.

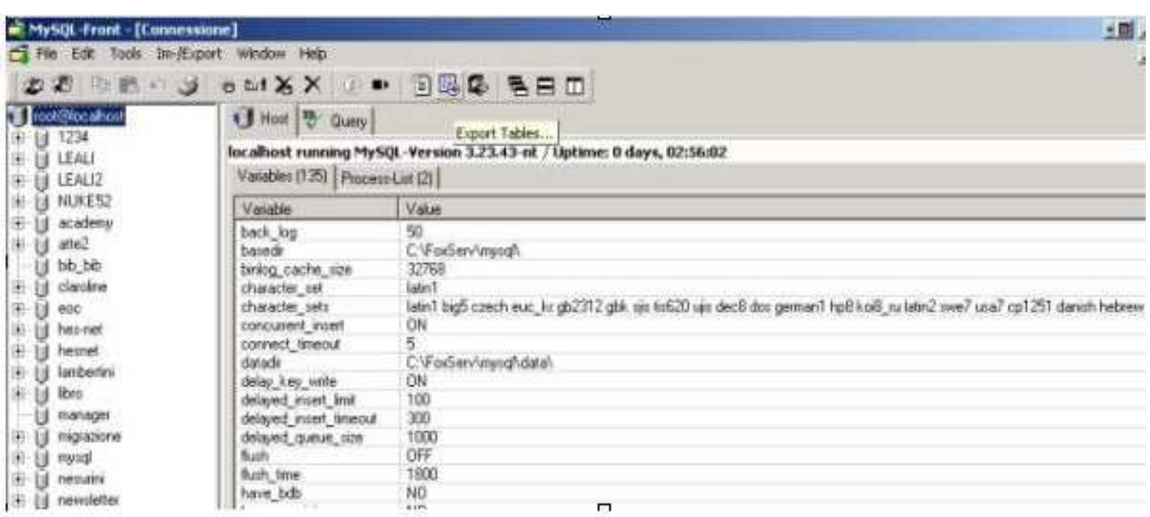
Figure 11−3. MySQL Front: Connection



MySQL Front: Connection

Download (or copy) of an existing DB: choose the database you wish to be exported and then click on "Export Tables" (fifth from the right, Figure 11−4). Then choose a destination path for the database dump. As you may notice, the system, through the use of flags, gives us even the possibility to choose individually those tables we want to import.

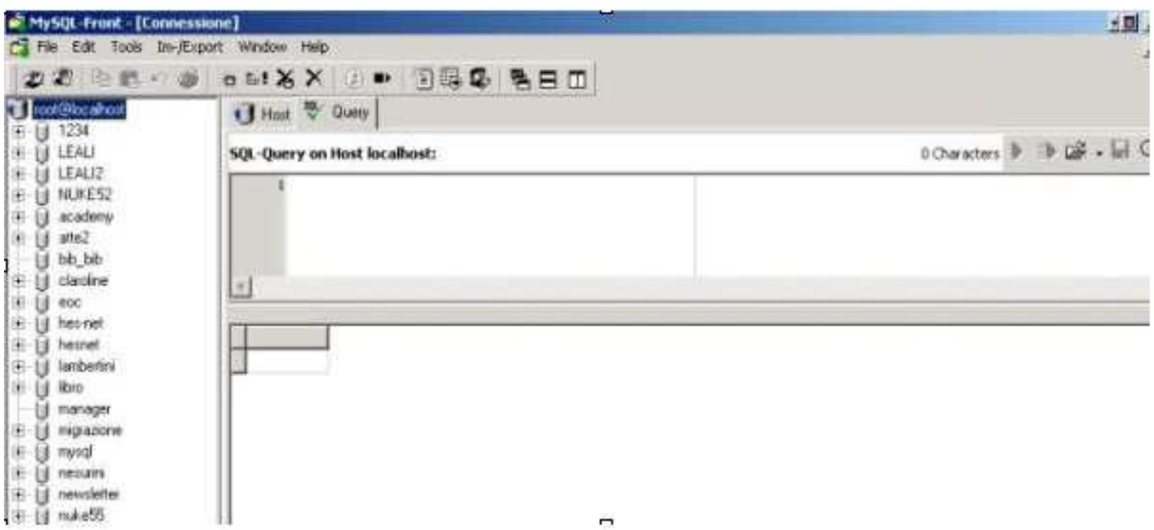
Figure 11−4. MySQL Front: export tables



MySQL Front: export tables

Upload of a DB: clicking on the SQL Query button (Figure 11−5), an interface appears that allows us to carry out the upload of an existing database. It suffices to click on the symbol of the open folder, choose the database dump and click on the button that looks"Play"−like button. Before starting this operation will have to select the destination database.

Figure 11−5. MySQL Front: SQL Query



MySQL Front: SQL Query

11.5. Foxserv, making PHP−Nuke work on Windows Systems

There exists an application that installs PHP + MySQL + Apache in a simple and fast way, configuring your operating system to emulate a local Web server. This is very useful, as it allows us to experiment with PHP−Nuke and our modifications locally.

The software is called Foxserv, has reached version 3.0, but I recommend using version 2.0 which is much easier to use.

The installation is very simple, it is enough to launch the FoxServ−2.0core.exe file, proceed with the installation inserting any data (nothing is needed besides user and password of the DB) and then start using it!

When launching the foxserv control panel, you will have to activate apache and mysql, type in your browser the address http://localhost/and the web page will appear as if it were on any Internet server.

Suppose that you have installed everything in c:\foxserv. The PHP folders have to be inserted under the folder www. For example if we want to insert a nuke56 installation, we create the folder c:\foxserv\www\nuke56\ and load there all the content of the HTML folder of our PHP−Nuke (in the config.php of PHP−Nuke we will leave "localhost" as our host, "root" as user and nothing as password).

In order to reach PHPMyAdmin (even this is preinstalled), the address will be http://localhost/phpmyadmin/.

Important note

The file php.ini (which is located, depending on the operating system, in c:\windows, c:\windows\system, c:\winnt or c:\winnt\system\) contains paths to a disc f:

The only modification that you will have to make consists in:

- create a folder c:\temp
- modify the parameter: session.save_path = c:\temp. It must be c:\temp and not f:\foxserv etc.

Attention

The folder "php" contains also a php.ini, but it is not the file that interests us.