iPHPortal
Руководство разработчика

6 Шаблонные директивы и методы класса “Шаблон”

1. Описание системы iPHPortal

1.1. Системные требования

Основные требования

- Веб-сервер Apache 1.3 и старше
- PHP **4.0.5** и старше, установленный как модуль Apache (+ zip extension для установки модулей)
- MySQL 3.23 и старше
- **!!!!Права доступа на запись (создание / чтение /изменение / удаление) файлов на сайте из скриптов PHP!!!!!**
- Возможность использование .htaccess файлов (в httpd.conf Apache должна быть установлена опция AllowOverride All)

Опционально

- Для работы с изображениями - модуль PHP GD или ImageMagick
- Для адресации статических файлов к материалам – модуль Apache mod_rewrite

1.2 Принципы работы системы

Сайты, построенные на базе системы состоят из 3 взаимосвязанных частей:

1. База данных
2. Административный интерфейс (бэкофис)
3. Внешнее представление сайта (фронтофис)

В базе данных хранится структура и материалы сайта, информация интерактивов и служебная информация необходимая для работы системы. Изображения (сопровождающие картинки материала и т.д.) хранятся в файлах.

Бэкофис – это основной инструмент администраторов и операторов системы. Бэкофис, на сайтах с установленной системой администрирования, как правило, остается неизменным по своей структуре и внешнему виду. Изменения в основном сводятся к скрытию неиспользуемых функциональностей.

Функции бэкофиса:

1. Изменение структуры и информационного наполнения сайта
2. Работа с шаблонами и обработчиками
3. Модерирование интерактивов всех типов
4. Управление пользователями и правами пользователей

Фронтофис:

Внешний вид и функциональность фронтофиса определяется структурой, шаблонами и обработчиками, на каждом сайте они отличаются.

В системе сайт рассматривается как дерево рубрик (директорий), в рубриках размещаются материалы (файлы) или различные интерактивы. Все рубрики (директории) и материалы (файлы) предгенерируются на сайт. Первая страница сайта – это индекс корневой рубрики (директория /).

Название	Директория	Материалы
Site.Ru	/	Нет
Новости	/news/	Да
Официальные новости	/news/official/	Да (1)
Анонсы конференций и семинаров	/news/conferences/	Да
Новые технологии	/news/tech/	Да
Новости портала	/news/portal/	Да (1)
Экспорт новостей	/news/rss-export/	Нет
Персональные настройки	/user/	Нет
Гостевые книги	/gb/	Нет
Библиотека	/lib/	Нет
Тесты	/tests/	Нет
Документация	/docs/	Нет (1)
Каталог ресурсов	/catalog/	Нет
Организации	/orgs/	Нет
Персоналии	/persons/	Нет
Проекты	/projects/	Нет
Конференции	/conferences/	Нет
Форумы	/forums/	Нет
Интернет-гостиные	/quests/	Нет

Рис.1.1. Дерево рубрик сайта.

Рубрики сайта хранятся в таблице rubrics. Дерево рубрик представлено в виде вложенных множеств, принцип описан тут http://sdm.viptop.ru/articles/sqltrees.html.

Рубрики и материалы связаны как многие-ко-многим, так что материал может принадлежать нескольким рубрикам. Страницы материала хранятся в отдельной таблице material_page.

Страницы (файлы) сайта генерируются с помощью пары “обработчик” (handler) и “шаблон” (template). Обработчик – это функция языка php , в которой извлекаются нужные данные из БД, создается и наполняется объект “Шаблон”. Шаблоны создаются на языке директив доработанного шаблонного “движка” TemplatePower. Тексты обработчиков и шаблонов сохраняются в базе данных и предгенерируются в файлы (для ускорения работы шаблонного “движка”).

Из пары обработчик + шаблон(ы) создаются “страницы” (в странице может быть больше одного шаблона, например при генерации статьи на сайте может использоваться один обработчик и два шаблона – шаблон материала и шаблон версии для печати), действия и “типы рубрик”.

Для “страницы” определяются опции редактирования рубрики/материала в бэкофисе – опции могут быть назначены отдельно для страницы и для всего сайта.

Созданные “страницы” связываются с рубриками – страница для индекса рубрики и страница для материалов рубрики. См. Раздел “Рубрики” руководства пользователя.

Созданный шаблонным движком страница индекса рубрики или материал сохраняется в файл (предгенерируются), либо сразу выводится на экран. Если установлена опция $site[‘regenerate_delay’] (по умолчанию установлена, опция появилась в версии 1.32 системы, до этого файлы перегенерировались полностью), то при перегенерации материала или индекса рубрики создается файл вида:

```
<?
 error_reporting (0);
 require_once ('common.inc.php');
 ins_class ('rubrics');
 $rubric_data = '';
 $rubrics->generate_index(1, $rubric_data);
 include ($_SERVER["SCRIPT_FILENAME"]);
?>
```

В этом файле (приведен пример для индекса рубрики) вызывается функция перегенерации индекса рубрики (т.е. того же самого файла) и затем новый файл показывается пользователю. Это позволяет снизить нагрузку на сервер при перегенерации большого количества материалов и индексов рубрик, т.е. нагрузка “растягивается” во времени.

В сохраненном в файл коде используются директивы php и поэтому расширение предгенерируемых файлов должно быть настроено в конфигурационном файле Apache как обрабатываемое PHP (по умолчанию используется расширение .php)

Генерация индекса рубрики происходит так:

1. Считывается информация о рубрике
2. Если у рубрики есть индекс (поле rubric_index != 0) продолжаем
3. Если тип индекса рубрики – “редирект” генерируем индексный файл рубрики вида <? Header(“Location:$rubric_index_redirect_url”) ?>
4. Иначе получаем информацию об обработчике и шаблонах индекса рубрики
5. Включаем файл с обработчиком и вызываем функцию обработчик

См. функцию generate_index в классе rubrics_basic.

Аналогично для материалов рубрики функция generate_material в классе materials_basic

При занесении материала пользователем генерируется файл материала и перегенерируется индекс рубрики, в которую входит материал, и индексы всех вышележащих рубрик. Индекс рубрики и всех родительских рубрик перегенерируется также при редактировании параметров рубрики.

Каждая рубрика и материал имеют “хозяина” (хозяин - это пользователь, который занес рубрику или материал).

Пользовательская система состоит из пользователей, групп пользователей и прав пользователей. Права пользователей определяют то, что пользователь может видеть и редактировать в бэкофисе, права назначаются для групп пользователей. Пользователь может входить в любое количество групп, его права – это объединение прав групп, в которые он входит. См. раздел “Пользователи” руководства пользователя.

2. Установка системы

Внимание!

Разработка сайта обычно ведется на локальном компьютере. После разработки, когда сайт полностью готов и правильно работает, он переносится на сервер хостинг-провайдера. Т.е. установка системы производится два раза – сначала на локальной машине, затем готовый сайт устанавливается на удаленном сервере. Естественно ничто не мешает сразу устанавливать систему на рабочий сервер и вести разработку сайта на нем.

2.2 Структура директорий сайта

/
/docs/ - www директория сайта (варианты htdocs, www и т.п.)
/include/

… другие директории

Директория include может быть вложена в www директорию сайта.

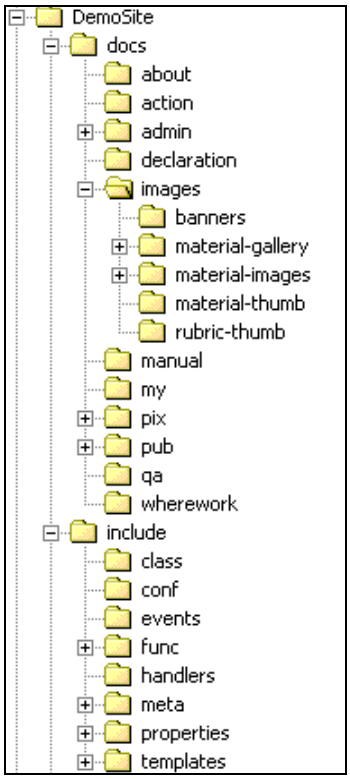

Рис. 2.1. Структура директорий сайта с установленной системой. В директории docs/admin *и* include *находятся файлы необходимые для работы системы. В поддиректориях директории* docs/mages/ *размещаются различные типы изображений. Остальные директории* docs/*** *созданы при создании структуры сайта (п. 3.2)*

Назначение директорий:

Директория	**Описание**
/docs/admin/	Скрипты административного интерфейса.
/docs/admin/doc	Документация
/docs/images/banners/	Баннеры (картинки, flash).
/docs/images/material-gallery/	Картинки фотогалереи материала, для каждой галереи создается поддиректория, имя которой равно material_id материала.
/docs/images/material-images/	Картинки материалов. Картинки загружаются через ”визуальный” или “простой” редактор текста.
/docs/images/material-thumb/	Сопровождающие картинки материалов.
/docs/images/rubric-thumb/	Сопровождающие картинки рубрик
/include/	Содержит основной файл конфигурации сайта common.inc.php.
/include/cache/	Файлы с кэшированными данными (права пользователя и т.д.)
/include/class/	Файлы классов объектов системы управления сайтом.
/include/conf/	Файлы конфигурации.
/include/events/	Файлы событий и действий (п. 3.7).
/include/func/	Функции
/include/handlers/	Предгенерированные файлы обработчиков.

/include/meta/	Предгенерированные файлы с META-данными материалов и рубрик
/include/templates/	Предгенерированные файлы с шаблонами.

Настройки имен директорий находятся в файле /include/conf/filepaths.inc.php.

Директорию images и поддиректорию images/.. необязательно создавать вручную, ее создаст скрипт при первой загрузке картинки.

2.2. Настройка Apache

Для того чтобы отрабатывались инструкции в файлах .htaccess в конфигурационный файл Apache httpd.conf нужно добавить

```
<Directory />
   AllowOverride All
</Directory>
```

Вместо "/" может быть указание директории, где находится сайт.

Если используется Russian Apache, то в конфигурационный файл нужно добавить строку:

CharsetRecodeMultipartForms Off

Иначе загружаемые через браузер файлы (картинки и т.п.) будут "битыми", их формат будет нарушен.

!!! Необходимо чтобы у скриптов php было право на запись во всех директориях на сайте. Для этого Apache должен работать от пользователя – владельца файлов, или права доступа на все директории и файлы на сайте 666. !!!

Для загрузки модулей системы должно быть подключено расширение php_zip. Без этого расширения модули, поставляемые в виде заархивированного xml-файла, нужно предварительно разархивировать.

2.3 Установка и конфигурация системы

Для установки системы нужно:

1. Разархивировать дистрибутив.
2. Настроить DocumentRoot веб-сервера на директорию www дистрибутива.

Например, вы распаковали дистрибутив в директорию /usr/local/www/distribsite/.

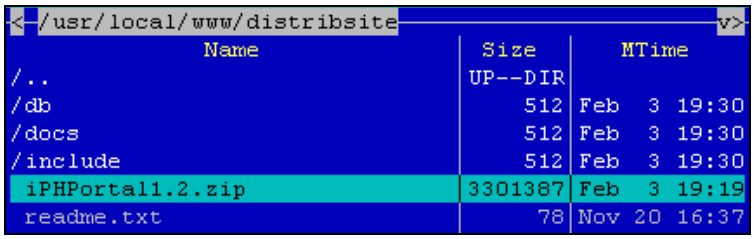

Рис. 2.2. Распакованный дистрибутив

Тогда настройка DocumentRoot Apache (устанавливается в файле httpd.conf) должна указывать на директорию /usr/local/www/distribsite/docs. Если на вашем хостинге невозможно изменить настройку DocumentRoot – то имя директорию docs можно переименовать, в дальнейшем в конфигурационный файл нужно будет внести изменения.

3. Запустить скрипт настройки admin/install/. Если система установлена на локальный сервер, то адрес инсталляционного скрипта http://localhost/admin/install/. Вместо запуска скрипта настройки можно отредактировать конфигурационные файлы системы (п.5).

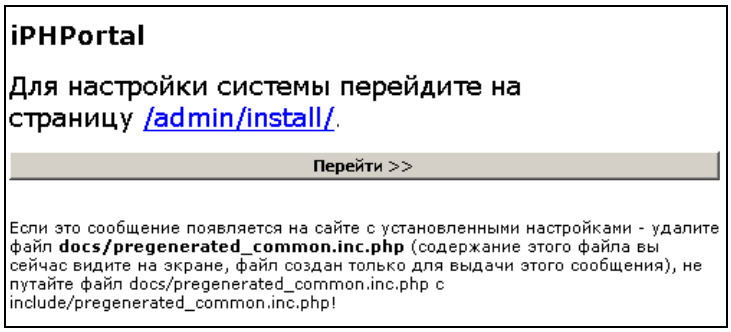

Рис. 2.3. Сообщение при заходе на сайт с установленной, но не настроенной системой

Возможные ошибки:

- Если при запуске скрипта выдает ошибку HTTP 500 – то в конфигурационном файле Apache не указана опция AllowOverride All.
- Если вместо запуска скрипта установки отображается его исходный код – то PHP не настроен на файлы с расширением .php
- Если вместо запуска скрипта установки показывается список файлов директории admin/install/ или выдается ошибка HTTP 403 - то в списке индексных файлов Apache не указан index.php

4. Провести настройку

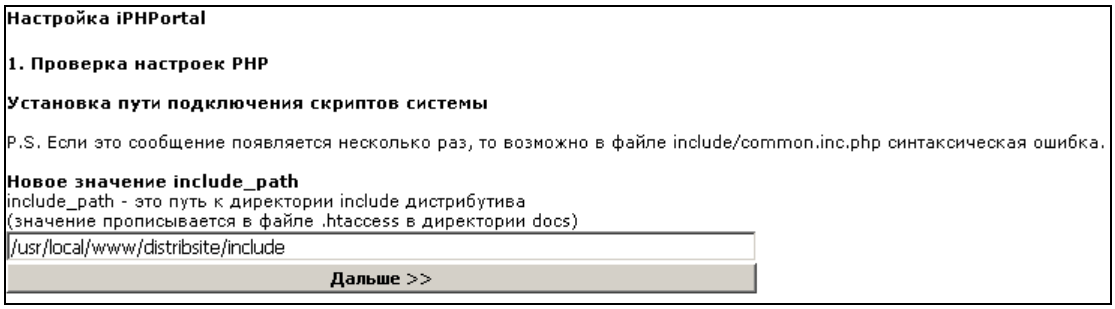

Рис. 2.4. Установка пути подключения PHP скриптов

Для того подключения скриптов системы необходимо указать путь к директории include дистрибутива. Если путь предложенный инсталлятором правильный – нажмите “Дальше”.

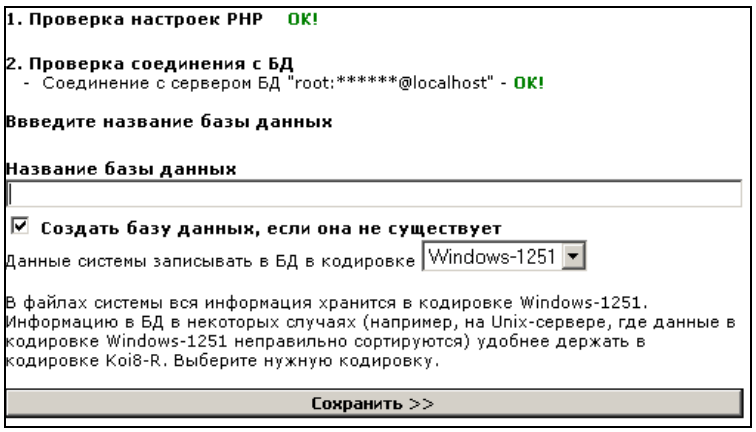

Рис. 2.5. Установка параметров соединения с БД

Для соединения с БД необходимо указать адрес сервера БД, логин и пароль пользователя БД (это не логин/пароль администратора системы, его нужно получить у администратора сервера БД). Этот шаг будет пропущен, если адрес вашего сервера БД “localhost”, логин “root” и пароль “”.

Затем введите название БД (например, iphportal) . После создания БД автоматически будет загружен дамп с таблицами системы (загрузка дампа может занять некоторое время). Если система не сможет автоматически загрузить дамп базы данных, то это необходимо сделать вручную. Информацию в БД можно загрузить в кодировке windows-1251 или koi8-r. В файлах системы вся информация хранится в кодировке Windows-1251. Информацию в БД в некоторых случаях (например, на Unix-сервере, где данные в кодировке Windows-1251 неправильно сортируются) удобнее держать в кодировке Koi8-R.

Ручная загрузка дампа базы данных:

Дамп загружается командой:
mysql –u пользователь –p пароль имя_базы_данных* < iphportal.sql**

* База данных должна быть предварительно создана
(выполнить в консоли MySQL): CREATE DATABASE iphportal

**Дамп базы данных iphportal.win.sql и iphportal.koi.sql находится в директории db дистрибутива.

Нельзя загружать дамп базы данных через PhpMyAdmin, т.к. он не загрузится полностью и работа системы будет нарушена.

После настройки БД вам нужно ввести администраторский логин/пароль (Administrator/123456).

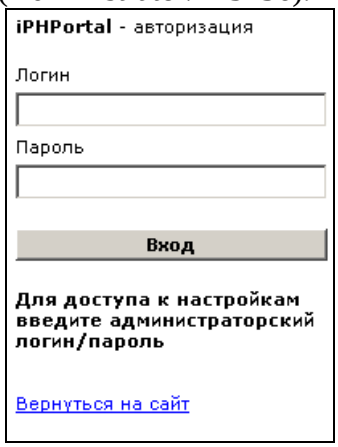

Рис. 2.6. Ввод логина/пароля администратора.

Внимание! Обязательно нужно изменить пароль администратора после установки!

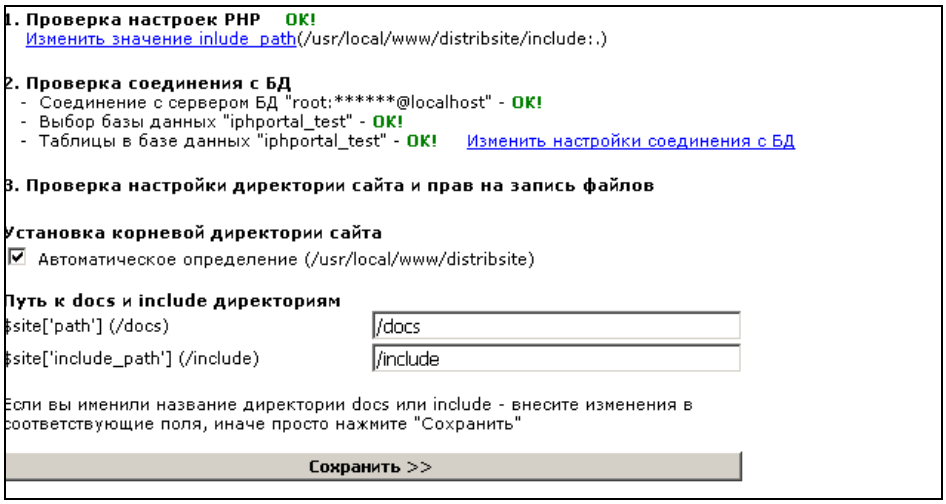

Рис. 2.7. Настройка переменной $site[‘root’]

Затем нужно установить переменную $site[‘root’] – директория в которую установлен дистрибутив. При автоматическом определении $site[‘root’] ставится как родительская директория DocumentRoot. Например, если настройка DocumentRoot равна /usr/local/www/distribsite/docs, тогда $site[‘root’] будет установлена как /usr/local/www/distribsite.
Переменные $site[‘path’] и $site[‘include_path’] изменяются, только если вы переименовали директории docs и include.

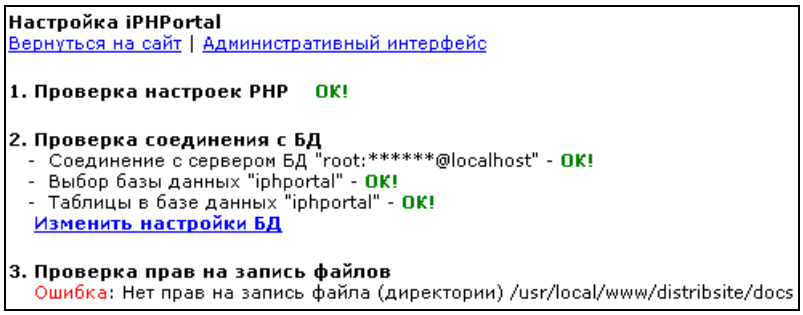

Рис. 2.7. Проверка прав на запись файлов.

Для работы системы необходимо чтобы у пользователя, от которого работает Apache, были права на запись в файлах и директориях:

docs
include/common.inc.php
include/cache/userrights/anonym
include/conf/functional.inc.php
include/conf/page_settings.inc.php
include/conf/settings.inc.php
include/events
include/handlers
include/meta
include/templates
include/temp

Скрипт настройки проверяет права на запись в этих директориях и в случае ошибки выдает предупреждение. Для выдачи прав на запись в директории / файле выполните в консоли команду **chmod -R 0777 {имя файла/директории}** или запустите shell-скрипт set_rights.sh, находящийся в корневой директории дистрибутива.

Рис. 2.8 Основые настройки системы

После выполнения обязательных проверок настроек PHP, соединения с БД и прав на запись файлов выполняется настройка параметров системы. Обязательно нужно заполнить поля “Название сайта” и “URL сайта”. В основных параметрах можно задать уровень отображения ошибок PHP и БД. По умолчанию показываются все ошибки PHP, и при возникновении ошибки в БД выводится сообщение об ошибке и прерывается выполнение скрипта.

Для очистки дистрибутива от занесенных рубрик, материалов, модулей и др. выберите закладку “Очистить дистрибутив” (рис. 2.9)

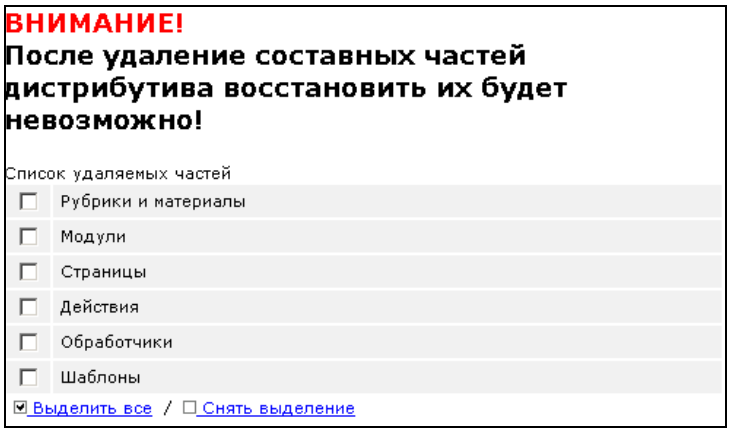

Рис. 2.9. Очистка дистрибутива

После установки системы перейти на настроечный скрипт можно нажав на ссылку “Настройки” в верхней части страницы административного интерфейса.

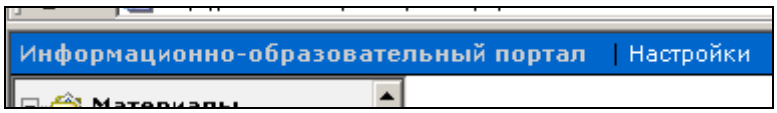

Рис. 2.10. Ссылка на настройки

5. Ручная настройка конфигурационных файлов (при необходимости)

Конфигурационны файлы:

- docs/.htaccess
- include/common.inc.php
- include/functional.inc.php (опционально)

В корне директории docs сайта размещается .htaccess файл следующего содержания:

```
#Путь для включения
#НЕОБХОДИМО ИЗМЕНИТЬ В СООТВЕТСТВИИ С ТЕМ, В КАКУЮ ДИРЕКТОРИЮ СКОПИРОВАН
#ДИСТРИБУТИВ
php_value include_path ".;E:/WWWApache/DevelopSite/include"


#ДАЛЕЕ НИЧЕГО ИЗМЕНЯТЬ НЕ НУЖНО
php_value register_globals 1
php_value magic_quotes_gpc  1
DirectoryIndex index.html index.php

#Для адресации "статических файлов"
<IfModule mod_rewrite.c>
#Выключить Multiviews если он включен в httpd.conf
Options -Multiviews
RewriteEngine  on
```

```
RewriteCond   %{REQUEST_FILENAME}  !-d
RewriteCond   %{REQUEST_FILENAME}  !-f

RewriteRule  (.*)/$                    $1.html

RewriteCond   %{REQUEST_FILENAME}  !-d
RewriteCond   %{REQUEST_FILENAME}  !-f

RewriteRule  (.*)            /action/error404.html
</IfModule>


# Если модуль mod_rewrite не подключен
<IfModule !mod_rewrite.c>
Options Multiviews
</IfModule>
```

Для работы системы необходимо чтобы директория include в дистрибутиве была включена в include_path PHP.

php_value include_path .:/full/path/to/site/root/include

Для того чтобы сгенерированные файлы материалов и рубрик обрабатывались php, в конфигурационном файле должна присутствовать строка:

AddType application/x-httpd-php .html .php .

В файле /include/conf/filepaths.inc.php можно настроить другое расширение для генерируемых файлов (переменная $site['material_file_ext']). В переменной $site['material_url_ext'] задается расширение “урла” материала. Расширение урла может быть '/', т.е. ссылка на материал вместо site.ru/material.html будет site.ru/material/ (реализуется через Options Multiviews или mod_rewrite).

При наличии установленного модуля “mod_rewrite” в .htaccess файле эмулируется Options Multiviews, когда при обращении к файлу /path/to/file/ Apache пробует открыть файл /path/to/file.html. Если файл не будет найден, то управление передается единому обработчику ошибок, который пробует сопоставить запрашиваемый URL с URL материала, для того чтобы отобразить статические файлы материала.

Если модуль mod_rewrite не установлен – тогда используется директива Options Multiviews, при этом теряется возможность адресации “статических файлов”. Функциональность “статические файлы” – это возможность загружать файлы в отдельную для каждого материала директорию, при отображении к содержанию статического файла добавляется header (шапка) и footer (подвал). Адресуются статические файлы по адресу материала. Например, материал имеет url http://site.ru/lib/programming/java/thinking-in-java/, тогда статический файл contents.html для этого материала будет иметь адрес http://site.ru/lib/programming/java/thinking-in-java/contents.html. Адресация статических файлов производится через скрипт docs/action/error404.html.

Файл include/common.inc.php

```
// Уровень вывода ошибок PHP
// На период разработки лучше выставить вывод всех ошибок
 $site['php_error_reporting'] = E_ALL;
 error_reporting($site['php_error_reporting']);

 // Данные доступа к базе данных

 $site['db_hostname'] = 'localhost';
 $site['database'] = 'iphportal';
 $site['db_username'] = 'root';
 $site['db_password'] = '';

//БД в кодировке koi8-r и нужна автоматическая перекодировка koi8r-windows1251
 $site['db_koiwin_recode'] = '1';

 // Директория сайта
 $site['root']='E:/WWWApache/site'; //корень всего сайта
 $site['path'] =  '/docs'; // Если нет отдельной web-директории, то $site[‘path’] = ‘’;


 $site['name'] = 'Site.Ru'; // название сайта
 $site['url'] = 'http://site.ru; // URL сайта

 $site['css_file'] = '/site-style.css'; // Css файл сайта, используется в визуальном редакторе


 $site['include_path'] = '/include'; // путь к директории где лежит common.inc.php, полный путь получается
                                     // $site[‘root’].$site[‘include_path’]. На эту директорию должен быть настроен
                                     // include_path в PHP
 $site['images_path'] = '/images'; // путь к директории с картинками, полный путь получается
                                     // $site[‘root’].$site[‘path’].$site[‘images_path’]

// Имя куки для авторизации на сайте
 $site['auth_cookie_name'] = 'authcook_558502';

  // модуль изменения картинок и поддерживаемые типы картинок (через запятую)
 $site['material_thumb_resize_module'] = 'IM'; // варианты IM (ImageMagick) или GD
 $site['image_magick_path'] = 'E:\ImageMagick\\'; //путь к директории где лежит ImageMagick
 $site['material_thumb_resize_formats'] = 'JPG,GIF';

//Если модуля изменения картинок нет , то последние строки можно удалить или закомментировать.

// Команда консоли для разархивирования zip архива
 $site['unzipfile_string'] = '$command = "F:\Distribs\zipstaff\powerarc.exe -e $filename";';
 // Для unix
 // $site['unzipfile_string'] = '$command = "/usr/local/bin/unzip -o $filename -d $full_real_path";';


// Права с которыми создаются новые директории
 // Если пользователь от которого закачиваются файлы и от которого работает Apache один и тот же - ставить 0700
 $site['dir_permission'] = 0777;

// При возникновении ошибок в БД
 // варианты: show,exit - показать описание ошибки и прекратить выполнение скрипта
 // show - показать описание ошибки и продолжить выполнение скрипта
 // exit - завершить выполнение не выводя сообщения
 // пусто - игнорировать ошибку и продолжить дальше выполнение скрипта
 $site['db_error'] = 'show,exit';
```

Если планируется управляет системой только частью сайта, например начиная с директории www.имя сайта.ru/site/, то настроить конфигурационный файл можно двумя способами :

1. Система настраивается на весь сайт, но для корневой рубрики не определяется “индексная страница” и в корневой директории остается файл, который правится вручную. Расширение этого файла должно отличаться от расширения файлов рубрик и материалов, которые создает система.
2. $site['path'] и $site[‘url’] прописываются вместе с директорией, например /docs/site и http://www.имя сайта.ru/site/. При создании обработчиков обязательно использовать полные ссылки на материалы, картинки и рубрики.

Если в базе данных уже существуют таблицы с именами, которые используется в таблицах системы, нужно переименовать таблицы БД системы и внести изменения в файл include/tables.inc.php

На некоторых хостингах пользователь, от которого работает Apache, и пользователь, от которого закачиваются по FTP файлы разные. И после закачки скрипты не могут изменять директории и файлы, которые были закачаны по FTP. В этом случае нужно обязательно изменить настройку права для создаваемых скриптом директорий – в файле conf/filepaths.inc.php $site['dir_permission'] нужно изменить на 0777 .

Скриптами изменяются файлы и директории в директориях[docs, include/conf/, include/events/, include/handlers/, include/meta/, include/templates, include/cache – на все эти и дочерние директории нужно дать права 777 (**все, что написано в данном абзаце относится только к случаю, когда разные пользователи Apache и FTP!!!)**. Затем либо поставить право 777 на все файлы в этим директориях.

Файл include/conf/functional.inc.php

```
// Проверять текущего пользователя на фронтофисе сайта или нет (больше нагрузка на сервер)
 $site['check_frontoffice_user']= 1;


 // Перегенерация материалов при первом просмотре
 $site['regenerate_delay'] = 1;

 // Использовать расширенную модель управления состоянием материалов или нет
 $site['use_extended_workflow_model'] = 1;


 //Имитировать что статические файлы материалов находятся в директории материала (Нужен ModRewrite)
 $site['material_static_files_rewrite'] = 1;

 // Возможно ли выполнение задач (публикация на опр.дату, expire и т.п.)
 // Для того чтобы производилось выполнение задач, должен периодически
 // вызываться файл admin/check_tasks.php (через cron или еще как-нибудь,
 //можно даже нанять человека:)
 $site['tasks'] = 0;


 // Возможен ли тип рубрики "Архив по дате"
 // Этот тип использовался 1 раз на первой версии runet.ru, так что скорее всего очмного глюков
 $site['rubric_type_date_archive'] = false;
```

```
//$site['banner_store_statistics'] = 14; // количество дней хранения статистика баннеров
$site['user_store_statistics'] = 30; // количество дней хранения логов пользователя
```

3. Создание сайта на базе системы

3.1 Порядок создания сайта

Все этапы создания сайта выполняются через административный интерфейс, находящийся по адресу имя сайта/admin

Для захода в административный интерфейс нужно ввести логин / пароль. Затем изменить логин и пароль администратора сайта на новый.

Логин / пароль администратора: Administrator / 123456

При установке системы на рабочий сайт необходимо изменить пароль администратора, а также проверить наличие неиспользуемых логинов и удалить их.

Основные этапы создания сайта:

1. Создание структуры сайта
2. Создание шаблонов для сайта
3. Создание обработчиков для сайта
4. Создание или загрузка страниц (наборов обработчик + шаблоны + опции редактирования)
5. Привязка страниц к рубрикам
6. Создание действий
7. Создание или загрузка и привязка к рубрикам типов рубрик (модули)
8. Определение видимых пользователю пунктов меню

Перед началом работы нужно очистить установленную систему от рубрик, обработчиков, шаблонов, версий шаблонов, действий, групп пользователей и пользователей, которые находятся в дистрибутиве (если они не будут использоваться).

3.2 Создание структуры сайта

Для создания структуры сайта выберите пункт меню “Рубрики”. См. раздел “Рубрики” руководства пользователя. При создании структуры сайта “страницы” индекса рубрики и материалов рубрики определять не нужно (“страниц” еще нет).

3.3 Создание шаблонов для сайта

Шаблоны для сайта создаются на основе имеющегося html-макета. Необходимо чтобы были html-макеты всех типов страниц сайта (индекс сайта, индекс рубрики, материал (статья) и т.п.). Для занесения шаблонов выберите пункт меню Генерация -> Шаблоны и затем “Добавить шаблон”.

Картинки дизайна сайта рекомендуется записывать в директорию docs/i/. Ссылки в шаблонах нужно прописывать с корня сайта, например, <img src=”/i/empty.gif”>, т.к. файл, сгенерированный по шаблону может находиться в любой директории и ссылка на картинку должна работать во всех случаях.

Шаблоны бывают четырех типов – шаблон страницы (этот шаблон можно выбрать при создании “страницы”), инклуд (эти шаблоны включаются в шаблоны страницы), шаблон инклуда (используются при предгенерации элементов сайта, например меню, а также при выводе баннеров) и “версии шаблонов” – используются для создания изменяемых элементов страниц, например меню, которое изменяется в зависимости от рубрики.

Для удобства создания и дальнейшего изменения, при создании шаблонов сайта, общие логические элементы для страниц (например “хидер” (шапка), “footer” (подвал), меню и т.п.) выделяются в “инклуды” (при занесении шаблона нужно выбрать “Тип шаблона” – “Инклуд”). В шаблоны страниц инклуды вставляются директивой

<!-- INCLUDE BLOCK : код_шаблона -->

Например, занесли шаблон - инклуд с названием “Меню” и кодом “menu” и шаблон страницы с названием “Материал” и кодом “material”. В то место текста шаблона страницы “Материал”, где должен быть текст инклуда нужно занести строку

<!-- INCLUDE BLOCK : menu -->

Место в шаблоне, где будет находиться некоторая строка, определяемая в обработчике - переменная шаблона, обозначается {имя переменной}.

Например, в шаблоне материала, в месте, где будет находиться заголовок материала нужно поместить {material_title}. Если переменной шаблона в обработчике ставится в соответствие массив, до доступ к элементам массива реализуется так {имя переменной.ключ массива}, например {image.width}. О том, как назначаются значения для переменных см. пункт 3.4.

Для создания динамических блоков используется директива

```
<!-- START BLOCK : имя_блока -->
 …. Текст блока
<!-- END BLOCK : имя_блока -->
```

Содержание динамического блока не будет выводится, если в обработчике не будет выполнен метод $tpl->newBlock (‘имя блока’). Если метод вызывается несколько раз – текст блока выводится несколько раз. Динамические блоки используются для вывода списка некоторых элементов, например строк таблицы. Данные для динамических блоков заносятся в обработчиках.

Пример динамического блока:

```
<table>
  <tr>
    <td>Название</td>
    <td>Количество</td>
  </tr>

  <!-- START BLOCK : row -->
```

```
  <tr>
    <td>{name}</td>
    <td>{quantity}</td>
  </tr>
  <!-- END BLOCK : row -->

</table>
```

Можно создавать вложенные динамические блоки.

Пример вложенных динамических блоков:

```
<table>
  <tr>
    <td>Название</td>
    <td>Количество</td>
    <td>Картинка
  </tr>

  <!-- START BLOCK : row -->
  <tr>
    <td>{name}</td>
    <td>{quantity}</td>
    <td>

<!-- START BLOCK : image -->

 <img src=”{image.url}” width=”{image.width}” height=”{image.height}”>

<!-- END BLOCK : image -->

</td>
  </tr>
  <!-- END BLOCK : row -->

</table>
```

Каждая директива шаблонного языка (INCLUDE BLOCK, START BLOCK и т.д.) должна быть на новой строке.

В шаблоны можно добавлять код PHP, заключенный в <? ?>. Например <?= date (‘d.m.Y H:I’) ?> - для вывода текущей даты. Этот код будет вызываться динамически – т.е. в предгенерированном файле будут эти самые команды, и они будут вызываться при каждом просмотре. Эта возможность используется для включения динамических блоков в предгенерированные статические страницы. Для того чтобы динамические вставки правильно показывались при прямом выводе содержания страницы методом $tpl->printToScreen() объект шаблона нужно создавать, указывая во втором параметре ‘NO_INCLUDE’

```
$tpl = new TemplatePower($templates['main_template'],’NO_INCLUDE’);
```

Если нужна возможность менять в title страниц и мета-теги то для их вывода используются директивы **<!—TITLE à и <!—META à**.

Пример:

```
<html>
 <head>
   <title><!—TITLE --></title>
    <!—META -->
 </head>
```

….

Для того чтобы вывелись title и meta, относящиеся именно к генерируемому материалу или рубрике, в обработчике необходимо указать

$tpl->assignPlacePath (‘путь к META-данным страницы’)

В обработчике, генерирующем материал, функция вызывается с параметром 'material/'.$material_data['material_id'], в обработчике, генерирующем индекс рубрики, с параметром 'rubric/'.$rubric_data['rubric_id']. Вызывается до метода prepare().

Иначе вместо <!—TITLE --> и <!—META --> будет пусто.

Редактировать обработчик можно двумя способами:

1. Через форму редактирования шаблона в административном интерфейсе
2. Редактировать файл шаблона. Текст шаблона хранится в базе данных, и после сохранения в базу записывается в файл include/templates/{код_шаблона}.tpl. При генерации страниц используется именно этот файл. Если его отредактировать, то измененный вид шаблона будет виден пользователям сайта при условии, что страница выводится динамически. Если шаблон используется при предгенерации материалов, то для перегенерации нужно выбрать пункт меню Генерация -> Шаблоны. Система автоматически находит изменившиеся файлы и загружает их в базу данных. Для того чтобы текст шаблона в базе данных обновился из файла, и перегенерировались все записи, созданные с использованием измененного шаблона нужно отметить чекбокс “Обновить из файла”. Если снять отметку с чекбокса, то содержание файла заменится текстом шаблона из базы данных.

3.4 Создание обработчиков для сайта

На разных сайтах версии системы отличаются, поэтому, скопировав обработчик со старого сайта необходимо проверить что, формат вызова функций классов (п.5 руководства) совпадает с форматом текущей версии.

Для комплексной загрузки обработчиков и шаблонов есть возможность загрузки “страниц” (см. п. 3.5).

Обработчик – это функция, автоматически вызываемая при генерации индекса рубрики, материала, при выполнении действия и при отображении модуля (типа рубрики). Для того чтобы увидеть список занесенных обработчиков выберите пункт меню Генерация -> Обработчики.

Для занесения обработчика нажмите на закладку “Добавить обработчик”. Затем введите название обработчика (например “Рубрика – список материалов”) и код обработчика (“rubric_index_materials” – название функции). Обработчик может быть трех типов: “Генерация в файл”, “Динамический” и “Инклуд”. Тип “инклуд” используется для разбиения больших обработчиков на части или для выделения части кода, которая

используется в нескольких обработчиках. Код “инклуда” включается в обработчик функцией include ('handlers/guestbooks_list.hdl');, где guestbooks_list – код обработчика-инклуда.

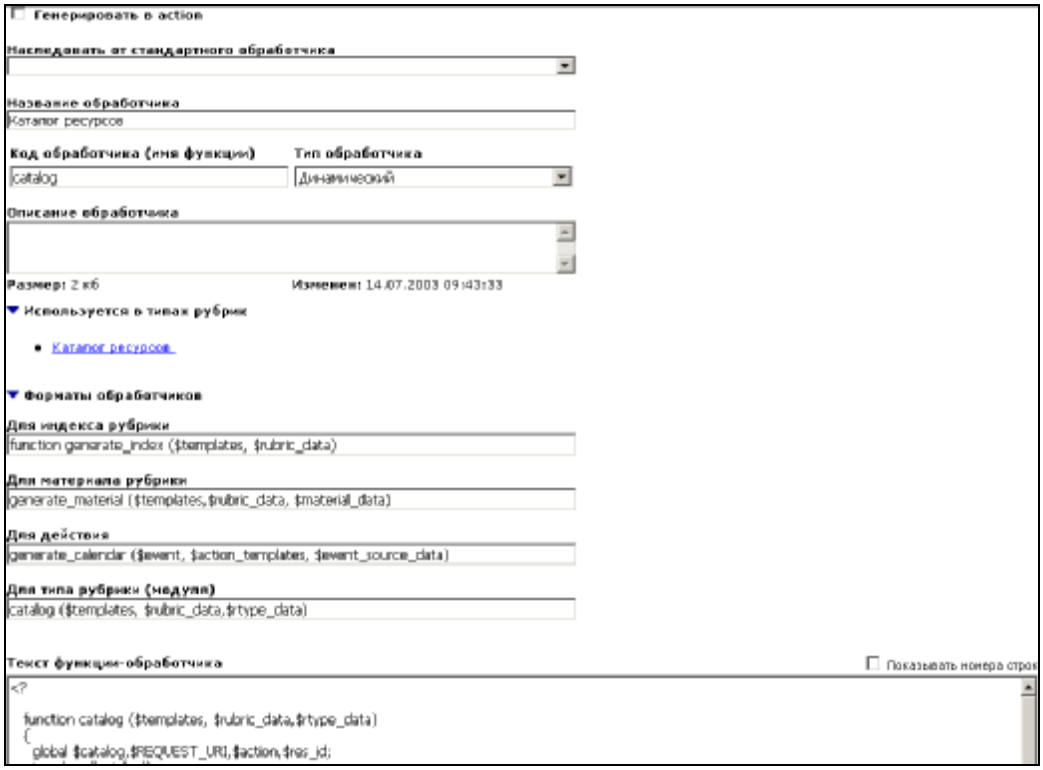

Рис. 3.1. Занесение / редактирования обработчика.

Для различных применений обработчики имеют разный формат вызова. Например, при генерации материала формат вызова такой:

```
function generate_material ($templates,$rubric_data, $material_data)
{
  // … текст функции
}
```

В массиве $templates находятся номера шаблонов для “страницы” (п.3.5) в которой используется обработчик. Главный шаблон - $templates[‘main_template’]. $rubric_data – данные рубрики, в которой генерируется материал, $material_data – данные генерируемого материала (про формат данных рубрики, материала см. раздел 5 руководства)

Чтобы увидеть возможные форматы нажмите на заголовок “Форматы обработчиков”. При редактировании обработчика показывается список рубрик, типов рубрик, где он используется. Для того чтобы показывалась текущая строка в поле редактирования текста обработчика, нажмите на чекбокс “Показывать номера строк”. Номер строки показывается в левом нижнем углу поля редактирования текста обработчика (Рис. 3.2). После изменения обработчика перегенерируются файлы, созданные с его использованием (если обработчик используется при генерации большого количества материалов то этот процесс может затянуться). Для того чтобы отменить перегенерацию снимите отметку с чекбокса “Обновлять связанные страницы”. Скрипты обновления вызываются в отдельном маленьком окне. Если нужно чтобы это окно не закрывалось после завершения работы скрипта (например, чтобы видеть отладочные сообщения) – для этого снимите отметку с чекбокса “Закрывать отладочное окно”.

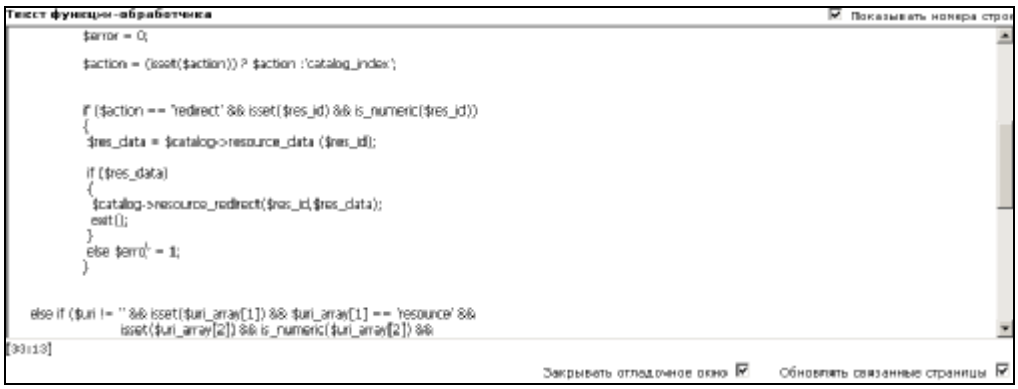

Рис. 3.2. Поле редактирования текста обработчика.

Редактировать обработчик можно двумя способами:

3. Через форму редактирования обработчика (рис. 3.1)
4. Редактировать файл обработчика. Текст функции-обработчика хранится в базе данных, и после сохранения в базу записывается в файл include/handlers/{код_обработчика}.hdl. При генерации страниц используется именно этот файл. Если его отредактировать, то измененная функциональность обработчика будет видна пользователям сайта при условии, что обработчик выполняется при каждом показе страницы сайта. Если обработчик используется при предгенерации материалов, то для перегенерации нужно выбрать пункт меню Генерация -> Обработчики. Система автоматически находит изменившиеся файлы и загружает их в базу данных (рис. 3.3). Для того чтобы текст обработчика в базе данных обновился из файла, и перегенерировались все записи, созданные измененным обработчиков нужно отметить чекбокс “Обновить из файла”. Если снять отметку с чекбокса, то содержание файла заменится текстом функции-обработчика из базы данных.

Рис. 3.5. Диалоговое окно при обновлении обработчика.

Результатом действия обработчика может быть сгенерированный файл индекса рубрики или материала, динамическая выдача индекса рубрики, генерация меню при выполнении действия и т.п.

Различие между типами обработчиков “Генерация в файл” и “Динамический” проявляется только, если обработчик используется в странице типа “Автогенерация” (индекс рубрики).

Если обработчик имеет тип “Генерация в файл” – он вызывается каждый раз при перегенерации индекса рубрики. Индекс рубрики перегенерируется при изменении параметров данной или дочерней рубрики или при изменении материала, входящего в данную или дочернюю рубрики. При этом результатом работы обработчика должна быть команда dump_rubric_file ($rubric_data, $tpl->getOutputContent()) – она записывает содержание страницы в файл.

При генерации индекса рубрики формат обработчика такой:

```
<?
function rubric_index_materials ($templates, $rubric_data)
{
  global $site;

  $tpl = new TemplatePower($templates['main_template']);
  $tpl->prepare();

  // Другие действия
  dump_rubric_file ($rubric_data, $tpl->getOutputContent());
}
?>
```

в этом обработчике создается объект “Шаблон”, в качестве параметра ему передается код шаблона страницы, который определен для генерируемого индекса рубрики. Далее шаблон парсится и результат записывается в индексный файл рубрики.

Если обработчик имеет тип “Динамический”, то обработчик выполняется каждый раз, когда просматривают соответствующую рубрику на сайте. Результатом действия “динамического” обработчика является выдача текста страницы, т.е. вместо dump_rubric_file ($rubric_data, $tpl->getOutputContent()); в конец функции-обработчика нужно поместить $tpl->printToScreen();. “Динамические обработчики” используются, например, в индексной странице рубрики “карта сайта”, на которой выводится дерево рубрик сайта.

В приведенном выше примере шаблону не передаются значения. Приведем пример шаблона и обработчика индекса рубрики, в котором выводится список материалов рубрики.

Шаблон страницы индекса рубрики:

```
<h3>{rubric.rubric_name}</h3>
<b>{rubric.rubric_desc}</b>

<ul>

   <!-- START BLOCK : material -->
   <li><a href=”{material.material_url}”>{material.material_title}</a>
   <!-- END BLOCK : material -->

 </ul>
```

Обработчик:

```
<?
function rubric_index_materials ($templates, $rubric_data)
{
  global $site,$materials;

  ins_class ('materials');

  $rubric_materials = $materials->materials_list($rubric_data['rubric_id'],false,
                                                 "material_publish='1'");

  $tpl = new TemplatePower($templates['main_template']);
  $tpl->prepare();

  $tpl->assign("rubric"   , $rubric_data );


  if ($rubric_materials)
  {
    foreach ($rubric_materials as $material)
    {
     $tpl->newBlock("material");
     $tpl->assign("material"   , $material);
    }
  }

  write_file($rubric_data['rubric_index_file'],$tpl->getOutputContent(),true);
}
?>
```

в обработчике сначала с помощью функции ins_class инициализируется класс "Материалы" (переменная класса $materials объявляется как global, т.к. возможно уже есть экземпляр этого класса).

Затем считывается список материалов рубрики методом materials_list класса materials. Метод имеет следующие параметры:

```
function materials_list ( $rubric_id = 1,
                          $include_subrubrics = true,
                          $add_condition = "  и др (см. раздел 5) )
```

$rubric_id - id рубрики материалов, по умолчанию 1 - корень сайта
$include_subrubrics - включать в список материалы, принадлежащие дочерним
рубрикам указанной рубрики
$add_condition - дополнительные условия для выборки

и др. (см. раздел 5)

и возвращает массив данных о материалах.

В приведенном примере $rubric_id = $rubric_data['rubric_id'] (id генерируемой рубрики), $include_subrubrics = false (не включать в список материалы дочерних рубрик), $add_condition = "material_publish = '1'" (включать в список только опубликованные материалы), остальные параметры по умолчанию.

Затем определяются значения для переменных шаблона rubric и material.

Если в рубрике есть опубликованные материалы – в шаблон заносятся заголовок и ссылка на эти материалы.

При генерации материала формат обработчика отличается количеством параметров в функции:

```
<?
function material ($templates,$rubric_data, $material_data)
 {
   ….
 }
?>
```

Обработчик генерации материала вызывается в следующих случаях:

- Занесение, обновление материала
- Изменение шаблона материала
- Изменение обработчика материала

Обработчик генерации индекса рубрики (генерация в файл) вызывается в следующих случаях:

- Занесение, обновление материала этой и нижележащих рубрик
- Изменение шаблона индекса рубрики
- Изменение обработчика индекса рубрики
- Сохранение данных этой или нижележащей рубрики

3.5 Создание “страниц” (наборов обработчик + шаблоны)

Для вывода занесенных “страниц” сайта выберите пункт меню Генерация -> Страницы.

“Страницы” – это набор из обработчика и шаблонов страниц, который используется для генерации индексов рубрик и материалов сайта + набор опций редактирования (для материалов).

“Страницы” могут быть двух типов “Автогенерация” и “Материал”. Тип “Автогенерация” используется для индексов рубрик, содержащих автоматически генерируемые списки, например список материалов рубрики или подрубрик рубрики.

В функцию - обработчик индекса рубрики или материала рубрики передается массив $templates, в котором определены индексы (если определен соответствующий шаблон):

$templates[‘main_template’] - Шаблон страницы (
$templates[‘print_template’] - Шаблон версии для печати
$templates[‘picture_template’] – Картинка

+ при необходимости другие дополнительные шаблоны (add_template1..3)

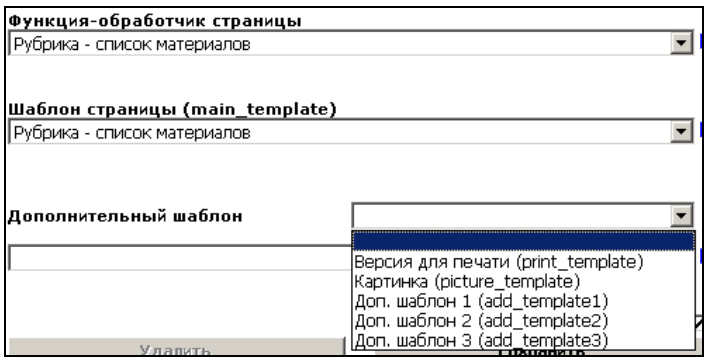

Рис. 3.6. Выбор дополнительных шаблонов страницы

Для страниц типа “Материал” можно указать, генерировать файл материала или нет. Если файл материала не генерируется – то указывать обработчик и шаблоны не нужно. Не генерируемые материалы могут использоваться при создании страниц с определенными списками (например, список работ компании), каждый элемент списка – материал, но отдельной страницы для каждого материала не требуется.

Для каждой страницы определяются опции редактирования рубрики (если страница установлена для индекса рубрики) или опции редактирования материала (если страница установлена для материала рубрики). Существуют “настройки страниц по умолчанию” – они используются на всех страницах, если на странице не отмечен чекбокс “Переопределить стандартные настройки”.

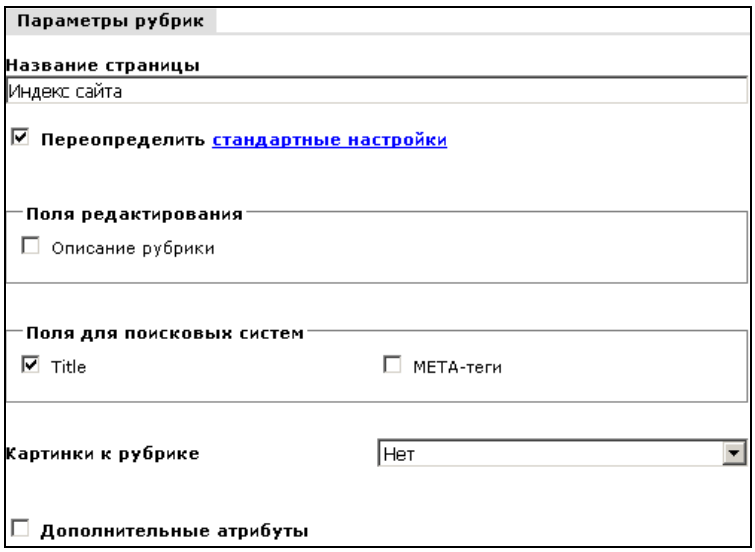

Рис. 3.7. Параметры редактирования рубрик.

В параметрах рубрик определяется, показывать ли поле “Описание рубрик”, поля для поисковых систем (Title и META-тэги) и количество картинок к рубрике. Для каждой картинки можно ввести название, если название не заполнено картинки именуются по порядку – “Сопровождающая картинка”, “Сопровождающая картинка 2” и т.д. Если для картинки определены ширина и высота, то при загрузке изображения любого размера, оно будет приведено к нужной ширине и высоте.

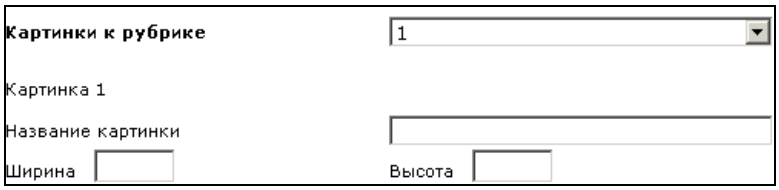

Рис. 3.8. Картинки к рубрике

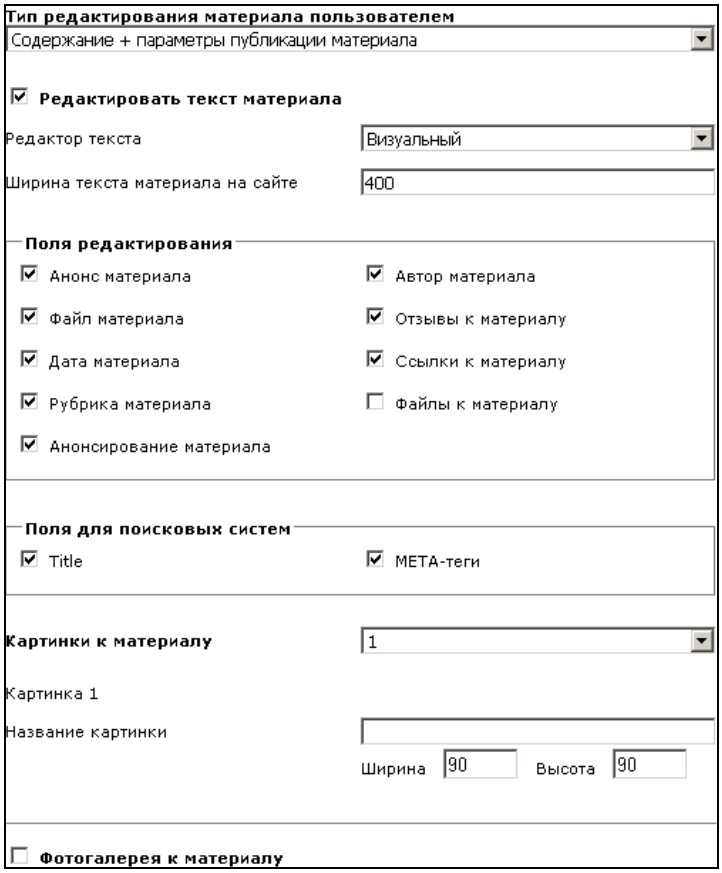

Рис. 3.9. Параметры редактирования материалов.

В параметрах редактирования материалов определяется тип редактирования материала. Варианты:

1. Одна страница со всеми параметрами материала
2. Содержание + параметры публикации материала
3. Многостраничный материал - одна страница со всеми параметрами
4. Многостраничный материал - содержание + параметры публикации'

Разделять страницы с содержанием и параметрами публикации имеет смысл, если редактируется много полей материала и удобнее разбить поля ввода для материала на 2 страницы.

Для некоторых типов материалов текст материала не нужен (например, материал состоит из заголовка, анонса и картинки). Чтобы при редактировании материала не показывалось поле с текстом материала - нужно снять отметку с чекбокса “Редактировать текст материала”.

Если текст материала редактируется, то нужно указать какой редактор будет использоваться по умолчанию – “визуальный” или “простой”. Указанный тип редактора будет использоваться, если у пользователя в настройках бэкофиса стоит “Для редактирования текста материала использовать редактор” – “По умолчанию для страницы материала”. При использовании “визуального редактора” контролируется размер таблиц, занесенных в материал. Если ширина таблицы будет больше значения поля “Ширина текста материала на сайте” – то перед сохранением материала пользователю будет выдано предупреждение. Это сделано для того, чтобы широкие таблицы не “распирали” дизайн сайта.

Набор параметров, которые нужно ввести для материала может меняться. Существует набор необязательных параметров материала, которые могут отсутствовать на странице (страницах) редактирования материала. Эти параметры:

1. Анонс материала
2. Файл материала
3. Дата материала
4. Рубрика материала
5. Анонсирование материала
6. Автор материала
7. Отзывы к материалу
8. Ссылки к материалу
9. Файлы к материалу
10. Статическое содержание материала
11. Связь с каталогом ресурсов

Для того чтобы поле показывалось при редактировании материала, нужно отметить соответствующий чекбокс в “Полях редактирования”.

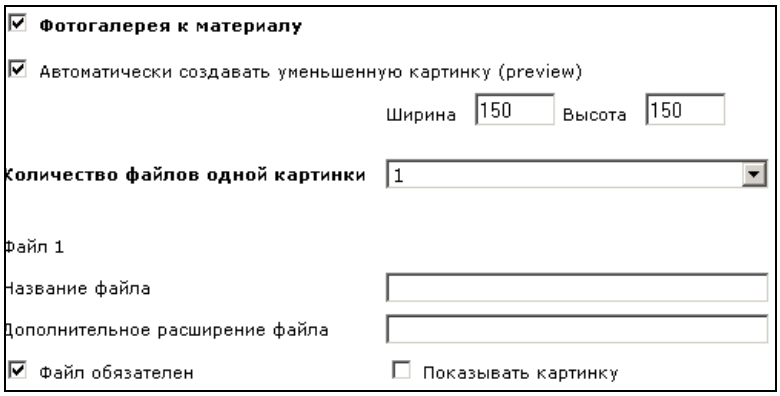

Рис. 3.10. Фотогалерея материала

В фотогалерее материала для каждой картинки может быть несколько файлов (например, разного качества). “Дополнительное расширение файла” требуется ввести, если для картинки заносится больше 1 файла. При отмеченном чекбоксе “Показывать картинку” загруженное изображение показывается на странице редактировании картинки, иначе показывается ссылка на изображение.

3.6 Дополнительные атрибуты

Для рубрик и материалов можно создавать дополнительные атрибуты. Для каждой “страницы” можно определить свой набор атрибутов.

Для занесения параметров дополнительных атрибутов выберите пункт меню Генерация –> Доп. атрибуты -> Добавить атрибут.

Типы атрибутов:

1. Строка
   Для ввода используется поле ввода или большое поле ввода <textarea> (чекбокс под выбором типа атрибута).

2. Число
3. Список

   При выборе типа “Список” необходимо еще ввести опции для этого списка. Под полем выбора типа атрибута выводится 5 пустых полей для ввода опций. Если опций больше 5 – введите сначала первые 5, затем сохраните атрибут, и под 5 введенными опциями появится пустое поле для ввода новой опции. Для удаления опции сотрите значение о описание нужной опции.

4. Логическое поле
5. Дата
6. Время
7. Дата и время

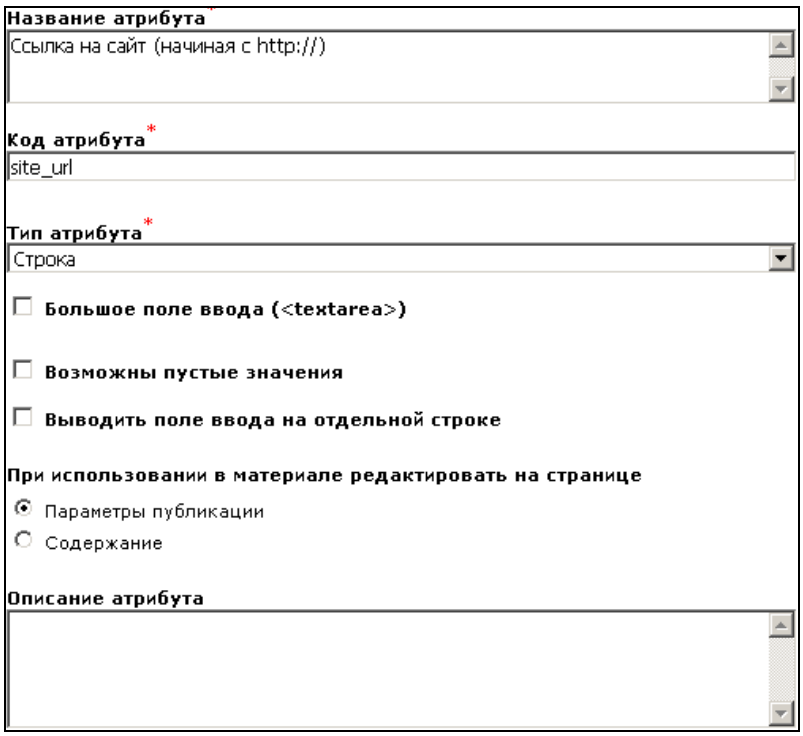

Рис. 3.11. Параметры дополнительного атрибута.

После занесения параметров дополнительного атрибута он связывается со “страницей”. Для этого при редактировании параметров страницы отметьте чекбокс “Дополнительные атрибуты”. В правом списке выводится список все занесенных атрибутов, в левой части выводятся атрибуты, которые используются на редактируемой странице.

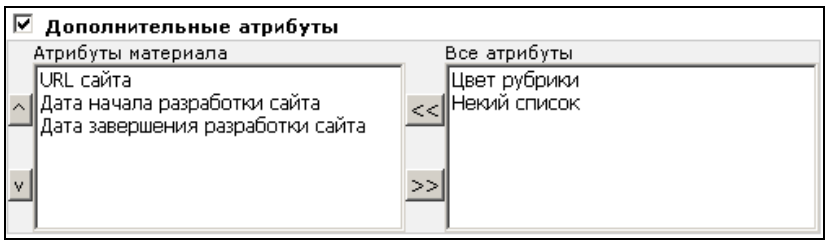

Рис. 3.12. Дополнительные атрибуты для страницы.

Дополнительные атрибуты используются для расширения стандартных атрибутов материала. С их помощью можно создавать документы (материалы) с произвольным набором полей. Для каждой рубрики или материала с дополнительными атрибутами в формируется массив вида

```
array ('some' =>
          array ('name' => 'Поле "Список"',
                 'value' => 'значение 1'),
        'some_text' =>
          array ('name' => 'Текстовое поле','
                  'value' => 'Текстик'),
        'some_boolean' =>
          array ('name' => 'Логическое поле',
                  'value' => '1'),
        'some_date' =>
           array ('name' => 'Поле "Дата"',
                   'value' => '1069016400'))
```

где ключ массива (some) – код атрибута. Этот массив хранится в сериализованном виде в таблице property_material_serialized (для материалов) и property_rubric_serialized(для рубрик).

В данных материала этот массив именуется $material_data[‘properties’]. Например, для того чтобы вывести на странице материала дополнительный атрибут “Текстовое поле” нужно в обработчике генерации материала добавить строку:

$tpl->assign (‘txt_property’,$material_data[‘properties’][‘some_text’]);

А в шаблон материала:

{txt_property.name} – {txt_property.value} (Выведется “Текстовое поле – Текстик’’’”)

Если запрашивается список материалов методом materials_list, то для того чтобы в данных материала были дополнительные атрибуты необходимо указать параметр $retr_properties равный ‘materials’ (см. раздел 5)

3.7 Привязка “страниц” к рубрикам и экспорт/импорт страниц

Созданные страницы привязываются к рубрикам. Для этого выберите пункт меню Рубрики и затем нужную рубрику.

Для того чтобы у рубрики был индекс, отметьте чекбокс “Индекс рубрики”. Индексы рубрик могут автоматически генерироваться по определенному шаблону (тип “Автогенерация”), могут быть редактируемыми как материал (тип “Материал”) или перенаправлять на другую страницу (тип “Редирект”). Индекс рубрики типа “Автогенерация” обычно содержит список ссылок на материалы рубрики (рубрика “О сайте, /about/”), список подрубрик рубрики (рубрика “Публикации”) или смешанные варианты.

Чтобы в рубрику можно было заносить материалы, нужно при создании рубрики отметить чекбокс “Материалы в рубрике”, затем выбрать в выпадающем меню страницу материала рубрики.

“Страница” состоит из множества элементов:

1. Обработчики
2. Шаблоны
3. Настройки страницы
4. Дополнительные атрибуты
5. Файлы (например, файлы картинок дизайна шаблона страницы)

Перенос страниц из одной копии системы в другую производится посредством экспорта/импорта страниц. При экспорте страницы все относящиеся к ней данные записываются в один xml-файл. При импорте этот файл просто нужно выбрать в форме занесения новой страницы.

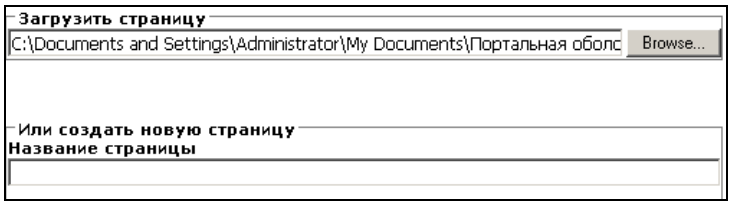

Рис.3.13. Загрузка страницы.

3.8 События и действия

Т.к. принцип системы управления сайтом основан на предгенерации страниц, бывает необходимо в ответ на какое то событие перегенерировать элемент сайта. Например, при изменении рубрики нужно перегенерировать меню, или при изменении материала изменить экспортный файл.

Для этого предназначена система событие – действие. Событие возникает при обновлении / добавлении / удалении рубрики (событие Рубрика), материала (событие Материал), отзыва (событие Отзыв), анонса (событие Анонс). Действие - это набор из обработчика и шаблонов, обработчик которого выполняется в ответ на определенное событие.

Для вывода списка занесенных действий выберите пункт меню Генерация –> Действия.

Перед занесением действия нужно создать обработчик, который будет выполнять это действие. Формат обработчика для действия такой:

```
<?
  function generate_menu  ($event, $action_templates, $event_source_data)
  {
     …
  }
?>
```

в переменной $event находится код события, вызвавшего действие;
в переменной $action_templates находится массив кодов шаблонов, которые связаны с этим действием;
в переменной $event_source_data находится данные элемента сайта, изменение которого вызвало действие, например для события Рубрика – данные рубрики.

При занесении действия заносится имя действия, указываются события, в ответ на которые выполняется действие и выбирается из списка обработчик, который будет выполнять действие.

После занесения действия нужно определить шаблоны для действия. Для этого на странице редактирование действия выберите закладку “Добавить шаблон”. Затем нужно ввести роль шаблона (название по-русски как шаблон используется в действии), код роли шаблона (название по-английски без пробелов) и выбрать шаблон. В списке представлены только шаблоны типа “Шаблон инклуда”.

Допустим занесли шаблон для действия с ролью “шаблон меню” и кодом роли “menu_template”, тогда в тексте обработчика создание объекта TemplatePower для этого шаблона выглядит так:

```
$tpl = new TemplatePower($action_templates['menu_template']);
$tpl->prepare();
```

Действие также происходит при изменении обработчика или шаблонов, входящих в это действие.

3.9 Типы рубрик (модули)

Модуль может быть двух видов:

1. С возможностью размещения в рубрике (например, “Форум” или “Каталог ресурсов”, т.е. имеющий представление на сайте)
2. Просто некая функциональность (например, “Баннеры” или “Голосование”).

Состав модуля:

1. Определение модуля (запись в базе данных)
2. Таблицы БД (дамп)

3. Классы, реализующие основную функциональность модуля (модель)
4. Страницы административного интерфейса
5. Пункты меню
6. Обработчики
7. Шаблоны
8. Группы прав, относящихся к модулю
9. Права, относящиеся к модулю
10. Объект права
11. Тип рубрики по умолчанию
12. Файлы к модулю
13. Настройки к модулю

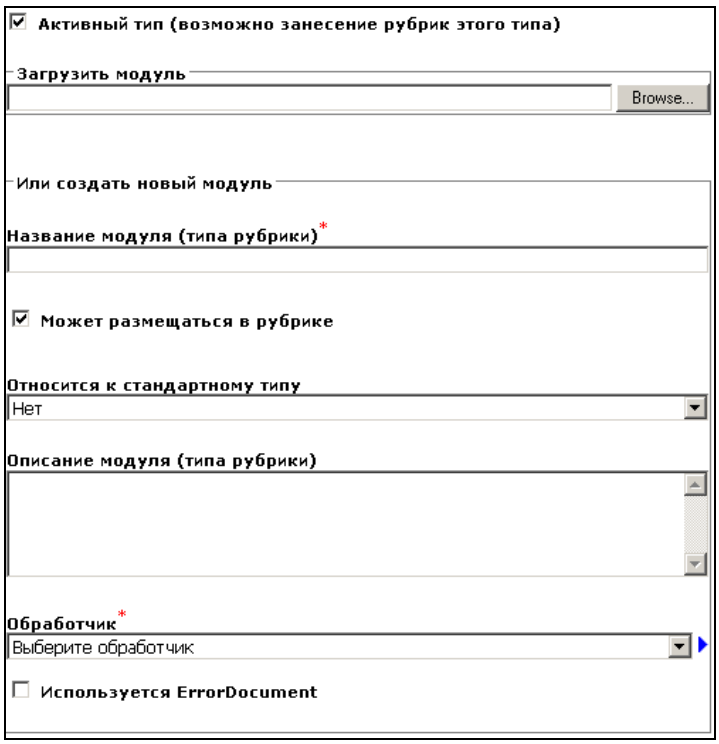

Рис. 3.14. Редактирование модуля (типа рубрики).

Перед занесением модуля, размещаемого в рубрике, нужно создать обработчик, который будет вызываться при заходе в рубрику. Формат обработчика такой:

```
<?
  function guest_book ($templates, $rubric_data, $rtype_data)
  {
  }
?>
```

$templates – массив с кодами шаблонов, которые определены для типа рубрики (модуля)
$rubric_data – данные рубрики
$rtype_data – данные типа рубрики

Функция-обработчик использует данные запроса (запрошенный URL или переданные через GET или POST данные) для вывода соответствующей страницы модуля. В обработчике типа рубрики (модуля) могут напрямую вызываться методы класса модуля,

либо главный обработчик в зависимости от выполняемой операции включает другие обработчики (обработчики типа “Инклуд”).

Рис. 3.15. Редактирование модуля (типа рубрики).

В функции-обработчике при создании объекта “Шаблон” используются на коды шаблонов, а их “алиасы” для типа рубрики. Алиасы (“код”) задаются при редактировании типа рубрики (модуля). Алиасы используются для того, чтобы при изменении шаблона для типа рубрики (модуля) не нужно было изменять код модуля. Например, код создания объекта шаблона в модуле “Форум”

```
$tpl = new TemplatePower($templates['forum_thread_messages'],'NO_INCLUDE');
```

при изменении шаблона, связанного с кодом “forum_thread_messages” изменять не нужно.

Тип рубрики “привязывается” к рубрике на странице редактирования параметров рубрики.

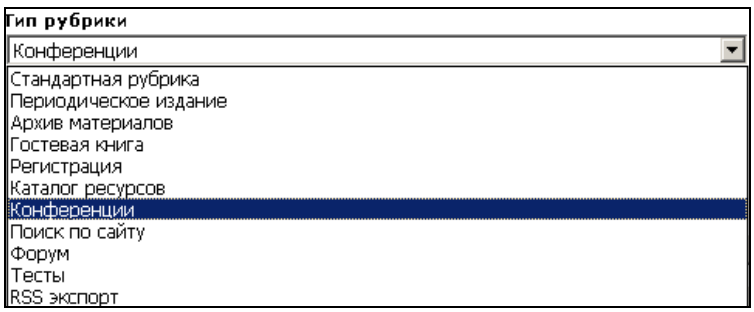

Рис. 3.16. Выбор типа рубрики при редактировании рубрики.

При этом в директории рубрики создается файл такого вида:

```
<?

// т.к. файл вызывается и при обработке 404 ошибки нужно выдать заголовок http 200 ok
header ("HTTP/1.0 200 Ok");
header ("HTTP/1.1 200 Ok");

// Подключение common файла
@require ("pregenerated_common.inc.php");
@require ("class/template.class.php");

// Подключение обработчика типа рубрики
@require ($site["handlers_path"]."/catalog.hdl");

// Создание экземпляра класса “Рубрики”
ins_class ('rubrics');

/*
 Запрос данных рубрики.
 Если в настройках сайта установлена опция “Авторизация на фронтофисе”, то при запросе данных рубрики запрашиваются и права пользователя на эту рубрику
*/
$rubric_data = $rubrics->rubric_data(134,  (isset($site['check_frontoffice_user']) && $site['check_frontoffice_user'] == '1')    ? true : false);

//сериализованный массив с данными типа рубрики
$rtype_data = unserialize ('a:19:{i:0;s:1:"6";……

….";s:13:"catalog-index";}}}');

/*
Вызов обработчика типа рубрики, обработчику передается массив с именами шаблонов типа рубрики, данные рубрики и данные типа рубрики
*/

catalog(array ('rubric_page' => 91,'res_page' => 92,'main_page' => 90),$rubric_data, $rtype_data);
?>
```

При занесении информации о типе рубрики нужно ввести название типа рубрики, определить относится ли тип к стандартному типу, выбрать обработчик и определить используется ли ErrorDocument. Стандартный тип рубрики используется для того, чтобы можно было определить, можно ли разместить в рубрике сущность некоторого модуля.

Примеры стандартных типов рубрик:

- Гостевая книга
- Форумы
- Тесты

Т.е. если тип рубрики (модуль) относится к стандартному типу “Форумы”, то значит в рубрике, имеющей этот тип, можно разместить форум, аналогично для гостевых книг и тестов. Различаются два подхода к созданию модулей – когда сущности модуля связаны с рубриками и когда не связаны. Например, каждый форум, гостевая книга и тесты привязаны к определенной рубрике, в которой они показываются (это сделано для удобства возможной рубрикации). Другие модули (каталог ресурсов, поиск по сайту) не требуют дополнительной рубрикации.

Записи о стандартных типах рубрик хранятся в таблице “rubric_type_default”.

Поля таблицы rubric_type_default.

Поле	Описание
def_rtype_code	Код стандартного типа рубрики
def_rtype_name	Название стандартного типа рубрики
def_rtype_script	Код, который выполняется при редактировании рубрики с установленным типом, относящимся к данному стандартному типу. Например, при редактировании рубрики с установленным типом рубрики “Гостевые книги” (стандартный тип “Гостевые книги”) исполняется код: ins_class ('guestbooks'); $guestbooks->show_guestbooks_info ($rubric_id); этот код показывает список гостевых книг, относящихся к редактируемой рубрике.

Прием ErrorDocument используется для имитации файлов и директорий на сайте. Если в типе рубрики отмечен чекбокс “Используется ErrorDocument”, то в директории рубрики создается файл .htaccess следующего содержания:

```
# Если используется Mod_Rewrite – отключить его, иначе управление будет передаваться
# главному обработчику 404 ошибки
RewriteEngine Off

#при 404 ошибке передавать управление индексному файлу рубрики
ErrorDocument 404 /catalog/index.html
```

К примеру, есть тип рубрики “Гостевая книга” и рубрика “Вопрос – ответ” этого типа. Для вывода сообщений гостевой книги используется адрес вида http://site.ru/question-answer/common/, хотя директории common нет. Т.к. в типе рубрики указано что используется ErrorDocument, даже если файл или директория не найдены выполняется обработчик типа рубрики. В обработчике этого типа делается следующее:

```
// обьявляется глобальной переменная с запрошенным адресом
global $REQUEST_URI;

// Запрошенный адрес без адреса рубрики и без последнего слеша
$uri = ereg_replace ("/$","",str_replace ($rubric_data['rubric_url'],",$REQUEST_URI));

//Запрошен индекс рубрики
if ($uri == '')
    {
     //считываем список гостевых книг
     $guestbooks_list = $guestbooks->guestbooks_list($rubric_data['rubric_id'],
                "guestbook_show = '1' AND guestbook_active = '1'");

    //выводим список гостевых книг
```

```
    ….
    // Завершаем работу
    exit();
  }
  else
  {
    // Разбиваем адрес
    $uri = explode ('/',$uri);

    //адрес состоит из одной директории
    if (sizeof($uri) == 1)
     {
       // считаем что это код рубрики
       $guestbook_code = $uri[0];

       // Считываем данные гостевой книги
       $guestbook_data = $guestbooks->guestbook_data($guestbook_code);

       // Если есть гостевая с таким кодом
       if ($guestbook_data)
       {
         //Выводим сообщения гостевой книги
         ….
         //Завершаем работу
         exit();
        }
     }
  }

  //Если дошли до этого участка кода и не завершили работу – значит либо url
  // непонятный, либо нет гостевой книги с указанным кодом – перенаправляем на индекс
 //рубрики

 header ('Location:'.$rubric_data['rubric_url']);
```

Порядок написания модуля:

1. Создать структуру таблиц БД, которые будет использовать модуль.
2. В директории include/class/ создать файл с классом модуля, имя файла должно быть равно названию класса. Например, если класс называется forums, то файл класса будет называться forums.class.php.
3. Создать страницы с административным интерфейсом в директории /docs/admin/. Если для модуля нужно отдельное право (например, право “Управление форумами”) – право заносится на странице “Настройки бэкофиса –> Права пользователей” (/admin/user_rights.php) . Если заносится новое объектное право, то нужно доработать функцию check_entry_user_right класса для проверки наличия права на объект.
4. Создать шаблоны для отображения модуля на сайте.
5. Создать обработчик (обработчики) для наполнения шаблонов
6. Создать модуль (тип рубрики), который объединяет обработчик (обработчики) и шаблоны и затем привязать тип рубрики к определенной рубрике.

Правила, которые необходимо соблюдать при создании модуля:

1. Все SQL запросы инкапсулированы в методы класса модуля, в обработчике модуля вызываются только методы, напрямую запрашивать информацию из таблиц модуля нельзя.
2. Страница администрирования модуля соответствует шаблону:

```
<?
 require ("admin_common.inc.php");
?>
<html>
<head>
<link rel="stylesheet" type="text/css" href="admin.css">
<title>Backoffice</title>
<meta content="text/html; charset=windows-1251" http-equiv=Content-Type>

<script language=”Javascript”>
// Java-скрипты административной страницы
</scipt>

</head>

<?
 // Закладки
 $bookmarks = array
 (
   'Закладка без ссылки',
   array (‘Закладка с ссылкой’,’some_link.php’)
 );

 // Включение меню бэкофиса (в этом же показываются и закладки)
 require ("./menu.php");
?>

<!--- код административной страницы  à

 <?
 require ("./menu_end.php");
?>
```

Оформление административных страницы должно быть в общем стиле, с использованием классов из файла admin/admin.css. Для каждой административной страницы нужно прописать права пользователя, требуемые для доступа к этой странице (пункт меню Настройки бэкофиса –> Страницы бэкофиса)

3. Общие блоки для шаблонов страниц выделяются в инклуды.
4. В обработчиках используются “алиасы” шаблонов, а не непосредственно имена шаблонов.

Требования к классу модуля

1. Названия таблиц в классе модуля должны быть прописаны следующим образом:

```
  // Конструктор
   function forums()
   {
       global $site;
       $this->mForumsTable = isset($site['forums_table']) ? $site['forums_table'] : 'forum';
```

```
}
```

Для возможности переопределения названий таблиц в файле include/conf/tables.inc.php

Так же должно присутствовать свойство класса $_used_tables, в котором перечислены таблицы, используемые классом. В модуле может быть несколько классов и в классе прописываются таблицы, используемые именно этим классом.

Пример:

```
var $_used_tables = array ('forum','forum_message');
```

2. В классе должны быть два метода для создания / удаления таблиц класса – install и uninstall. Метод install вызывается при импортировании модуля, метод uninstall при удалении модуля. SQL-файл со скриптом создания и наполнения таблиц модуля должен лежать в директории include/class/sqldump и должен иметь имя равное имени файла класса, только расширение не “.class.php”, а “.sql”

Пример метода install:

```
//Инсталлятор таблиц
// $table_prefix - префикс для создаваемых таблиц,
//в случае если таблицы с такими именами уже есть
function install ($table_prefix = '', $ignore_warnings = '')
{
 global $module_exchange;
  // Создание экземпляра класса module_exchange
 ins_class ('module_exchange');


 // В warnings записывются предупреждения при инсталляции таблиц модуля
 $warnings = array();

// Оставить имеющиеся таблицы
if (isset($ignore_warnings['install']['forum_tables']) &&
    $ignore_warnings['install']['forum_tables'] == 'exists') return $warnings;

 // Проверка наличия таблиц форума в БД
// Если есть хотя бы одна таблица, из перечисленных в переменной $_used_tables – возвращается true
$tables_exists = $module_exchange->check_class_tables ($this->_used_tables);

// Предупреждения, возвращаемые из этого метода, показываются пользователю с выбором
// “Создать таблицы заново” (overwrite) или оставить имеющиеся (exists, по умолчанию)
// После того как пользователь сделал выбор, метод вызывается заново, в переменной $ignore_warnings
// содержится выбор пользователя
if ($tables_exists &&
    (!isset($ignore_warnings['install']['forum_tables']) ||
    $ignore_warnings['install']['forum_tables'] != 'overwrite') )
{
$warnings['forum_tables'] = 'Таблицы БД модуля Форумы уже существуют';
}


        if (!$warnings)
        {
         // Создание таблиц
        // Запускается sql файл со скриптом создания и наполнения таблиц модуля
         $module_exchange->process_sql_dump ('forums');
```

```
        }

        return $warnings;
}
```

Пример метода uninstall

```
// Удаление модуля
function uninstall ()
{
 global $module_exchange;
        ins_class ('module_exchange');

 // Для каждой таблицы, перечисленной в переменной    $_used_tables вызывается команда
// DROP TABLE IF EXISTS $table_name
 $module_exchange->drop_class_tables ($this->_used_tables);
}
```

В методе uninstall желательно производить удаление файлов, которые могли быть загружены для записей модуля (например, в модуле “Баннеры” для записи в таблице banners может загружаться картинка баннера).

Принцип экспортирования модуля

При экспорте модуля все данные, относящиеся к модулю, записываются в один xml-файл. В xml-файл прописываются не только описания других файлов (классов, обработчиков, шаблонов и т.д.), но и содержание файлов, закодированное в формате base64. Затем этот файл автоматически архивируется для уменьшения размера.

Пример экспортного файла (поля с текстом base64 не заполнены)

```
<?xml version="1.0" encoding="windows-1251"?>
<module version="1.0" name="Форум" desc="Древовидный форум" handler_code="forum" errordocument="true"
def_rtype_code="forum" def_rtype_name="Форум" def_rtype_script="">
    <!--Классы, используемые модулем-->
    <classes>
        <class code="forum" editdate="" text="" sqldump="" sqldump_editdate=""/>
    </classes>
    <!--Административные страницы модуля-->
    <adminpages>
        <adminpage filename="forums.php" editdate="" text="">
            <requredright code="manage_forum"/>
        </adminpage>
    </adminpages>
    <menugroups>
        <menugroup name="Интерактив" image=""/>
    </menugroups>
    <!--Пункты меню модуля-->
    <menuitems>
        <menuitem menugroup="Интерактив" name="Форумы" link="forums.php"
highlightpages="forums.php,forum_edit.php">
            <requredright code="manage_forum"/>
            <menuitem name="Форумы" link="forums.php" highlightpages="forums.php,forum_edit.php">
                <requredright rightcode="manage_forum"/>
            </menuitem>
        </menuitem>
    </menuitems>
    <!--Обработчики-->
    <handlers>
```

```
        <handler name="Форум" desc="" code="forum" type="dynamic" text="" editdate=""/>
    </handlers>
    <!--Шаблоны-->
    <templates>
        <template name="Форум - список форумов" desc="" code="forum-list" alias="forum_list" text="" editdate=""/>
    </templates>
    <!--Группы прав пользователя-->
    <userrightgroups>
        <userrightgroup name="Форум"/>
    </userrightgroups>
    <!--Права пользователя-->
    <userrights>
        <userright name="Управление форумами" desc="Описание права" code="manageforum" rightobject="forum"
rightgroupname="Форум"/>
    </userrights>
    <!--Объекты прав пользователя-->
    <rightobjects>
        <rightobject name="Форум" code="forum" name2="форумы" name3="" table="" id_field="" name_field=""/>
    </rightobjects>
    <!--Настройки, относящиеся к модулю-->
    <settings>
        <setting group_code="site" name="Кол-во тем форума на странице" value="15" type="numeric" allow_empty="0"
code="site_forum_threads_show" desc=""/>
    </settings>
    <!--Файлы, относящиеся к модулю-->
    <files>
        <file path="" data="" editdate=""/>
    </files>
</module>
```

Экспортирование модуля

На странице редактирования модуля выберите закладку “Экспорт модуля”. Для экспортирования модуля обязательно должна быть заполнена мета информация, которая не обязательна для функционирования модуля:

1. Классы, которые использует модуль
2. Административные страницы модуля
3. Пункты меню модуля
4. Права, которые использует модуль
5. Настройки, используемые модулем
6. Файлы, которые использует модуль

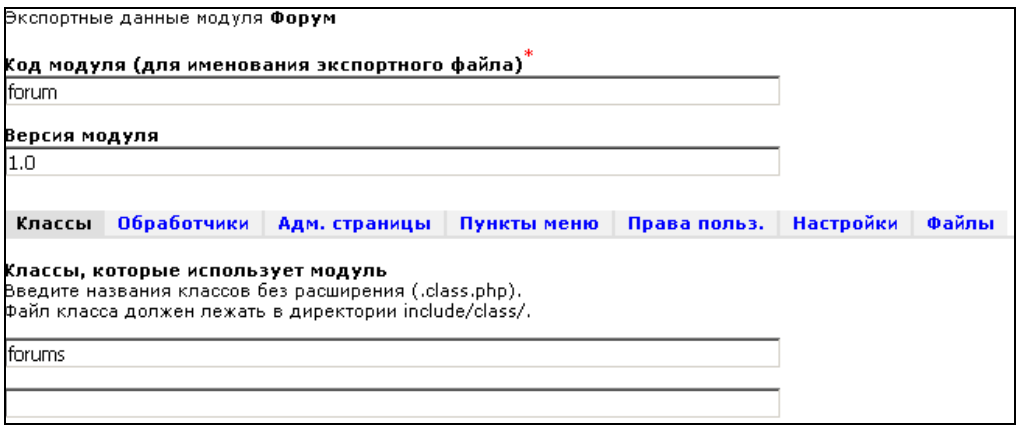

Рис. 3.18. Экспортные данные модуля

Примечание:

- Все права, которые прописаны для административных страниц и для пунктов меню модуля автоматически прописываются в список прав, используемых модулем.

- Шаблоны, которые включается в шаблоны модуля, автоматически включаются в экспортный файл (аналогично для обработчиков).

После заполнения информации о модуле нажмите кнопку “Экспортировать модуль”.

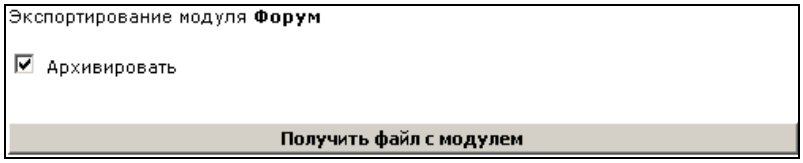

Рис. 3.19. Получение экспортного файла модуля

Импортирование модуля

Для импортирования модуля выберите пункт меню Генерация -> Модули, затем закладку ”Добавить модуль”.

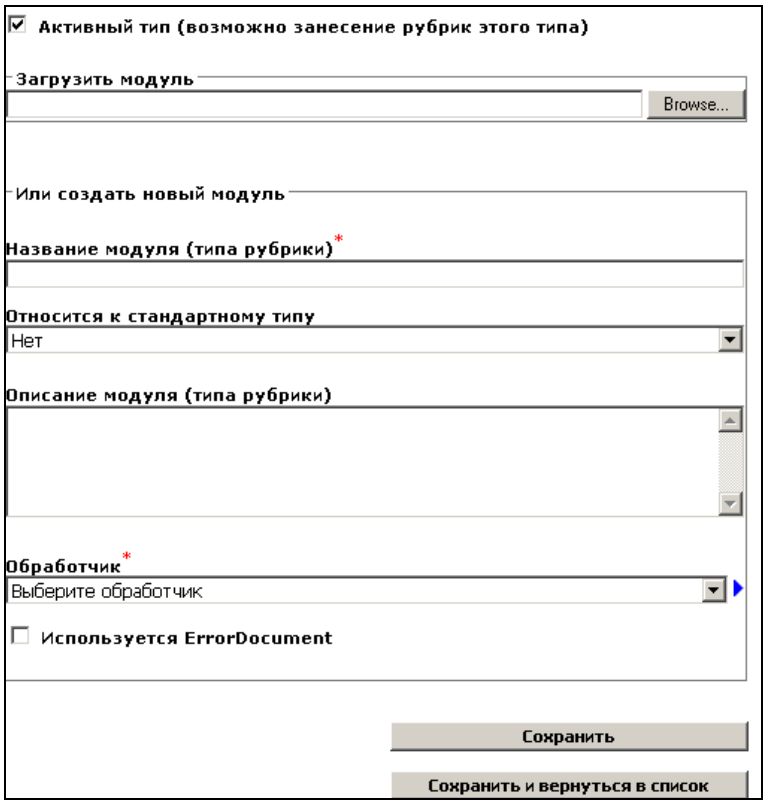

Рис. 3.20. Страница занесения модуля

Модуль можно создать вручную (ввести название, обработчик, шаблоны и т.д.), либо загрузить предварительно созданный экспортный файл.

При импортировании модуля производится проверка:

1. Проверка наличия модуля с таким же кодом или названием
2. Проверка наличия классов модуля
3. Проверка наличия административных страниц модуля
4. Проверка наличия объектов прав пользователя
5. Проверка наличия прав пользователя

Если существуют объект с таким же названием (например, класс, административная страница, обработчик и т.д.) и его параметры отличаются от данных в экспортном файле – выводится предупреждение.

По умолчанию ответ на предупреждение “Записать версию из экспортного файла” (для всех объектов, кроме шаблонов).

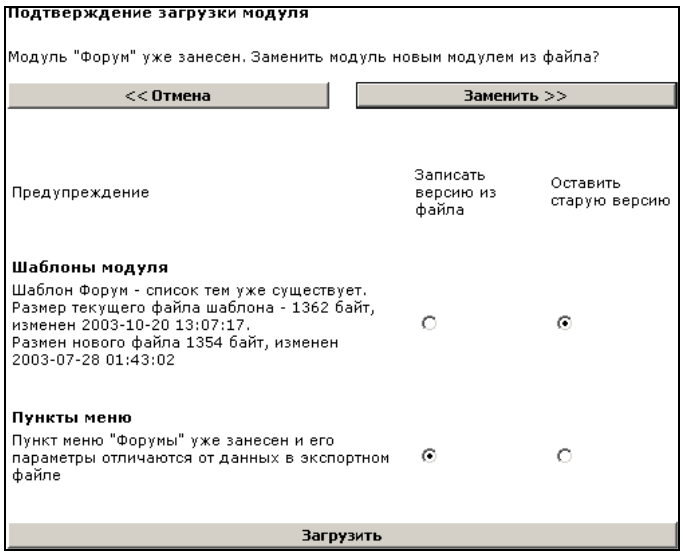

Рис.3.21. Подтверждение загрузки модуля

Если в БД уже существует хотя бы одна таблица, используемая модулем, выводится предупреждение. Можно оставить имеющиеся таблицы или загрузить дамп таблиц модуля заново.

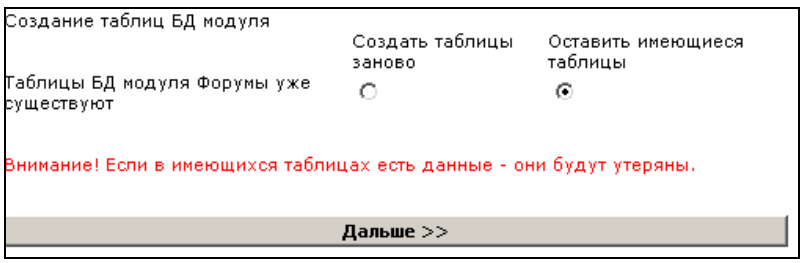

Рис.3.22. Создание таблиц модуля

После импортирования новый модуль можно разместить в рубрике.

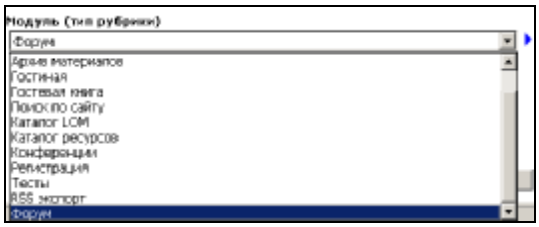

Рис.3.23. Выбор модуля (типа рубрики) размещенного в рубрике

4. Страницы бэкофиса для управления системой

Страницы бэкофиса, предназначенные для разработчика, не показываются в меню бэкофиса. Попасть на эти страницы, можно только набрав нужный адрес в браузере. Пользователь должен входить в группу “Администраторы”, чтобы иметь возможность видеть эти страницы.

4.1 Пункты меню

Для редактирования меню бэкофиса введите адрес страницы /admin/admin_menu_items.php (пункт меню Настройки бэкофиса -> Пункты меню). Неиспользуемые на конкретном сайте пункты меню нужно скрыть (снять отметку с чекбокса “Показывать пункт меню”).

Название	Ссылка	Права	Страницы	
Материалы				
↑↓ ☑ Добавить материал	material_new.php	Автор материалов Автор материалов + занесение материалов других авторов	material_new.php	+
↑↓ ☑ Занесенные материалы	materials.php	Просмотр материалов Автор материалов Автор материалов + занесение материалов других авторов Редактор	materials.php, ,material_edit_content.php, ,material_edit_properties.php	+
↑↓ ☑ Анонсы материалов	highlights.php	Управление группами анонсов Анонсирование материалов	highlights.php, ,highlight_group_edit.php	+

Рис. 4.1. Пункты меню бэкофиса

Для каждого пункта меню вводится название пункта меню, ссылка, права пользователя, необходимые для того, чтобы видеть этот пункт меню. Также нужно ввести через запятую имена файлов страниц администратора, при нахождении на которых пункт меню выделяется.

4.2 Права пользователя

Для редактирования прав пользователя нужно перейти на страницу /admin/user_rights.php (пункт меню Настройки бэкофиса -> Права пользователей).

Неиспользуемые права нужно скрыть от пользователя. При создании новых функциональностей, на которые нужно отдельное право, заносятся новые права пользователей.

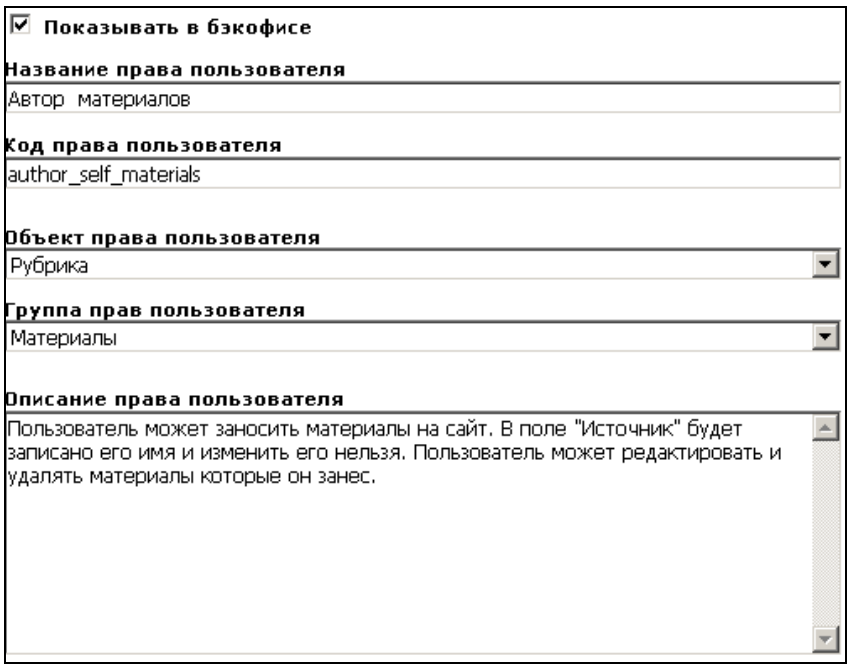

Рис.4.2. Занесение права пользователя

Право может быть обычным (право у пользователи или есть или нет) или объектным – можно выбрать объекты на которые распространяется это право. Объекты права хранятся в таблице uright_object.

Поля таблицы uright_object:

Поле	Описание
uobj_name	Название объекта права. Например, объект права для рубрик и материалов – “рубрика”
uobj_code	Код объекта (‘rubric’)
uobj_name2	Название права в множественном числе (“рубрики”)
uobj_name3	Название подрубрик объекта (для рубрик – “с подрубриками”)
uobj_class	Имя класса, который использует этот объект (например, rubrics)
uobj_table	Таблица, где хранятся записи с объектами права. В поле записывается название свойства класса, в котором записано название таблицы. Для объекта “рубрика” в этом поле записано “mTreeTable”
uobj_id_field	Имя поля в таблице с id записи (для рубрик – rubric_id)
uobj_name_field	Имя поля в таблице с названием записи (для рубрик – rubric_name)

Для того чтобы создать новый объект права нужно занести в таблицу uright_object новую запись. Затем указать, что созданный объект является объектом определенного права и создать страницу, на которой выбираются объекты для права. Страница именуется user_group_rights_edit_{код объекта права}.php, например для рубрик - user_group_rights_edit_rubric.php. По умолчанию право дается на объект с entry_id = 1 – на все объекты. Это было сделано т.е. в дереве рубрик корневой раздел имеет id = 1. Если структура записей в таблице с объектами не древовидная, то в этой таблице не должно быть записи с id=1. Т.е. если объект права – форум, в таблице не должно быть записи с forum_id = 1.

4.3 Страницы администратора

Для редактирования списка страниц бэкофиса сайта нужно перейти на страницу /admin/admin_pages.php (пункт меню Настройки бэкофиса-> Страницы бэкофиса). При заходе на любую страницу бэкофиса проверяются права на доступ к этой странице. Если страница не занесена в базу данных, то это равносильно тому, что у пользователя нет прав на эту страницу.

5. Основные функции для создания обработчиков

5.1 Класс Materials

Класс предназначен для управления материалами на сайте.

Данные материала:
material_data($material_id)

$material_id – id материала

Возвращает массив $material_data со всеми полями таблицы material для записи с material_id = $material_id

Кроме того:

material_data['material_file_name'] – текущее имя файла с расширением
$material_data[‘material_rubrics’] - список рубрик, которым принадлежит материал, см. функцию material_rubrics

$material_data[‘image’] – список сопровождающих картинок материала (не фотогалерея!)

определить наличие у материала сопр. Картинки можно так

if (isset($material_data[‘image’][0])) {// Есть первая картинка}

У каждой картинки есть параметры

url – ссылка на картинку
full_url – полная ссылка (с именем сайта)
width – ширина
height – высота
size – размер в байтах
type – тип,GIF,JPG или PNG

Доступ к этим параметрам - $material_data[‘image’][0][‘url’]

$material_data['pages'] – список страниц материала,
$material_data[‘material_pages’] - количество страниц,
material_url, material_full_url,material_print_url – ссылка версии для печати

Список материалов:

```
function materials_list ( $rubric_id = 1,
                          $include_subrubrics = true,
                                                $add_condition = '',
                                                $order = '',
                                                $limit = '',
                                                $retr_properties = '',
                                                $join_pages = false,
                                                $join_highlights = false,
                                                $retr_material_text = false )
```

$rubric_id - id рубрики материалов, по умолчанию 1 - корень сайта
$include_subrubrics - включать в список материалы, принадлежащие дочерним рубрикам указанной рубрики
$add_condition - дополнительные условия для выборки (sql where условие, например “material_publish = ‘1’ AND material_date > ‘$some_date’”)
$order - порядок сортировки (по умолчанию material_date DESC)
$limit - количество возвращаемых материалов (по умолчанию – неограниченно)
$retr_properties – строка, определяющая запрашивать или нет дополнительные атрибуты материалов и рубрик материала. Может быть пустой (не запрашивать атрибуты), ‘material’ – запрашивать атрибуты материалов, ‘rubrics’ – запрашивать атрибуты рубрик материала или ‘material,rubrics’ – запрашивать все атриубуты. Набор атрибутов материала / рубрики зависит от настроек страницы материала / индекса рубрики. Доступ к свойствам - $material[‘properties’][код атрибута], аналогично для рубрики.

$join_pages – выполнять ли объединение с таблицей страниц материала (используется для поиска)

$retr_material_text - запрашивать или нет текст материала, если true – в поле $material[‘material_text’] содержится текст первой страницы материала.

Если есть материалы, удовлетворяющие условию - возвращает массив данных о материалах иначе false.

Формат элемента списка аналогичен формату, в котором возвращает данные функция material_data, за исключением отсутствия списка страниц материала ['pages'] - при необходимости его нужно запрашивать отдельно функцией material_pages($material_id).

Если запрашиваются свойства материала, то в массиве данных материала появляется новый элемент ‘properties’, содержащий массив свойств материала. Аналогично для рубрик.

Предвыборка материалов:

Для больших списков материалов целесообразно использовать функцию **materials_list_rlimit.**

function **materials_list_rlimit** ($params, $rubrics_list, $action = 'list', $limit_from = 0, $limit_len = 0, $mshort_list = '')

$params - массив, параметры поиска (индексы массива аналогичны параметрам функции materials_list)

$rubrics_list - список рубрик, по которым будут определяться права (если права на материал определять не нужно, то можно просто указать false)

$action – list возвращается либо список материалов,
count - возвращается кол-во результатов
preselect - возвращается список материалов в кратком формате
если к $action добавить norights, например, preselect_norights – права пользователей на материалы определяться не будут

$limit_from, $limit_to для списка соответственно лимиты возвращаемых записей
$mshort_list - список материалов в кратком формате созданный этой функцией
$join_pages - выполнять или нет объединение со страницами материалов
$join_properties - выполнять или нет объединение с атрибутами материалов
последние 2 параметра - для поиска

Пример использования функции для вывода списка материалов по страницам:

```
// Текущая страница
 if (!isset($page)) $page = 1;

// Кол-во результатов на странице
 $show_results = 20;

  // Не включать в список материалы подрубрик
  $include_subrubrics = false;
```

```
// Данные материалов в кратком формате
$preselected_materials = $materials->materials_list_rlimit (
array ('rubric_id' => $rubric_data['rubric_id'], 'include_subrubrics' => $include_subrubrics,
      'add_condition' => "material_publish='1'",", 'preselect_norights' );

// Количество материалов
$found  = ($preselected_materials) ? sizeof($preselected_materials) : 0;


// Данные материалов в полном формате (только те, которые будут выводиться)
$materials_list = ($found) ? $materials->materials_list_rlimit (
array ('rubric_id' =>  $rubric_data['rubric_id'], 'include_subrubrics' => false,
      'add_condition' => $include_subrubrics, 'join_pages' => 1,'join_properties'=>1), '', 'list_norights',
                  ($page-1)*$show_results, $show_results, $preselected_materials ) :
 array();


// Далее вывод материалов из полученного списка
```

Список рубрик материала:
material_rubrics($material_id)

возвращает список рубрик, которым принадлежит материал. В данные рубрики входят все поля таблицы rubrics и кроме того:

rubric_url, rubric_full_url,image (аналогично material_data)

Список ссылок к материалу:

material_links ($material_id) – возвращает список данных о ссылках к материалу (поля из таблицы material_link)

mlink_pos – порядковый номер ссылки
mlink_name – название ссылки
mlink_url - ссылка

Список файлов к материалу

material_accom_files ($material_id) – поля из таблицы material_accom_file, количество файлов к материалу - $material_data[‘material_accom_files’]

Список картинок фотогалереи к материалу

material_accom_images ($material_id) – поля из таблицы material_accom_image, кроме того

mimage_thumb – имя файла уменьшенной копии картинки,
mimage_thumb_url – ссылка на уменьшенную копию картинки,
mimage_thumb_full_url – полная ссылка

mimage_thumb_width, mimage_thumb_height,mimage_thumb_size – ширина, высота, размер в байтах

image – список изображений картинки (файлов изображений может быть больше 1), формат аналогичный $material_data[‘image’]

mimage_name – имя картинки, mimage_comment - комментарий

количество изображений в фотогалерее материала - $material_data[‘material_accom_images’]

Экспорт материалов в формат RSS

export_rss2 ($rubric_id, $top_materials = 20,$date = '')

$rubric_id – рубрика, материалы которой экспортировать
$top_materials – количество последних материалов по дате в экспортном файле
$date – дата, с которой выводить материалы

5.2 Класс Rubrics

Класс предназначен для управления рубриками на сайте

Данные рубрики:
function rubric_data ($rubric_id, $determ_user_rights = false)

$determ_user_rights – определять права пользователя на рубрику (false)

возвращает поля из таблицы rubrics, кроме того

image (аналочино $material_data), rubric_url,
subrubrics_amount – количество подрубрик

Список дочерних рубрик:
```
childs2table($rubric_id = 1, $order = ‘’,
                $delta_level = 0, $except = true,
                $add_condition = '',
                $count_materials = false,
                $retr_relate_rubric_data = false,
                $retr_owner_info = false, $retr_rubric_type = false,
              $determ_user_rights = false)
```

$rubric_id - id родительской рубрики, по умолчанию 1 (все рубрики сайта)
$order – порядок сортировки, варианты 'L_index' (по порядку нахождения в дереве) и level (по уровню). По умолчанию L_index
$delta_level - на какую глубину (уровней) узнавать дочерние рубрики, если 0 - узнаем все дочерние рубрики. (0)
$except - исключить или нет из списка рубрику с rubric_id = ‘$rubric_id’ (true)
$add_condition - дополнительные условия для выборки

$count_materials - подсчет материалов входящих в рубрику, поле ‘material_amount’ (false)
$retr_relate_rubric_data – запрос информации о рубрике, от который зависит рубрика списка (false)
$retr_owner_info - запрашивать или нет данные хозяина рубрики (false)
$retr_rubric_type – запрос информации о типе рубрики (false)
$determ_user_rights – определять права пользователя на рубрики (false)

возвращает массив данных о рубриках или false

возвращает поля из таблицы rubrics, кроме того

image (аналочино $material_data), rubric_url, subrubrics_amount – количество подрубрик рубрики

Список родительских рубрик:
parents2table($rubric_id, $delta_level=0, $except = true,
$add_condition = '', $determ_user_rights = false)

$rubric_id - id дочерней рубрики
$delta_level - на какую глубину (уровней) узнавать родительские рубрики, если 0 - узнаем все родительские рубрики. (0)
$except - исключить или нет из списка рубрику с rubric_id = ‘$rubric_id’ (true)
$add_condition - дополнительные условия для выборки
$determ_user_rights – определять права пользователя на рубрики (false)

5.3. Класс Highlights

Класс используется для анонсирования материалов на первой или другой странице сайта. Для анонсирования нужно создать группу анонсов (в меню бэкофиса “Анонсы материалов” -> “Добавить группу анонсов”). При занесении материалов в анонсы в списке выводятся материалы рубрики, определенной в поле “Исходная рубрика”. При сохранении перегенерируется рубрика, указанная в поле “Целевая рубрика”.

Список хайлайтов
highlights_list ($highlight_group_id = '', $retr_properties = '')

$highlight_group_id может быть числом (id нужной группы анонсов), либо кодом группы анонсов. Код группы анонсов может быть один, либо список через запятую, например:
$anonses = $highlights->highlights_list (‘main_anonses,top_anonses’);

Если $highlight_group_id – код группы, список анонсов возвращается в формате списка материалов ($materials->materials_list …)

5.4. Класс Search

Класс используется для поиска по материалам и модулям, которые поддерживают интерфейс search.

Поиск
do_search ($search_str, $rubric_id = 1)

поиск. $search_str – строка поиска, которая вводится в форме для поиска. $rubric_id – скаляр или массив id рубрик, в которых производить поиск (по умолчанию $rubric_id = 1 – поиск ведется по всем рубрикам). Если в выбранной рубрике помещен модуль, то вызывается (при наличии) метод search этого модуля. Для того чтобы по модулю производился поиск необходимо отметить в параметрах модуля “Возможен поиск по модулю”. Метод search модуля должен возвращать список массивов вида:

```
array ('res_id'  => …  // id записи в таблице модуля
        'res_date' =>   … // дата записи
        'res_class' => … // класс модуля
        'res_rubric_id' => $rubric_id[0] // id рубрики где расположен модуль. Используется для модулей, которые могут быть размещены на сайте в одном экземпляре, например, “Каталог ресурсов”. Для модулей, в таблицах которых есть информация, в какой рубрике они размещены (например, “Форумы”) эта запись не нужна. Значение этого поля необходимо для  создания ссылки на найденную запись (метод search_result_data класса search)

        'res_type' => … // Тип найденной записи. В модуле может быть несколько типов записей, например в каталоге ресурсов – ресурс и рубрика каталога, если тип записей один – это поле не требуется. Значение этого поля необходимо для запроса полной информации о найденной записи (метод search_result_data класса search)
);
```

В методе do_search результаты поиска по модулям объединяются и сортируются по дате.

search_result_data ($result_data)

Метод вызывает одноименный метод модуля, имя которого указано в $result_data[‘res_class’]. Метод **search_result_data** модуля возвращает массив с данными записи:

res_title – заголовок записи
res_anons – анонс
res_date – дата
res_url, res_full_url – ссылка

res_type – тип результата (например, “Материал”, “Отзыв”, “Сообщение”)
res_predicate – ‘в рубрике’, ‘на материал’, ‘в гостевой книге’

res_path – путь к результату, массив записей с полями ‘path_name’, ‘path_url’ и ‘path_full_url’. Для материалов – это рубрика материала + родительские рубрики. Для отзыва то же самое + материал, к которому относится отзыв. Для сообщения в гостевой книге – название гостевой книги + рубрика, где размещена гостевая книга + родительские рубрики.

5.5 Класс Templates

Обновление шаблона
update_template ($template_id, $template_data)

метод используется при обновлении шаблона, например при генерации меню в обработчике

```
$templates->update_template(56,
                array ('template_text' => addslashes($tpl->getOutputContent())));
```

В примере обновляется шаблон с id=56, текст шаблона – результат работы шаблонного “движка”

Занесение (или обновление версии шаблона)
function replace_template_version ($template_code, $template_version,
$template_text)

$template_code - код шаблона
$template_version – код версии шаблона
$tenplate_text – текст версии шаблона

Версии шаблонов используются для создания изменяемых элементов страниц, например меню, которое изменяется в зависимости от рубрики. Например, на сайте есть меню, в котором на первой странице сайта все пункты оформлены одинаково, а на страницах рубрик – пункт текущей рубрики выделен. Для этого в обработчике, генерирующем меню, обновляется шаблон-инклуд меню (‘menu’) и генерируются версии шаблона меню

```
$templates->replace_template_version ('menu',
                $rubric['rubric_dir'],$tpl->getOutputContent());
```

Для каждой рубрики 1 уровня создается версия меню, с выделенным пунктом текущей рубрики. Затем в обработчике материала или индекса рубрики присваивается версия шаблона (метод assignInclude класса TemplatePower).

5.6 Класс Polls (Голосования)

Каждое голосование относится к профилю. В параметрах профиля задается, как будет выводиться голосование. Форма голосования может выводиться в двух режимах:

1. Через шаблоны – задаются шаблон формы голосования и шаблон результатов голосования.
2. Через обработчик – задается обработчик, который выводит форму голосования. Этот режим используется если форма голосования нестандартная и не подходит

под стандартный обработчик, который наполняет шаблоны формы голосования и результатов голосования.

Рис. 5.1. Редактирование профиля голосования

Для вывода формы голосования в шаблон страницы нужно вставить следующий код:

```
<?
  require_once ('common.inc.php');
  ins_class ('polls');

  $polls->show_poll('Индекс сайта');
?>
```

Этот код будет отображать форму с голосованием, относящимся к профилю “Индекс сайта”

Пример шаблона формы голосования:

```
<table>
<tr>
        <td colspan="2"><strong>{poll_question}</strong></td>
</tr>


<!-- START BLOCK : poll_answer -->
<tr>
        <td width="2"><input {input_data}></td>
        <td width=100%> {poll_answer_text}</td>
</tr>
<!-- END BLOCK : poll_answer -->


<tr>
 <td colspan="2">
  <input type="submit" value="Голосовать">
 </td>
</tr>

</table>
```

Форма создается автоматически. На каждый вопрос создается блок poll_answer.

Пример шаблона результатов голосования:

```
<table>
<tr>
        <td><strong>{poll_question}</strong></td>
</tr>

<!-- START BLOCK : poll_answer -->

<tr>
 <td>{poll_answer_text}<br>

 ({answer_percent}%)
 <IMG SRC="/i/percent.gif" WIDTH={graph_width} HEIGHT=7 BORDER="1">
 </td>
</tr>

<!-- END BLOCK : poll_answer -->


<tr>
 <td>{add_str}</td>
</tr>


</table>
```

На каждый ответ выводится количество процентов ответов за этот вопрос и график.

5.7 Класс Forums (Форумы)

Класс предназначен для работы с древовидными форумами. Форумы могут быть организованы в категории.

Список форумов
forums_list ($forum_parent_id = '', $rubric_id = '',$add_condition = '')

$forum_parent_id - id категории форумов (возвращаются форумы из этой категории)
$rubric_id - id рубрики, в которой размещен форум
$add_condition - дополнительное условие выборки

Данные форума
forum_data ($forum_id)

$forum_id - id форума или путь к форуму

Количество сообщений в форуме
count_forum_messages ($forum_id = '', $add_condition = '')

$forum_id - id форума
$add_condition - дополнительное условие выборки

Сообщения из форумов
forum_messages ($forum_id = '', $thread_id = '',$add_condition = '',$order = '')

$forum_id - id форума

$thread_id - id ветки форума
$add_condition - дополнительное условие выборки
$order – сортировка

Данные сообщения форума
forum_message_data ($mes_id)

$mes_id - id сообщения

5.8 Функции

Файл include/func/db/common_db.inc.php

Файл подключается в common.inc.php, в этом файле производится подключение к базе данных. Ресурсный идентификатор соединения (link identifier) записывается в переменную $connect, в дальнейшем при запросах по умолчанию используется это соединение. При необходимости можно открывать другое соединение и указывать его при вызове функции запроса к базе данных.

resource **run_sql** ($sql_query, $sql_connect = 0)

выполняет sql-запрос содержащейся в строке $sql_query, если не указывается идентификатор соединения $sql_connect, то используется глобальное соединение $connect (аналогично для всех функций работы с БД). Возвращает идентификатор ресурса.

array **sql2table** ($sql_query,$sql_connect = 0)

Возвращает двумерный массив с результатами запроса в случае если запрос возвратил данные или false если данных нет.

К примеру, вывод списка записей:

```
foreach (sql2table (“SELECT * FROM entries”) as $entry)
{
  print “<br> - “.$entry[‘entry_name’]
}
```

mixed (array или scalar) **retr_sql** ($sql_query,$sql_connect = 0)

Возвращает ассоциативный массив с результатом запроса, если полей в запросе несколько, или скалярное значение, если поле в запросе одно.

К примеру, считывание информации о записи:

```
$entry_data = retr_sql (“SELECT * FROM entries WHERE entry_id = ‘1’”);
```

int **insert_sql** ($sql_query,$sql_connect = 0)

Возвращает id занесенной записи.
Пример: $entry_id = insert_sql (“INSERT INTO entries (entry_name) VALUES (‘some_name’)”);

Файл include/func/common_func.inc.php (подключается в common.inc.php)

void **ins_class** (имя класса,[имя класса])

создает экземпляр класса (классов), параметры 1 или несколько имен классов

string **str_to_upper**($Str)
переводит строку в верхний регистр. Дублирует стандартную функцию strtoupper(), нужна на хостинге, где strtoupper() работает неправильно.

array **rubric_childs** ($rubric_data, $rubrics_list, $delta = 0, $include_self = false)

возвращает список дочерних рубрик рубрики.

$rubric_data – данные рубрики
$rubrics_list – список рубрик
$delta - на какую глубину (уровней) узнавать дочерние рубрики,
если 0 - узнаем все дочерние рубрики. (0)
$include_self – включать в возвращаемый список саму рубрику или нет

из массива $rubric_data используются элементы ‘level’,’L_index’,’R_index’. Если в массиве $rubric_data определен только level – возвращается список рубрик с уровнем больше level – используется, напримерm если нужен список рубрик первого уровня.

array **rubric_parents** ($rubric_data, $rubrics_list, $delta = 0, $include_self = false)

аналогично функции rubric_childs.

Файл date2str.inc.php

string **date2str** ($cur_date,$part="ALL")

возвращает дату в формате “день месяц (по – русски) год”
если $part !- ‘ALL’ год не выводится

В func/file/common_file.inc.php

void **check_directory_exists** ($dirname)

$dirname – полный путь к директории, например ‘/users/home/222-www/docs/images/2/’;

Проверяет наличие указанной директории, если ее нет – создается.

void **dump_material_file** ($material_data,$rubric_data,$text, $page = 1, $ext = '')

Функция используется в обработчике страницы материала рубрики.

$material_data – данные материала
$rubric_data – данные рубрики
$text – текст (html произведенный шаблонным движком), записываемый в файл материала
$page – номер страницы, по умолчанию 1. Если номер страницы = 1, то материал записывается в файл вида site.ru/rubric/material.html, иначе site.ru/rubric/material,2.html (2 – номер страницы)

$ext – дополнительное расширение, например ‘image' (по умолчанию равно ‘’), если в в файл записывается картинка фотогалереи материала. Если указано дополнительное расширения то файл материала имеет вид site.ru/rubric/material,image,2.html (для второй страницы). Разделение запятыми нужно для того чтобы при удалении материала удалялись все файлы, записанные для этого материала.

void **dump_rubric_file ($rubric_data, $text, $page = 1)**

аналогично dump_material_file

6. Шаблонные директивы и методы класса “Шаблон”

6.1. Директивы

{переменная} – переменная, назначается функцией assign.

<!—START BLOCK : имя блока à
<!—END BLOCK : имя блока à

динамический блок, содержание блока не показывается до тех пор пока не будет вызвана функция newBlock (‘имя блока’)

<!—INCLUDE BLOCK : код шаблона – инклуда à

включение шаблона в шаблон страницы. В шаблон-инклуд другие инклуды включать нельзя. Если для инклуда определена версия функцией assignInclude – включается версия инклуда, иначе сам инлуд.

<!—TITLE à , <!—META à - title и META-тэги для страницы. Информация о том, какие META-тэги ставятся на страницу определяется функцией assignPlacePath.

6.2. Методы класса “шаблон”

assign (‘имя переменной’,значение)

Присваивает значение переменной шаблона. Значение может быть переменной или массивом. Если значение – массив, доступ к элементам массива в шаблоне производится так {имя переменнной.ключ_массива}. Если переменной шаблона не присвоить значение – оно будет равно ‘’ (пустая строка).

Значение присваивается для текущего блока. Т.е. если присвоили значение переменной rubric_name в корневом блоке и если переменная rubric_name еще используется в динамическом блоке - ее значение будет не определено, его нужно назначать еще раз.

assignGlobal (‘имя переменной’,значение)

назначает значение переменной для всех блоков

assignInclude ('код инклуда’,’версия инклуда’)

назначает версию для инклуда, вызывается до метода prepare().

assignPlacePath (‘путь к META-данным страницы’)

В обработчике, генерирующем материал функция вызывается с параметром 'material/'.$material_data['material_id'], в обработчике, генерирующем индекс рубрики с параметром 'rubric/'.$rubric_data['rubric_id']. Вызывается до метода prepare().

getOutputContent()

возвращает строку со сгенерированной страницей
gotoBlock (‘имя блока’)

делает блок текущим. Используется если было создано несколько динамических блоков функцией newBlock, а потом нужно назначить значение для переменной в корневом блоке. Для этого нужно вызвать функцию с параметров gotoBlock (‘_ROOT’).

newBlock (‘имя блока’)

Создает новый блок и делает его текущим. Если в шаблоне блока с таким именем нет, то ничего не создается.

prepare()

“Парсит” шаблон, выполняется после создания объекта шаблона и функций assignInclude и assignPlacePath. Без вызова этой функции объект работать не будет.

printToScreen()

выводит созданную страницу на экран

FAQ

1. При заходе в рубрику показывает ошибку 403 Forbidden

Такое сообщение показывается, если у рубрики не определена индексная страница и в настройках Apache не установлена опция Option Indexes. Выберите нужную страницу индекса рубрики в параметрах рубрики.

2. Как отлаживать обработчики.

См. пункт 3.4. “Создание обработчиков”. Функции print, print_r , echo выводят текст в стандартный поток вывода. Когда обработчик выполняется динамически можно просто вставить в нужное место обработчика вывод отладочного сообщения. Если обработчик используется при перегенерации материалов / индексов рубрик после изменения обработчика на странице редактирования обработчика снять отметку с чекбокса “Закрывать отладочное окно” и отладочные сообщения останутся во всплывающем окне.