

SAPID CMF

платформа разработки контент-зависимых веб-приложений

Руководство разработчика

Содержание

Содержание	1
Предисловие	6
Введение в платформу	7
Что такое SAPID CMF	7
Почему именно SAPID CMF	7
История платформы	7
Базовая концепция	9
Организация данных в платформе	9
Объекты информации	9
Документы	9
Записи	10
Файлы	11
Объекты организации информации	11
Структура	11
Переменные окружения	11
Шаблоны	13
Наборы полей	13
Сценарии функциональности	14
Списки записей	14
Пользователи	14
Отчеты	15
Конфигурация	15
Архитектура платформы	17
Архитектура БД	17
База данных платформы	17
Вспомогательные таблицы	17
Компонентная модель	18

Установка и обновление платформы	20
Установка платформы на сервер хостинг-провайдера	20
Установка платформы на свой локальный компьютер	20
Программа установки платформы	20
Обновление платформы	23
Создание информационной системы	24
Создание ресурса	24
Создание структуры	24
Создание шаблонов	25
Создание функциональных сценариев	26
Администрирование проекта	28
Административная панель	28
Раздел Структура	29
Раздел Сервисы	33
Пользователи	38
Отчеты	39
Конфигурация	40
Администрирование непосредственно на сайте	41
Отладка информационной системы	44
Адаптация платформы	45
Создание plugin	45
Определенные в системе события	46
Программный код адаптации	47
Структура \$env	47
Функции общего назначения	49
Использование стороннего визуального редактора (WYSIWYG)	49
Использование FCKEditor	49
Адаптация административной панели	51

Разработка тем оформления административного интерфейса	53
Создание административных приложений	54
Состав приложения	54
Создание приложения.....	55
Создание приложений со списками	57
Создание интерфейсов с управляемыми списками	59
API платформы SAPID CMF	65
ADO API (Интерфейс управления БД)	65
All()	65
One()	65
Update().....	65
lastNumRows()	65
lastAffected().....	66
lastInsertId().....	66
StartTransaction()/CompleteTransaction().....	66
prepare().....	66
DM API (интерфейс управления документами).....	67
Определение документа	67
Атрибуты документа	67
addSite()	68
add()	68
update()	69
delete()	69
updateData().....	69
copy().....	70
get()	70
getList().....	70
getByCondition()	71

getData()	71
getSiteID()	72
identify()	72
setPermission()	72
backup()	73
restoreBackup()	73
BackupList()	73
API (интерфейс управления записями)	74
Определение записи	74
Атрибуты записи	74
add ()	75
update()	75
delete()	76
updateData()	76
get()	76
getList()	77
getByCondition()	77
getData()	77
copy()	78
copyList()	78
addIndex()	78
deleteIndex()	78
setPermission()	79
backup()	79
UM API (интерфейс управления пользователями)	79
get ()	80
update ()	80
delete ()	80

getData ().....	81
updateData ().....	81
deleteData ().....	81
getByLogin()	82
RD API (интерфейс адресов сервисов).....	82
clean ().....	83
set()	83
create ()	83
Справочник разработчика проекта	84
Переменные среды	84
Запросы содержания (QC).....	85
Выражения	86
CMS – приложения	87
get_infochannel()	87
pagination().....	89
get_tree()	90
get_track().....	91
get_search ().....	92
get_ad ()	92
Приложения	94
Поддержка.....	94

Предисловие

Данное руководство написано для людей, которые желают быстро и качественно создавать информационные системы на базе Веб и контент-зависимые приложения с помощью платформы SAPID CMF.

В документе приведены общие сведения о платформе и ее устройстве, принципах разработки информационных систем на базе SAPID CMF, а также справочник API платформы для адаптации платформы под специфические локальные нужды (см. раздел «Адаптация»)

Ожидается, что разработчики уже знакомы с основами HTML, XML, PHP, SQL, а также имеют опыт работы с FTP-менеджером. Прежде чем начать работу с данным руководством, желательно ознакомиться с документом «Спецификация XML Sapiens 2.0». Также приветствуется опыт работы в парадигме MVC (Модель-Отображение-Контроллер).

SAPID CMF – это бесплатное программное обеспечение. Вам нет необходимости платить за него, и вы можете использовать его как пожелаете. Это проект Open Source, что значит, что вы имеете полный доступ к его исходному коду. Последнюю версию платформы вы можете скачать на сайте <http://sapidcmf.sourceforge.net>

Введение в платформу

Что такое SAPID CMF

SAPID CMF - платформа разработки контент-зависимых приложений с открытым кодом. Это полноценная система управления контентом (CMS), которую можно использовать для создания сайтов и это фреймворк, позволяющий неограниченное развитие CMS в соответствии с вашими нуждами.

SAPID CMF содержит набор библиотек, классов и набор run-time средств для программистов, создающих веб-приложения. Компонентная модель SAPID CMF опирается на шаблон MVC (Модель-Отображение-Контроллер). SAPID CMF организует данные, их представление и функциональность на базе концепции XML Sapiens.

Почему именно SAPID CMF

- Гибкая модель шаблонирования структур данных, их представления и функциональности на базе языка разметки XML Sapiens 2.0, обеспечивающая широкие возможности повторного использования кода (reusing)
- Оптимизировано для создания приложений эпохи Веб 2.0
- AJAX-базированные пользовательские интерфейсы и набор готовых виджетов (дерево, грид, формы, панели и прочее)
- Архитектура Model-View-Controller (MVC)
- Способность к развитию (плагины, темы оформления, аспект ориентированная событийная модель)
- Совместимость с PHP4 и PHP5
- Диспетчер запросов, с возможностью создавать и использовать человеко-понятные ссылки (ЧПУ)
- Верификация данных на уровне настраиваемых запросов (QC)
- Быстрое и гибкое шаблонирование (XML Sapiens)
- Гибкие списки контроля доступа (Access Control Lists)

История платформы

Декларативный язык XML Sapiens был разработан программистом Red Graphic Systems Дмитрием Шейко в 2003 году в поиске оптимальной модели разделения данных, их представления и функциональности для системы управления корпоративным контентом Site Sapiens (www.sitesapiens.ru). Осенью 2004 года компания Red Graphic Systems выступила инициативой Open Source и опубликовала спецификацию XML Sapiens. Для ознакомления с концепцией XML Sapiens Максимом Барышниковым была разработана

система управления сайтами SAPID. CMS SAPID была проста в установке, не требовала наличия БД и позволяла администрирование в INLINE режиме (как вижу так и редактирую), что обеспечило ей популярность. Однако SAPID CMS не проектировалась как система для разработки серьезных веб-проектов, лишь для демонстрации возможностей XML Sapiens. Спустя три года после выхода первой версии SAPID CMS появился фреймворк SAPID CMF. Фреймворк предназначен для скорой разработки контент-зависимых веб-приложений, создания информационных пространств на базе веб. Его основное отличие перед SAPID CMS – способность к неограниченному развитию. Достижение этой цели стало возможным благодаря использованию таких технологий как аспект ориентированная событийная модель (AOSD), плагинов и XML Sapiens 2.0. SAPID CMF содержит легковесный AJAX-фреймворк, что также облегчает наращивание пользовательских интерфейсов и сервисов.

Базовая концепция

Организация данных в платформе

Информационное веб-пространство – это совокупность всех сайтов проекта, управляемых централизованно, посредством единой системы. Все данные информационного веб-пространства на базе платформы SAPID CMF могут быть представлены в виде информационных объектов 3-х типов: документов, записей и файлов.

Любому из этих объектов могут быть назначены атрибуты, содержание и права доступа. Документы и записи могут включать вложенный список записей.

Для организации документов информационного веб-пространства служит иерархическая структура.

Шаблоны документов служат для задания оформления, а также для задания структуры содержания документа. Шаблоны могут включать друг друга.

PHP-скрипты, как правило, включаются из шаблонов или DDC и служат для реализации контроллеров страниц.

Наборы полей предназначены для задания структуры записей и профилей описания файлов и пользователей.

Сценарии функциональности (DDC) применяются для создания функциональных блоков. Подробнее о DDC можно узнать из документа «Спецификация XML Sapiens 2.0» (www.xmlsapiens.org).

Объекты информации

Документы

Документ – это информационный объект иерархического дерева структуры ресурса. Документ логически может быть разделом структуры сайта. Документ может содержать вложенный список записей.

Для получения списка документов, формирования навигационных меню используется DDC (см. “Спецификация XML Sapiens 2.0”) и CMS-приложение `get_tree()`.

Атрибуты документа

Название – определяет документ в структуре;

Шаблон – шаблон оформления документа;

Активность документа – определяет, будет ли отображаться документ в навигации;

Теги – ключевые слова для задания ассоциативных зависимостей между документами.

Содержание

Содержание документа определяется запросами содержания, представленными в шаблоне документа и подключаемых им шаблонах. Таким образом, если назначить документу новый шаблон, документ будет

Содержание

Содержание записи определяется запросами содержания, представленными в наборе полей. Файлы наборов полей расположены в папке /views/delivery/fieldsets/. Запрос содержания в наборе полей определяется конструкцией языка XML Sapiens `<sapi:apply name="qc.name.value" />`. Заголовок запроса определяет атрибут title элемента sapi:apply, а тип запроса, соответственно, атрибут type.

Типы запросов содержания определены в одноименных файлах, расположенных в папке /views/default/qcs/.

Права доступа

По умолчанию к новой записи пользователи административных групп SAPID CMF имеют доступ на чтение и запись, неавторизованные пользователи – доступ на чтение. Однако есть возможность установить определенные права доступа для различных групп пользователей.

Для управления правами доступа к записи служит административный пользовательский интерфейс управления свойствами записи, закладка «Безопасность». Для программного управления правами доступа записи в ходе адаптации платформы используется класс Rec (см. раздел RM API).

Файлы

Файл – это информационный объект виртуальной файловой системы. Файл может быть добавлен в платформу посредством административного интерфейса или путем непосредственной записи в одну из вложенных папок /views/delivery/files/.

Объекты организации информации

Структура

Структура информационной сети представлена виде иерархического дерева и определяет базовую зависимость документов. Первый уровень структуры представлен ресурсами информационного пространства. Ими могут быть сайты или же языковые версии сайтов. Каждый ресурс содержит собственное дерево. Прежде чем формировать структуру ресурса, следует определиться с тем, какая информация будет представлена как документы дерева, какая – как записи линейных списков.

Переменные окружения

Вы можете получить доступ к переменным окружения в шаблонах, в наборах полей, в сценариях функциональности или в шаблонах запросов посредством следующего синтаксиса:

document_template – шаблон оформления открытого документа (например, gallery.tpl);

document_id - идентификатор открытого документа (например, 9);

document_data_x – содержание, где x – имя поля содержания;

record_data_x – содержимые записи, где x – имя поля содержания.

Шаблоны

Шаблон определяет, что будет собой представлять страница сайта, которой он назначен. Шаблон задает оформление (HTML, RSS и т.д), он использует язык XML Sapiens для определения структуры содержания (QC), для определения функциональности (DDC), для отображения переменных среды и включения других шаблонов. Удачная архитектура подразумевает наличие шаблонов для всех повторяющихся блоков страниц сайтов информационной системы, включаемых в шаблонах этих страниц. Таким образом, если вы что-либо изменяете в шаблоне строки Copyright, это изменение отразится во всех страницах с шаблонами, включающими данный.

Файлы шаблонов расположены в папке /views/delivery/templates/.

Наборы полей

Набор полей служит для определения структуры записи. Запросы содержания в файле набора полей указывают, какого рода содержание будет запрошено для определенной записи в административном интерфейсе и, соответственно, какое содержание записи будет доступно на страницах сайта. Наборы полей могут быть определены для различных состояний записи. Так, одна и та же запись может быть отображена в общем списке на сайте в одной конфигурации, и при выборе страницы этой записи в другой конфигурации. Вывод записей обуславливает сценарий функциональности (DDC). В DDC можно описать оформление содержания записи указанием переменных содержания (&this.this.имя_поля_QC.value;) или же запросом набора полей для определенного состояния <sapi:include href="файл_набора_полей" parse="fieldset" state="состояние_A" />.

Набор полей допускает задание кода оформления содержания.

Пример набора полей:

```
<?xml version="1.0"?>
<sapi version="2.0" xmlns:sapi="http://www.xmlsapiens.org/spec/sapi.dtd">
  <sapi:fieldset type="default" state="admin" title="">
    <sapi:body>
      <sapi:apply name="qc.x1.value" type="type1" />
    </sapi:body>
  </sapi:fieldset>
  <sapi:fieldset type="default" state="delivery" title="">
    <sapi:body>
      <sapi:apply name="qc.x2.value" type="type2" />
    </sapi:body>
  </sapi:fieldset>
</sapi>
```

Файлы наборов полей расположены в папке `/views/delivery/fieldsets/`.

Сценарии функциональности

Сценарии функциональности (DDC) служат для описания поведения динамических блоков сайта, таких, как списки записей (ленты новостей, каталоги, галереи), голосования и т.д. С подробным описанием DDC можно ознакомиться в документе «Спецификация XML Sapiens 2.0». Включение тех или иных сценариев функциональности (DDC) описывается в шаблонах документов или в других DDC.

Для включения кода контроллера в шаблоне документа или в теле DDC используется конструкция `<sapi:include href="имя_скрипта" parse="php" />`.

Файлы сценариев функциональности расположены в папке `/views/delivery/ddcs/`.

Списки записей

Документ или запись может содержать список записей. Запись списка, в свою очередь, может содержать записи. Для отображения списков записей на сайтах используются DDC (См. документ «Спецификация XML Sapiens 2.0») и CMS-приложение `getinfochannel`. Для программного управления списками записей в ходе адаптации платформы используется библиотека `Rec`.

Пользователи

Пользователь платформы содержит следующие атрибуты:

Имя – идентификатор пользователя. Обычно совпадает с логином;

Логин – логин для авторизации пользователя;

Пароль – пароль для авторизации пользователя;

Название – имя пользователя в общем списке;

Email – email адрес пользователя;

Язык – язык интерфейс для данного пользователя;

Состояние активности – флаг состояния активности;

Профиль – набор полей для описания пользователя. Профиль определяет поля запросов содержания для формы описания пользователя. Файлы профилей пользователя расположены в папке `/views/delivery/templates/users/`.

Пользователи могут быть объединены в группы. В свою очередь, пользовательские группы также могут быть объединены в группы. Для программного управления пользователями в ходе адаптации платформы используется класс `User`.

Пользователю платформы может быть назначен доступ к информационным объектам. Пользовательской группе может быть предоставлен доступ к заданному административному интерфейсу в конфигурационном файле данного интерфейса. Например, `app/structure/app.db`.

Синтаксис:

```
<?xml version="1.0" encoding="utf-8" ?>
<treeitem titlekey="Structure" sortkey="1">
  <acl>
    <group name="developers" permissions="7" />
  </acl>
</treeitem>
```

Элемент ACL обозначает секцию доступа к интерфейсу указанному в элементе treeitem.

Элемент GROUP определяет доступ в данный интерфейс для заданной группы и содержит следующие атрибуты:

name – имя группы или пользователя;

permissions – права доступа двоичной системе (0-1 – нет доступа, 2-3 – доступ записи, 4-7 – доступ к чтению, 6-7 – доступ к чтению и записи).

Отчеты

Любые операции с API или действия в административном интерфейсе отражаются в отчете «Журнал событий в системе».

Конфигурация

Базовая конфигурация платформы назначается в файле /config/rc.conf.php.

HTTP_PATH – веб-адрес сайта по умолчанию;

SITEID – ID ресурса по умолчанию;

DEFAULT_LANGUAGE – язык интерфейса по умолчанию;

ROOT_PATH – полный корневой адрес на сервере;

ADMIN_EMAIL – email адрес для обратной связи по умолчанию;

VERSION – версия SAPID CMF;

CONFIGURED – дата конфигурации платформы;

DBPREFIX – префикс для таблиц базы данных;

ADMINAREA_PREFIX – виртуальная папка для административной панели;

CONTROLAREA_PREFIX – виртуальная папка AJAX-контроллеров платформы;

TOTALCACHING – включить/выключить кеширование страниц;

VENDORS – адрес папки со сторонними библиотеками;

DEBUG – режим отладки (0/1/2);

DOCCOPIESMAXNUMBER – хранимое общее число версий содержания документа/записи;

DOCCOPIESMAXNUMBERPERUSER – хранимое число версий содержания документа/записи для каждого пользователя;

BACKUPING - включение/выключение хранения версий содержания;

\$GLOBALS["DBConnect"] – данные БД;

SOAP_SERVER_URL – адрес внешнего SOAP-сервера;

FILESDATEFORMAT – формат даты в файлах по умолчанию;

INLINEMODE – наличие режима редактирования INLINE (значения on и off);

THEME – текущая тема оформления административного интерфейса;

PAGINATIONFRAMELENGTH – длина фрейма для пагинации.

DBTABLESSTRUCTSPATH – расположение скриптов для восстановления БД;

GMTOFFSET – смещение часового пояса по Гринвичу;

Дополнительная конфигурация вносится посредством административного интерфейса «Конфигурация» в реестр. Данные конфигурационного реестра доступны в шаблонах и DDC как переменные окружения &config_переменная.value; и, соответственно, доступны при адаптации (в plugins) как &env->Value["config_переменная"].

Архитектура платформы

Архитектура БД

База данных платформы

_config – конфигурационный реестр;
_doc_data – содержание документов;
_doc_data_bak – версии содержание документов;
_doc_structure – структура документов;
_log – протокол сеансов администрирования;
_permissions – группы доступа к информационным объектам;
_rec_data – содержание записей;
_rec_data_bak – версии содержания записей;
_rec_structure – структура записей;
_tagcloud_index – индексы тегов структуры;
_user_accounts – структура пользователей;
_user_profilesdata – данные описаний пользователей;
_user_settings – данные личных настроек интерфейса;
_whole_enums_data – данные перечислений.

Вспомогательные таблицы

_advertising – учетная информация по рекламным блокам;
_maillist – настройки почтовых рассылок;
_rec_rate – таблица рейтингование записей;
_thesaurus – тезаурус для содержания;

Компонентная модель

APP – бизнес-логика и контроллеры административных интерфейсов

CACHE – файлы кеша

CONFIG – конфигурация

DOCUMENTATION - документация

INSTALL – система развертывания проекта

LANGS – файлы языков интерфейса

LIBS – библиотеки

- CMSAPPS - CMS-приложения платформы

- MODEL

 - DBO – библиотеки для взаимодействия с абстрактными слоями БД

 - PHP4 – библиотеки модели для PHP4

 - PHP5– библиотеки модели для PHP5

- VIEW

 - PHP4 – библиотеки вида для PHP4

 - PROCESSOR – библиотеки процессора XML Sapiens 2.0

 - PHP5 – библиотеки вида для PHP5

 - PROCESSOR– библиотеки процессора XML Sapiens 2.0

PLUGINS – адаптация платформы

TESTS – unit tests

VENDORS – внешние библиотеки

- adodb_lite

- fckeditor

- graphic_snips

- jscalendar

- lastRSS

- mmplayers

simple_openid

simpletest

tinymce

yui

VIEWS – шаблоны вида и функциональности

DEFAULT – шаблоны для административной панели, тема оформления DEFAULT

CSS – таблицы стилей

DDCS – сценарии функциональности

FIELDSETS – шаблоны наборов полей

JS – сценарии JS

PIC - изображения

QCS – шаблоны запросов

TEMPLATES – шаблоны интерфейсов

DELIVERY - шаблоны сайтов

CSS – таблицы стилей

DDCS – сценарии функциональности

FIELDSETS – шаблоны наборов полей

JS – сценарии JS

FILES – архив файлов для загрузки с сайтов

IMG – архив изображений

QCS – шаблоны запросов

TEMPLATES – шаблоны интерфейсов

THUMBNAILS – иконки для изображений из IMG

WIDGETS – виджеты сайтов

Установка и обновление платформы

Для того чтобы установить SAPID CMF, разумеется, требуется дистрибутивный пакет платформы, который можно скачать на сайте <http://sapidcmf.sourceforge.net>.

Установка платформы на сервер хостинг-провайдера

Для того чтобы заставить исполняться PHP скрипты платформы потребуется веб-сервер. Вы можете заказать хостинг план для PHP4 или PHP5 у любого хостинг провайдера. Удостоверьтесь о наличии БД MySQL 4 или 5 версии и модуля REWRITE в случае сервера Apache в вашем хостинг плане. Вам также потребуется библиотека расширения PHP mbstring (содержит функции для обслуживания строк с многобайтовыми/UTF8 символами).

Получив от хостинг провайдера данные для FTP-доступа, воспользуйтесь Total Commander (www.ghisler.com) или любым FTP-клиентом для того, чтобы скопировать файлы дистрибутивного пакета SAPID CMF в папку html (имеется в виду корневая папка веб-проекта) на вашем сервере. После этого установите права доступа к папкам cache, cache/ddc, tmp, views/delivery/files, views/delivery/img, а также к файлу config/rc.conf.php – 777 (полный доступ). В Total Commander для этого следует выбрать папку или файл, а затем в меню программы Files/Change Attributes отметить все чекбоксы или же указать 777 в поле маски доступа.

Далее вам следует вызвать веб-браузер (например, FireFox) и указать в адресной строке имя вашего сайта (скажем, myownstartup.myhost.com). Вы должны увидеть окно программы установки SAPID CMF. О том, как работать с программой см. в разделе «Программа установки SAPID CMF».

Установка платформы на свой локальный компьютер

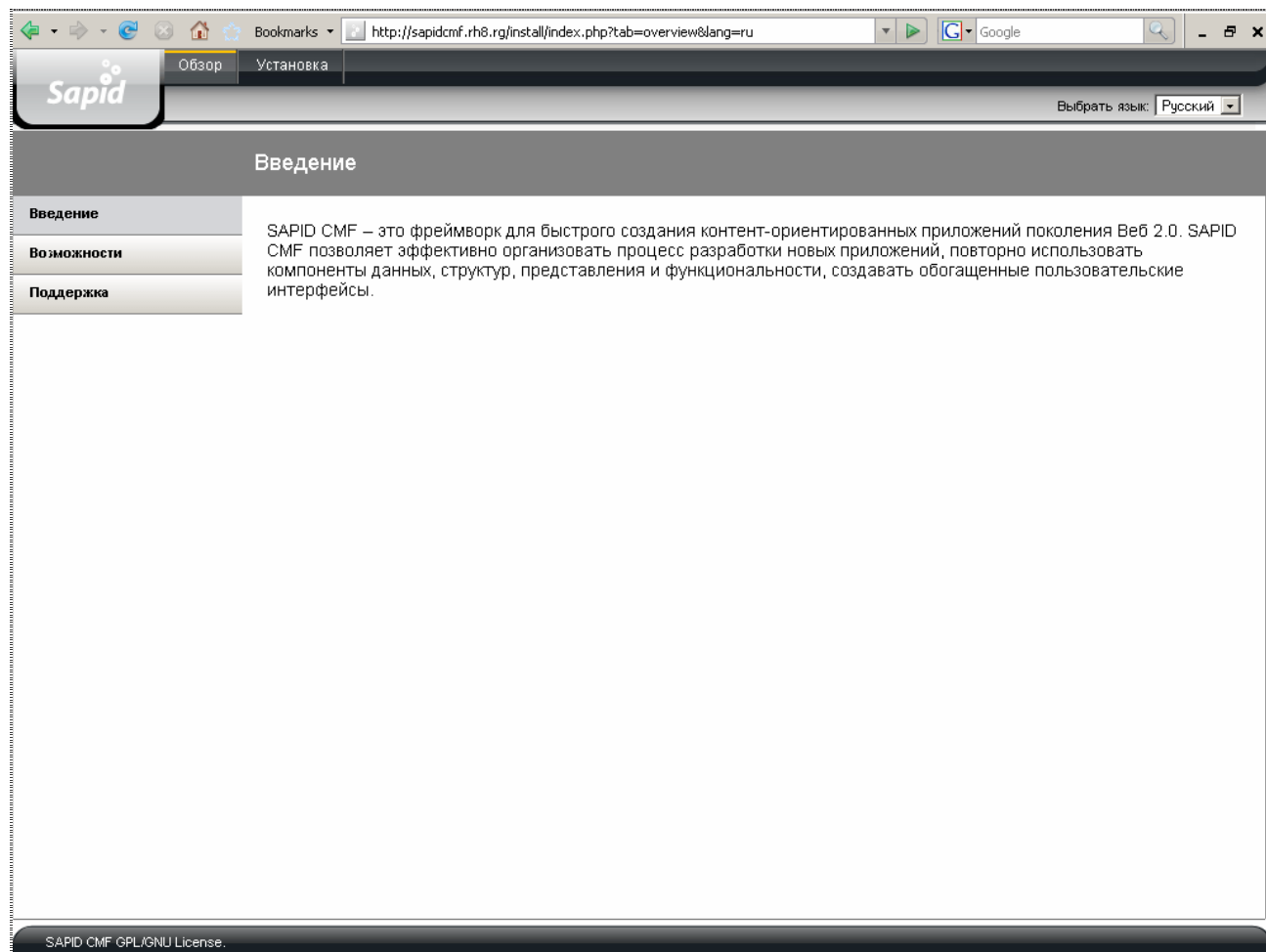
Если вы желаете установить SAPID CMF на ваш локальный компьютер, воспользуйтесь программой Денвер для автоматической настройки веб-сервера под ОС Windows. Скачать последнюю версию программы можно бесплатно по адресу <http://www.denwer.ru/dis/base/>

Денвер – это простейший способ установить собственный веб-сервер Apache. Вам достаточно лишь загрузить его, запустить и следовать его инструкциям на русском языке по мере установки сервера. В большинстве случаев – это будет лишь несколько вопросов, на каждый из которых можно смело отвечать «Да».

После установки сервера вы, скорее всего, обнаружите на своем компьютере дополнительный диск Z. На нем в папке home будут папки веб-проектов localhost, test1.ru и т.д. Вы можете использовать папку test1.ru (можно ее переименовать по своему усмотрению, скажем, в sapidcmf.ru). После этого следует перезапустить сервер. Теперь требуется скопировать в папку веб-проекта (test1.ru) файлы дистрибутива SAPID CMF. Осталось лишь вызвать ваш веб-браузер (например, Internet Explorer) и указать в адресной строке имя проекта (test1.ru). Вы должны увидеть окно программы установки SAPID CMF. О том, как работать с программой см. в разделе «Программа установки SAPID CMF».

Программа установки платформы

При первом запуске программы вы увидите экран приветствия.

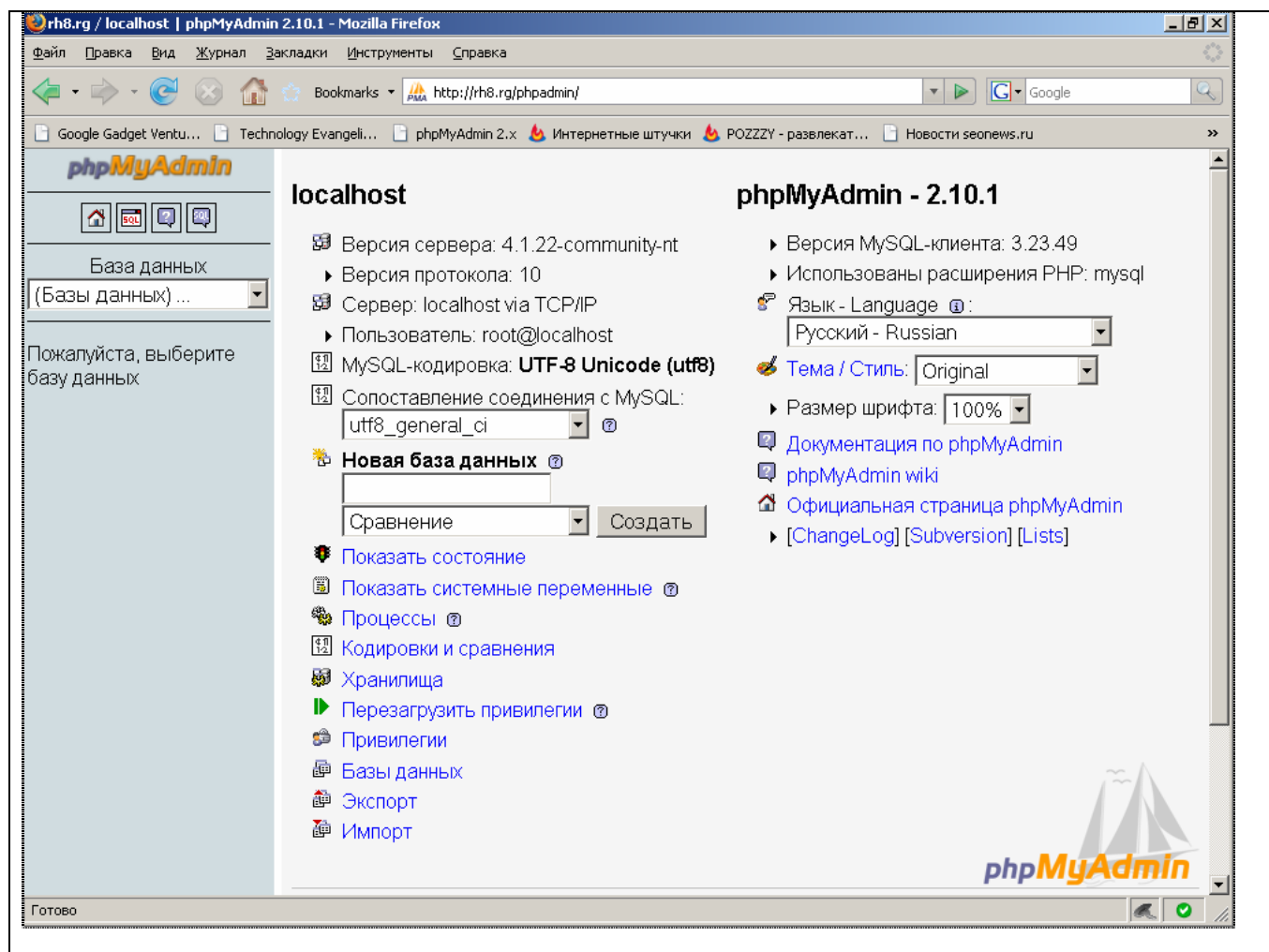


Для начала установки следует воспользоваться разделом верхнего горизонтального меню «Установка». Вам откроется страница «Добро пожаловать» где вы сможете нажать кнопку «Перейти к следующему шагу». Приложение расскажет вам о том, все ли из того, что требуется для работы SAPID CMF установлено на вашем сервере. Когда вы перейдете к следующему шагу (кнопка «Начать установку»), у вас будут запрошены данные для соединения с сервером базы данных.

Если на вашем сервере используется PHP 5, наверняка в списке возможных библиотек объектов БД появиться PDO. Это «родная» библиотека PHP для управления объектами БД и очевидно с ней ваш проект будет работать быстрее, нежели с популярным, но сторонним решением AdoDB Lite. Далее вам потребуется указать классический набор данных доступа к БД. В том случае если вы используете внешний сервер, эти данные вам должен предоставить хостинг провайдер. Если вы устанавливаете проект на своей компьютер под управлением ОС Windows – можете оставить все как есть. Однако, до того как вы перейдете к следующему шагу следует создать новую базу данных (например, с именем sapidcmf) и указать его в поле «имя базы данных».

Создание БД в PhpMyAdmin

В комплекте Денвером поставляется PhpMyAdmin (<http://test1.ru/phpmyadmin/>). Откройте его и укажите в поле «Новая база данных» sapidcmf, ниже в поле «Сравнение» - utf8_general_ci и нажмите рядом кнопку «Создать»



На следующем шаге вам будут заданы вопросы для конфигурации проекта под управлением SAPID CMF. Не забудьте выбрать свой язык административного пользовательского интерфейса, а также логин и пароль администратора проекта.

На завершающей стадии вам будет предложено удалить или переименовать папку install и перейти на стартовую страницу демонстрационного сайта или в административный режим.

Обновление платформы

Для обновления версии SAPID CMF вам следует загрузить архив с файлами обновления с сайта <http://sapidcmf.sourceforge.net> и распаковать их поверх файлов вашего веб-проекта. В архивах обновлений нет шаблонов и структур области доставки содержания, так что вы не сможете затереть обновлением вашу работу в проекте.

Если архив обновления содержит в корневой папке скрипт update.php, после того как файлы обновления будут перенесены на сервер, стартуйте это скрипт и следуйте его инструкциям.

Создание информационной системы

Создание ресурса

Прежде всего, вам следует перейти в административную панель платформы (<http://вашcat/admin>). Теперь можно нажать ссылку напротив верхней позиции («Информационное пространство») и ввести название ресурса.

Затем требуется навести курсор мыши на пиктограмму созданного в структуре ресурса. Вы найдете ID созданного ресурса во всплывающей подсказке (например, #1). Укажите его в файле конфигурации `config/rc.conf.php`

```
if(ereg("lang.site.com", $_SERVER["HTTP_HOST"])) {  
    define("HTTP_PATH", "http://lang.site.com/");  
    define("SITEID", 59);  
} else {  
    define("HTTP_PATH", "http://sapidcmf.rh8.rg/");  
    define("SITEID", 1);  
}
```

В данном примере, доменное имя `lang.site.com` соотносится с ресурсом с ID 59, доменное имя `sapidcmf.rh8.rg` с доменом с ID 1.

Создание структуры

Для создания структуры нового ресурса можно воспользоваться административным интерфейсом.

Также можно восстановить структуру из текстового файла. Для этого подготовьте файл в формате

```
@site:Имя нового ресурса  
Номер уровня вложенности;Название документа; переменная адреса;  
Номер уровня вложенности;Название документа; переменная адреса;  
...
```

или

```
@site:Имя нового ресурса  
Номер уровня вложенности;Название документа; переменная адреса; имя файла  
шаблона;  
Номер уровня вложенности;Название документа; переменная адреса; имя файла  
шаблона;  
...
```


Создание функциональных сценариев

Функциональные сценарии или DDC подробно описаны в документе «Спецификация XML Sapiens 2.0». В SAPID CMF файлы функциональных сценариев располагаются в папке /views/delivery/ddcs/, причем название файла ОБЯЗАТЕЛЬНО должно совпадать с именем DDC.

Вывод значений перечисления определяется как &this.this.переменная.value;, где this обозначают текущий DDC и перечисление. Соответственно переменные вложенного перечисления следует именовать как &this.this.this.переменная.value;.

Пример:

```
<?xml version="1.0"?>
<sapi version="2.0" xmlns:sapi="http://www.xmlsapiens.org/spec/sapi.dtd">
  <sapi:ddc name="test" title="ddc" ns="site1" category="navigation">
    <sapi:choose>
      <sapi:when exp="1">
        <sapi:for-each select="get_infochannel()" name="enum" title="Get
channel">
          <sapi:params>
            <sapi:param name="sources">4v*:link</sapi:param>
            <sapi:param name="limit">0,10</sapi:param>
            <sapi:param name="name">test</sapi:param>
          </sapi:params>
          <sapi:ifempty>Records was not found</sapi:ifempty>
          <sapi:fallback>
            infochannel CMS-application error
          </sapi:fallback>
          <sapi:choose>
            <sapi:when exp="1">
              <sapi:code>
                <b><a
href="&this.this.href.value;">&this.this.name.value;</a>
&this.this.channel_id.value;</b><br />
              </sapi:code>
            </sapi:when>
          </sapi:choose>
        </sapi:for-each>
      </sapi:when>
    </sapi:choose>
  </sapi:ddc>
</sapi>
```

При разметке функциональности для получения данных от платформы SAPID CMF используются CMS-приложения. См. раздел «CMS-приложения».

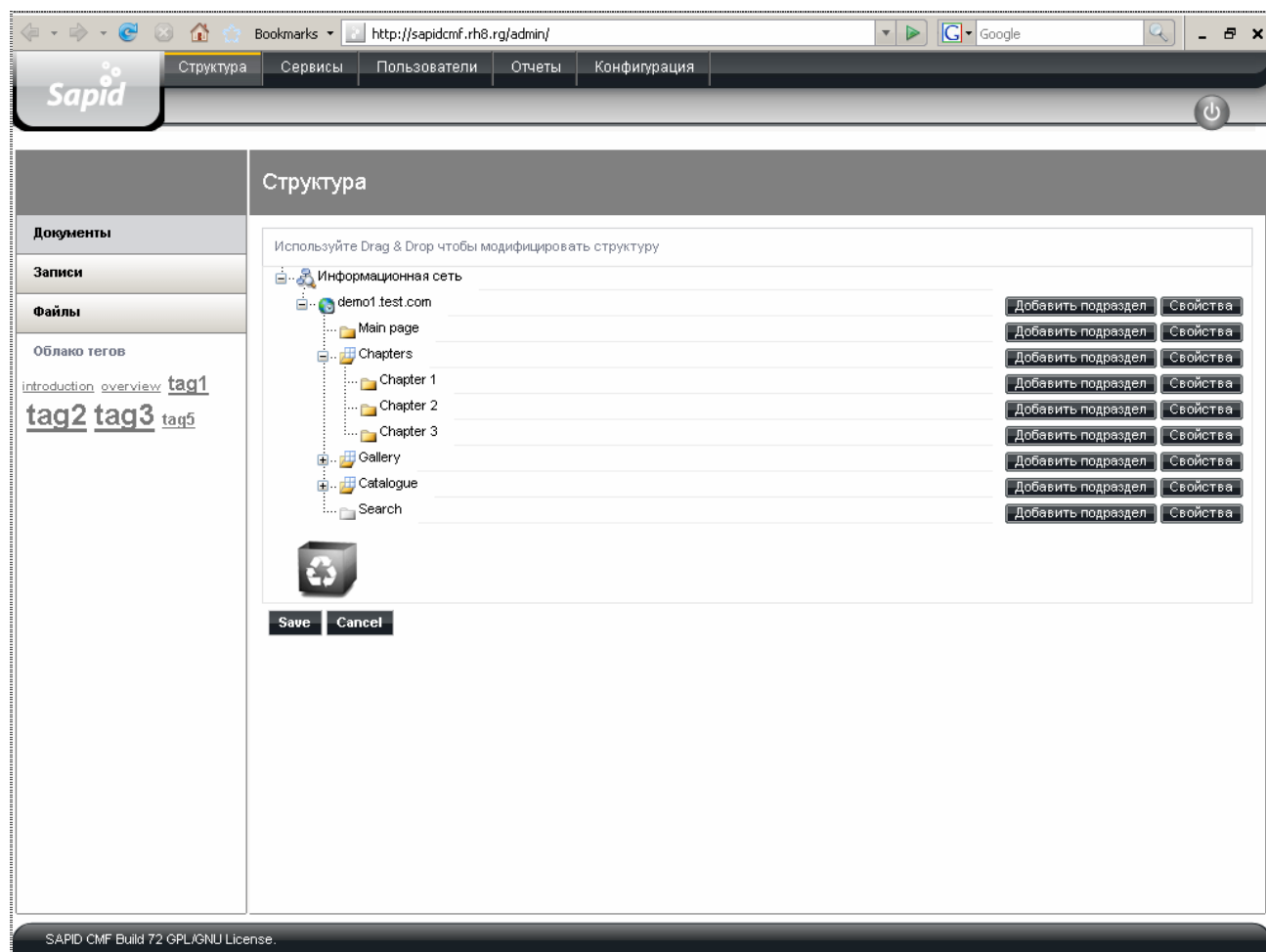
Администрирование проекта

Административная панель

Для управления проектом на базе SAPID CMF используется административная панель. Для того чтобы перейти в нее следует набрать в адресной строке вашего веб-браузера адрес http://имяпроекта/адрес_административной_панели. Адрес административной панели вы указывали при установке SAPID CMF (по умолчанию этот адрес admin). Таким образом, если адрес проекта test1.ru, то для перехода в административную панель следует набрать <http://test1.ru/admin/>

При входе в административную панель вам потребуется ввести логин и пароль администратора (вы указали эти данные при установке SAPID CMF). Если вы отметите чекбок «Запомнить» следующий раз при авторизации вам не обязательно будет вводить логин и пароль. Так что не стоит использовать эту опцию на чужом компьютере.

После того как вы авторизуетесь, будет отображен пользовательский интерфейс административной панели.



В верхнем горизонтальном меню представлены разделы: Структура, Сервисы, Пользователи, Отчеты, Конфигурация. Впрочем, вы можете расширить это список по своему усмотрению.

В левом вертикальном меню представлены доступные приложения текущего раздела. В правом верхнем углу расположена кнопка «Выход».

Раздел Структура

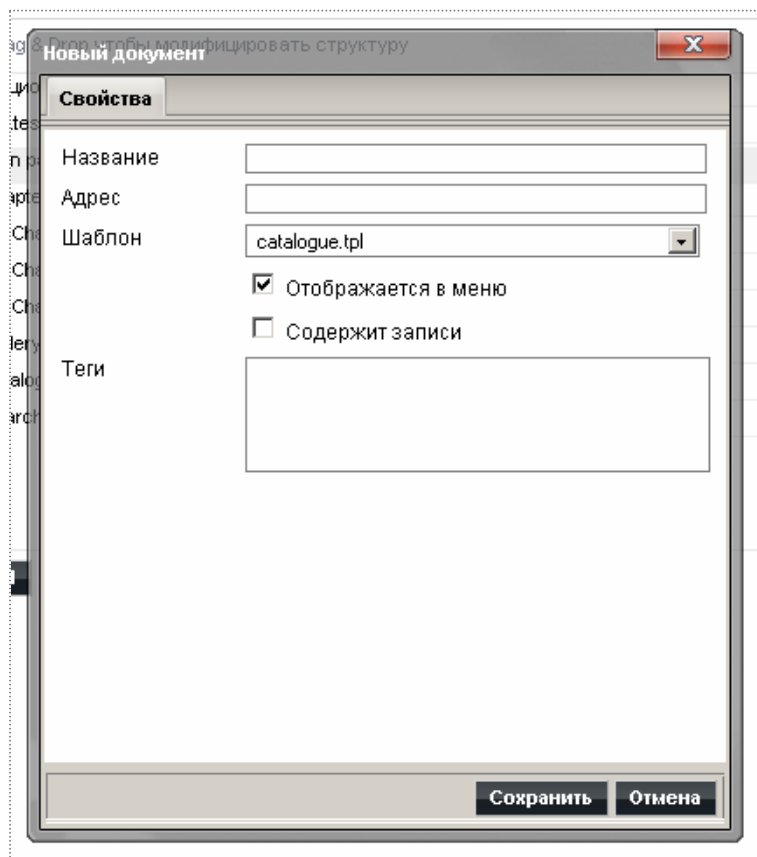
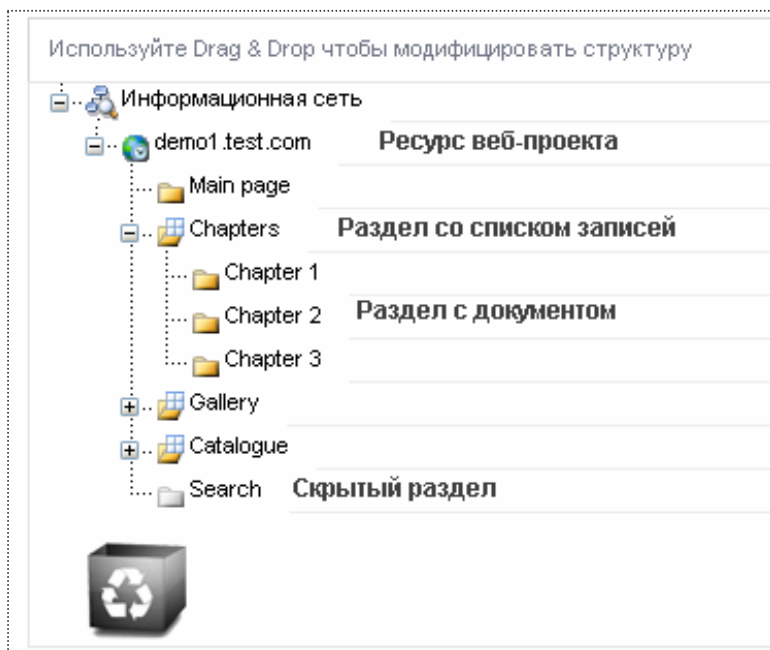
Данный раздел предназначен для управления документами, записями и файлами веб-проекта.

Управление документами

Интерфейс дерева управления структурой документов хорошо вам знаком по таким программам, как Проводник Microsoft Windows. С помощью курсора мыши вы можете раскрывать и закрывать ветви дерева, перемещать и копировать ветви (при копировании удерживается клавиша Ctrl), вы можете удалять ветви (перетаскивая их в корзину). При нажатии правой клавиши мыши над ветвью показывается меню доступных действий над данной ветвью.

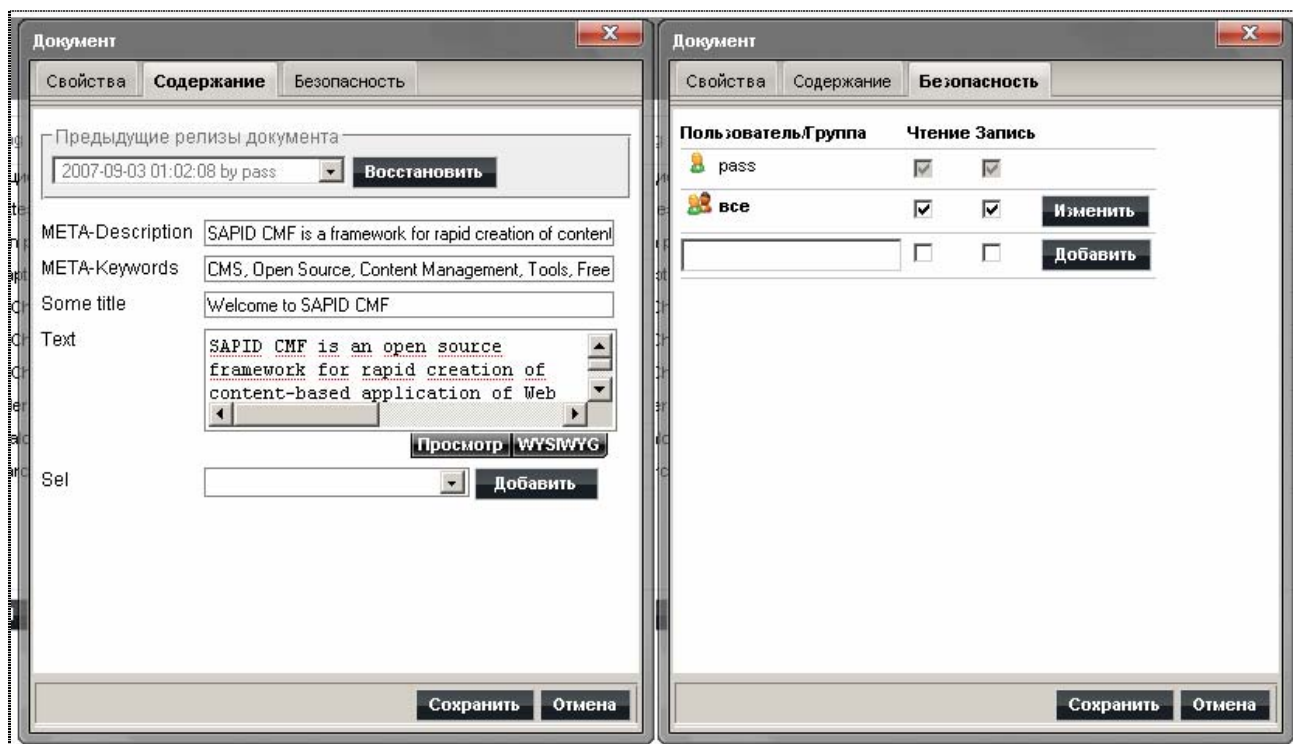
Напротив каждой ветви дерева имеются кнопки «Добавить подраздел» и «Свойства».

Для создания подраздела напротив выбранного раздела следует нажать кнопку «Создать подраздел» (Для создания ресурса следует нажать эту кнопку напротив элемента «Информационное пространство»). Перед вами предстанет форма для ввода названия нового раздела/документа, его адреса (имеется ввиду составная часть адреса, например about_us), связанного с ним шаблона. Там же вам будет предложено указать будет ли отображаться данный раздел в меню сайта (или же он будет скрытым), будет ли он содержать записи. Там же можно будет указать соответствующие разделу/документу теги (ассоциируемые с ним ключевые слов, по которым в дальнейшем можно будет восстановить нелинейные



зависимости между различными документами).

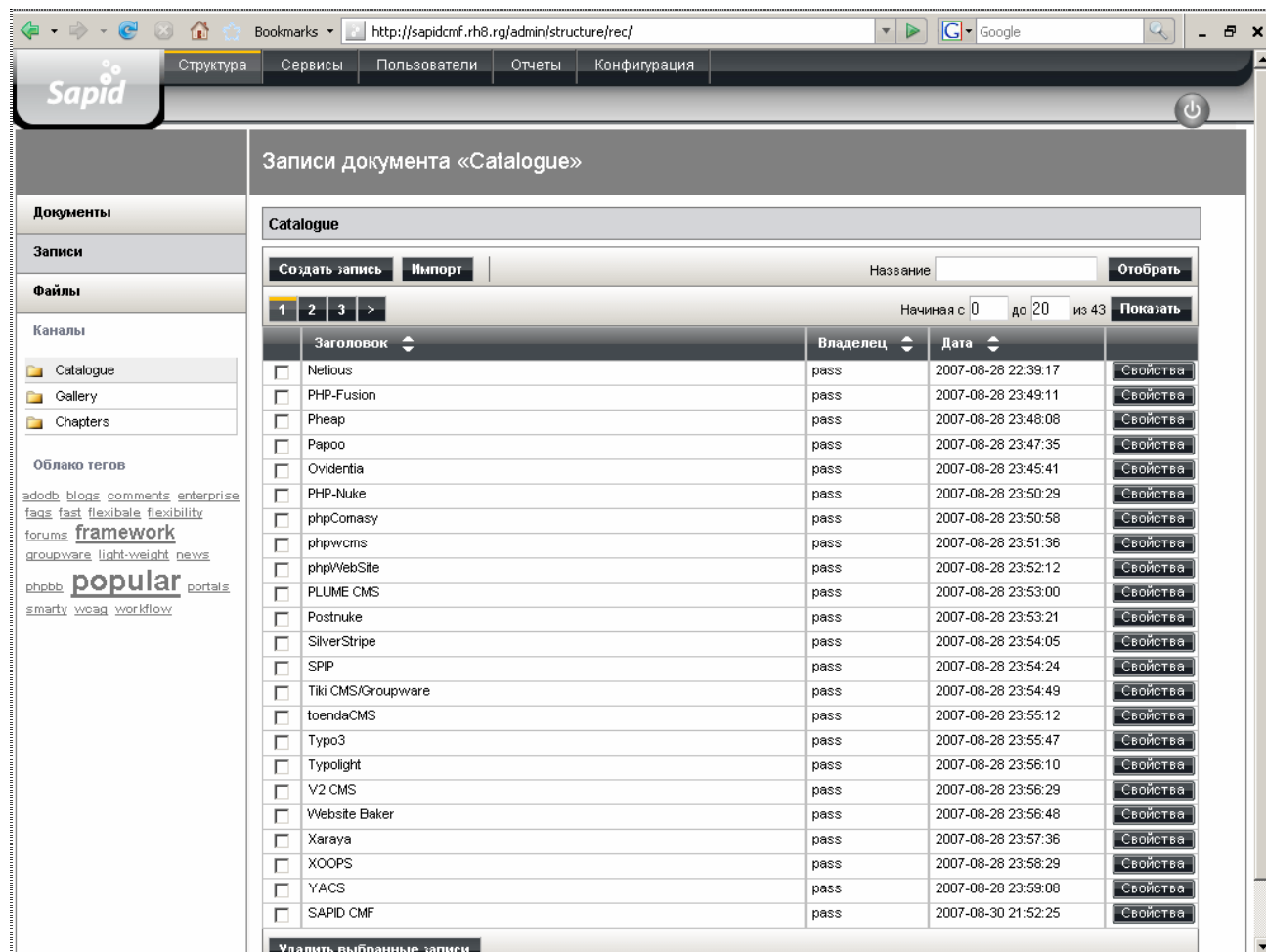
Кнопка «Свойства» служит для изменения свойств и содержания выбранного раздела/документа. При нажатии на нее вы увидите форму с тремя закладками «Свойства», «Содержание», «Безопасность». По умолчанию будет открыта первая закладка «Свойства». Она содержит форму аналогичную форму создания нового подраздела/документа. При выборе закладки «Содержание» вы обнаружите форму с запросами полей содержания, указанный в выбранном шаблоне документа (QC). Помимо этого будет предоставлен сервис для восстановления архивных копий содержания. Если в конфигурационном файле константе BACKUPING задано значение enabled при сохранении содержания платформа автоматически создает архивную копию содержания. Копии содержания будут накапливаться при каждом сохранении не более числа указанного в константе DOCCOPIESMAXNUMBERPERUSER на одного пользователя и не более числа DOCCOPIESMAXNUMBER на всех пользователей для данного документа.



Закладка «Безопасность» содержит список доступа для пользователей и групп для данного документа. Для добавления новой группы/пользователя в список следует указать его в соответствующем поле и нажать «Добавить».

Управление записями

В приложении Документа раздела Структура вы можете указать для определенного документа наличие записей в нем. Если это флаг выставлен в меню озаглавленном «Каналы» (ниже меню приложений) появиться данный раздел. При его выборе будет отображен грид (таблица).



Записи документа «Catalogue»

Catalogue

Создать запись | Импорт

Название: Отобразить

1 2 3 > Начиная с 0 до 20 из 43 Показать

Заголовок	Владелец	Дата	Свойства
<input type="checkbox"/> Netious	pass	2007-08-28 22:39:17	Свойства
<input type="checkbox"/> PHP-Fusion	pass	2007-08-28 23:49:11	Свойства
<input type="checkbox"/> Pheap	pass	2007-08-28 23:48:08	Свойства
<input type="checkbox"/> Papoo	pass	2007-08-28 23:47:35	Свойства
<input type="checkbox"/> Ovidentia	pass	2007-08-28 23:45:41	Свойства
<input type="checkbox"/> PHP-Nuke	pass	2007-08-28 23:50:29	Свойства
<input type="checkbox"/> phpComasy	pass	2007-08-28 23:50:58	Свойства
<input type="checkbox"/> phpwcms	pass	2007-08-28 23:51:36	Свойства
<input type="checkbox"/> phpWebSite	pass	2007-08-28 23:52:12	Свойства
<input type="checkbox"/> PLUME CMS	pass	2007-08-28 23:53:00	Свойства
<input type="checkbox"/> Postnuke	pass	2007-08-28 23:53:21	Свойства
<input type="checkbox"/> SilverStripe	pass	2007-08-28 23:54:05	Свойства
<input type="checkbox"/> SPIP	pass	2007-08-28 23:54:24	Свойства
<input type="checkbox"/> Tiki CMS/Groupware	pass	2007-08-28 23:54:49	Свойства
<input type="checkbox"/> toendaCMS	pass	2007-08-28 23:55:12	Свойства
<input type="checkbox"/> Typo3	pass	2007-08-28 23:55:47	Свойства
<input type="checkbox"/> Typolight	pass	2007-08-28 23:56:10	Свойства
<input type="checkbox"/> V2 CMS	pass	2007-08-28 23:56:29	Свойства
<input type="checkbox"/> Website Baker	pass	2007-08-28 23:56:48	Свойства
<input type="checkbox"/> Xaraya	pass	2007-08-28 23:57:36	Свойства
<input type="checkbox"/> XOOPS	pass	2007-08-28 23:58:29	Свойства
<input type="checkbox"/> YACS	pass	2007-08-28 23:59:08	Свойства
<input type="checkbox"/> SAPID CMF	pass	2007-08-30 21:52:25	Свойства

Удалить выбранные записи

В гриде имеется кнопка «Создать запись». При нажатии на нее появиться форма «Запись» с полями для ввода названия записи, набора полей и тегов. Набор полей для записей, что-то вроде шаблона для документов. Но в отличие от шаблона, набор полей используется лишь для задания каркаса содержания записи, но не ее оформления. Кроме того, набор полей может содержать различные каркасы содержания для различных состояний платформы. Состояние может быть указано при обращении к набору полей

```
<sapi:include href="имя_набора_полей" parse="fieldset" state="состояние" />
```


Запись

Свойства Содержание Безопасность

Название: PHP-Fusion

Набор полей: product.xml

Теги: light-weight, news, forums, polls, shoutbox, comments, ratings

Сохранить Отмена

pass

Файлы наборов полей содержатся в папке `views/delivery/fieldsets/`.

Запись может подобно документу содержать список записей. Что бы указать платформе, что запись сама является каналом, в назначаемом ей наборе полей добавляется конструкция `sapi:adopt`. В этой конструкции задаются наборы полей, доступные для записей потомков.

```
<?xml version="1.0"?>
<sapi version="1.0" xmlns:sapi="http://www.xmlsapiens.org/spec/sapi.dtd">
<sapi:fieldset type="default" state="admin" title="">
  <sapi:apply name="qc.recontent.value" type="article"
title="Description" />
  <sapi:apply name="qc.rectable.value" type="article"
title="Specification" />
  <sapi:adopt fieldset="default" />
</sapi:fieldset>
</sapi>
```

В форме «Запись» подобно форме «Документ» помимо закладки «Свойства», имеются также закладки «Содержание» и «Безопасность». Организованы они также как и в случае документов.

Управление файлами

При выборе приложения «Файлы» вам будет представлено меню доступных папок (физически это папки `views/delivery/files` и `views/delivery/img` и вложенные в них). При выборе любой из этих папок будет показан грид (таблица) с ее содержанием. Для добавления файла в папку имеется кнопка «Загрузить».

Имя файла	Размер	Дата
<input type="checkbox"/> alboun_sample1.jpg	48.3 KB	26.08.2007 16:18
<input type="checkbox"/> c-bury.png	370 bytes	26.08.2007 19:34
<input type="checkbox"/> c-vote.png	402 bytes	26.08.2007 19:34
<input type="checkbox"/> close_btn.gif	85.0 bytes	10.08.2007 21:28
<input type="checkbox"/> frame.gif	1.30 KB	26.08.2007 18:23
<input type="checkbox"/> leftcor.gif	211 bytes	10.08.2007 21:28
<input type="checkbox"/> livedemo.gif	7.83 KB	02.09.2007 14:28
<input type="checkbox"/> loading.gif	2.70 KB	10.08.2007 21:28
<input type="checkbox"/> openid.gif	550 bytes	03.09.2007 20:52
<input type="checkbox"/> pframe.gif	1.19 KB	06.09.2007 22:10
<input type="checkbox"/> rightcor.gif	212 bytes	10.08.2007 21:28
<input type="checkbox"/> rss_icon.jpg	1.03 KB	06.09.2007 22:15
<input type="checkbox"/> sample1.gif	327 bytes	10.08.2007 21:28
<input type="checkbox"/> working.gif	1.12 KB	10.08.2007 21:28
<input type="checkbox"/> x.gif	43.0 bytes	10.08.2007 21:28

Раздел Сервисы

Раздел сервисы содержит секции «Реклама», «Резервное копирование», «Системная консоль», «Список рассылки», «Тезаурус». Причем вы можете обогатить это меню на свое усмотрение (См. раздел «Создание административных приложений»).

Управление рекламой

В примере веб-проекта из дистрибутивного пакета вы можете найти решение по использованию рекламы на вашем сайте. В шаблонах указывается рекламное место посредством DDC

```
<sapi:apply name="ddc.ad.value" />
```

Реклама, которая будет отображаться в этой позиции управляется из приложения «Реклама» раздела «Сервисы». Для внесения нового рекламного носителя (баннера) в ротации следует нажать «Добавить новую позицию». У вас будут запрошены название баннера, адрес для перехода при клике, его HTML код или графический файл на ваш выбор. Статистика показов баннеров будет также доступна в гриде секции. Там есть инструменты для удаления баннеров и изменения их свойств.

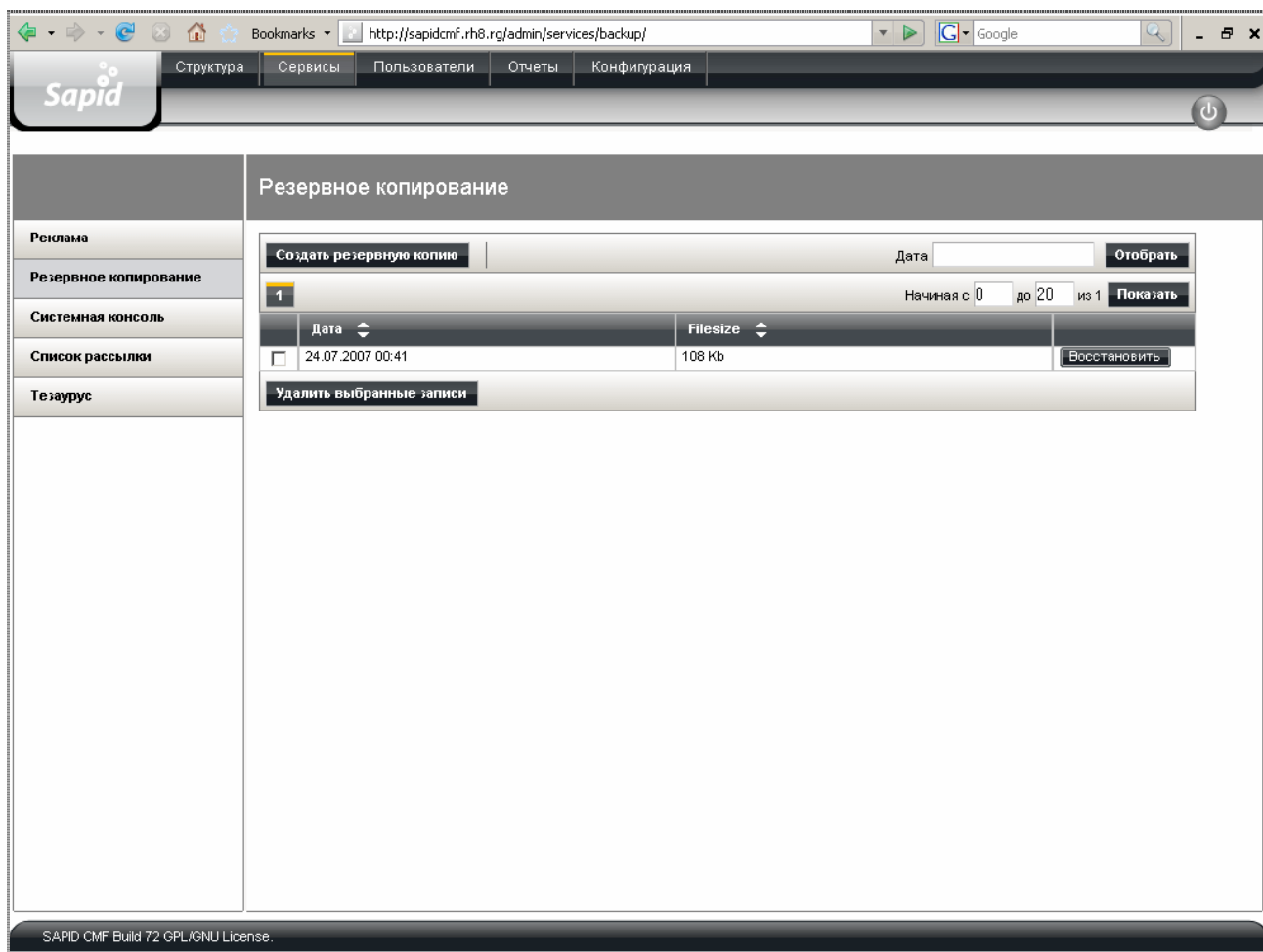
Скриншот административного интерфейса SAPID CMF. В верхней части видна панель навигации с вкладками: Структура, Сервисы (выбрана), Пользователи, Отчеты, Конфигурация. Слева находится боковая панель с меню: Реклама, Резервное копирование, Системная консоль, Список рассылок, Тезаурус. Основное содержимое — раздел «Сервисы». В нем есть кнопка «Добавить новую позицию» и поле «Название» с кнопкой «Отобразить». Ниже — таблица с 5 записями. Каждая запись имеет чекбокс, заголовок, количество показов и переходов, а также кнопку «Свойства».

	Заголовок	Показов	Переходов	
<input type="checkbox"/>	Site Guide Toolkit	15	0	Свойства
<input type="checkbox"/>	AntiSpam Feedback Toolkit	20	0	Свойства
<input type="checkbox"/>	Site Sapiens 3.0 ECMP	57	2	Свойства
<input type="checkbox"/>	Thesaurus Tooltip Kit	57	4	Свойства
<input type="checkbox"/>	Easy Grid	56	1	Свойства

В нижней части интерфейса видна строка с текстом: SAPID CMF Build 72 GPL/GNU License.

Управление резервным копированием

Для создания новой резервной копии базы данных следует воспользоваться приложением «Резервное копирование» раздела «Сервисы». При нажатии кнопки «Создать резервную копию» создается дамп БД в папке tmp и строка о нем появляется в гриде секции. Напротив каждой из строк есть кнопка «Восстановить», для восстановления связанного со строкой дампа.



Системная консоль

Приложение «Системная консоль» служит для низко уровняемого управления платформой. В поле «Команда API» можно задать один из трех режимов работы с консолью API: , SITE: , SQL:

Режим API подразумевает непосредственный ввод команд API платформы. Например, `print_r($doc->get(1));`

Режим SQL используется для задания команд SQL, скажем `SELECT * FROM sf_thesaurus LIMIT 0,10`

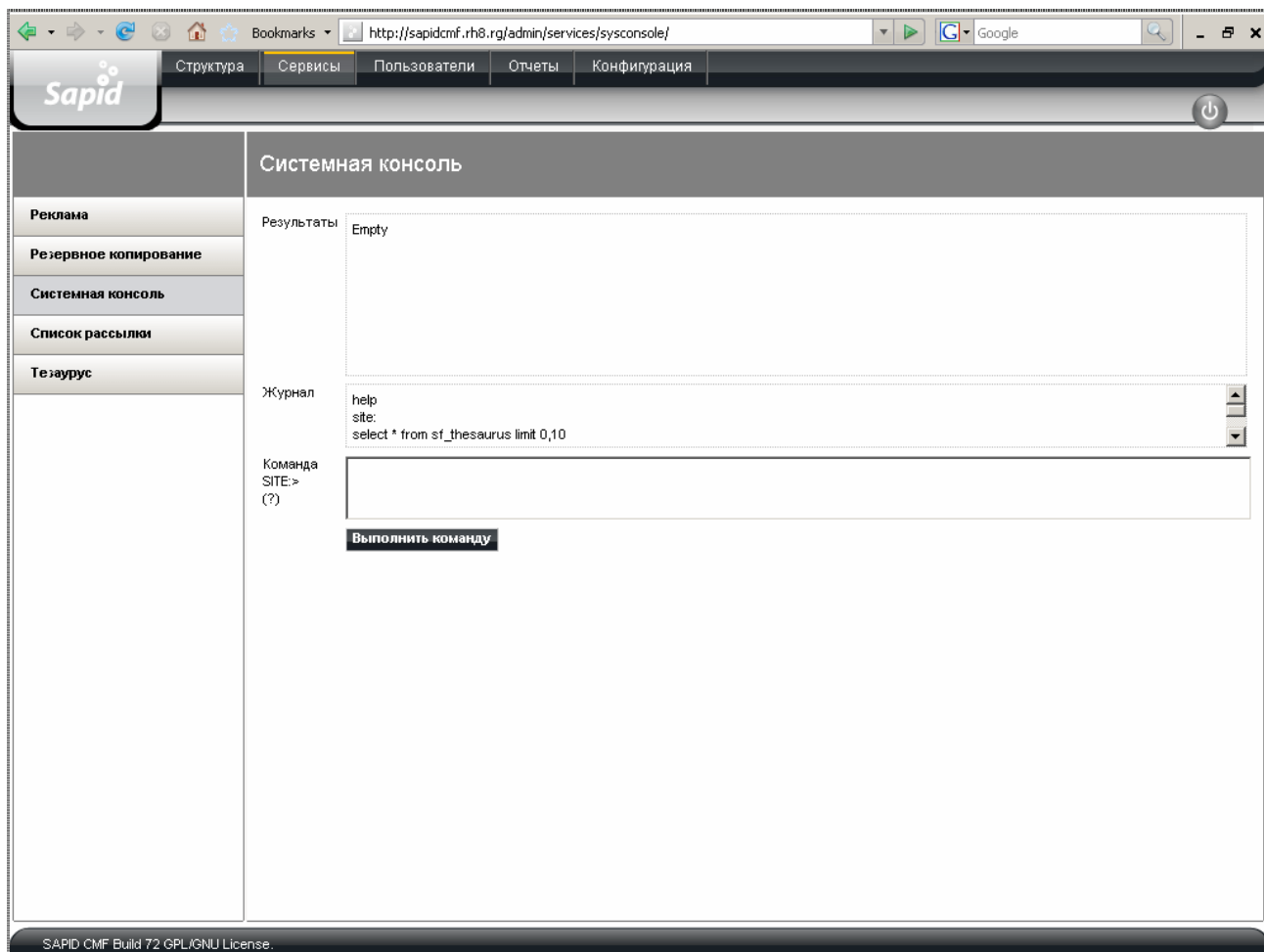
Режим SITE можно использовать для манипуляций над структурами и содержанием. В настоящий момент доступны лишь команды:

Help – получить подсказку по командам

Dir – показать содержание раздела

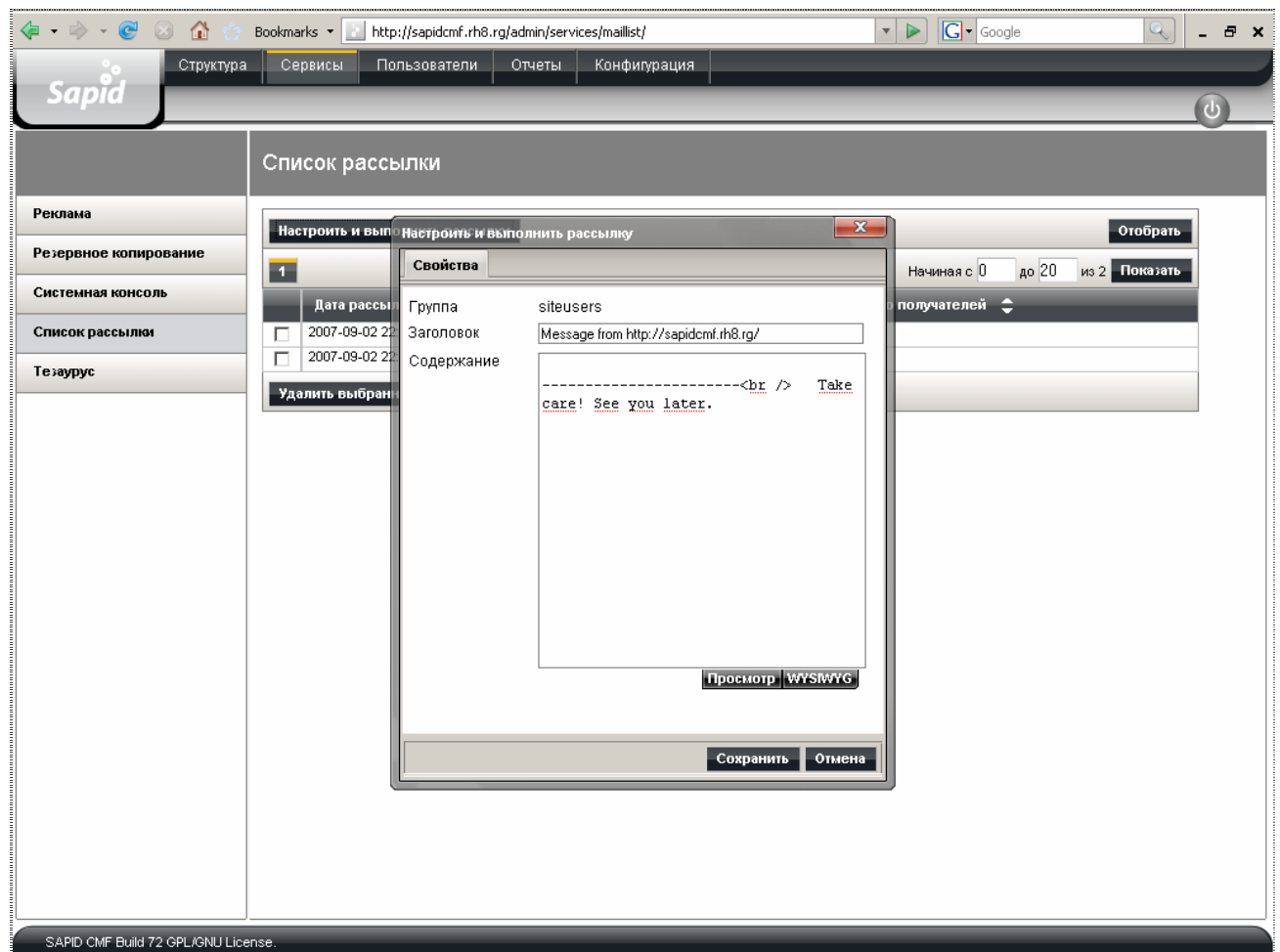
Cd – выбрать раздел

Index – индексировать повторно теги документов и разделов.



Управление рассылками

Для выполнения рассылки из проекта следует воспользоваться секцией «Список рассылки» раздела «Сервисы». При переходе по кнопке «Настроить и выполнить рассылку» будут запрошены заголовок и содержание письма. Причем в поле содержания будет предложено оформление согласно шаблону `views/delivery/templates/maillists/default.tpl`. Имейте в виду, вы вольны указать в данном шаблоне вызовы DDC для отображения скажем списка последних новостей.

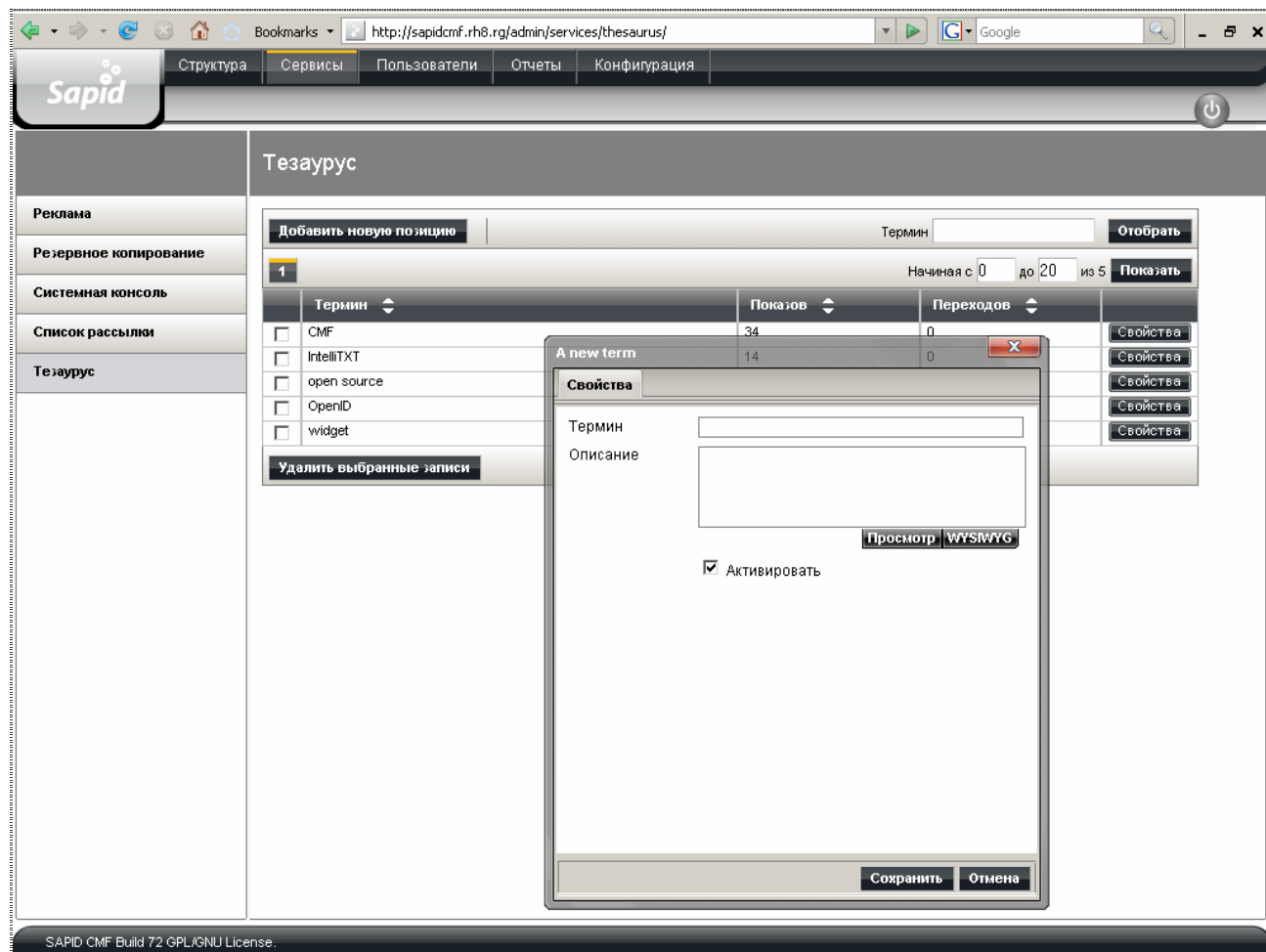


Управление тезаурусом

Вы можете ввести словарь терминов в приложении «Тезаурус» и включить в шаблоны сервис Тезаурус.

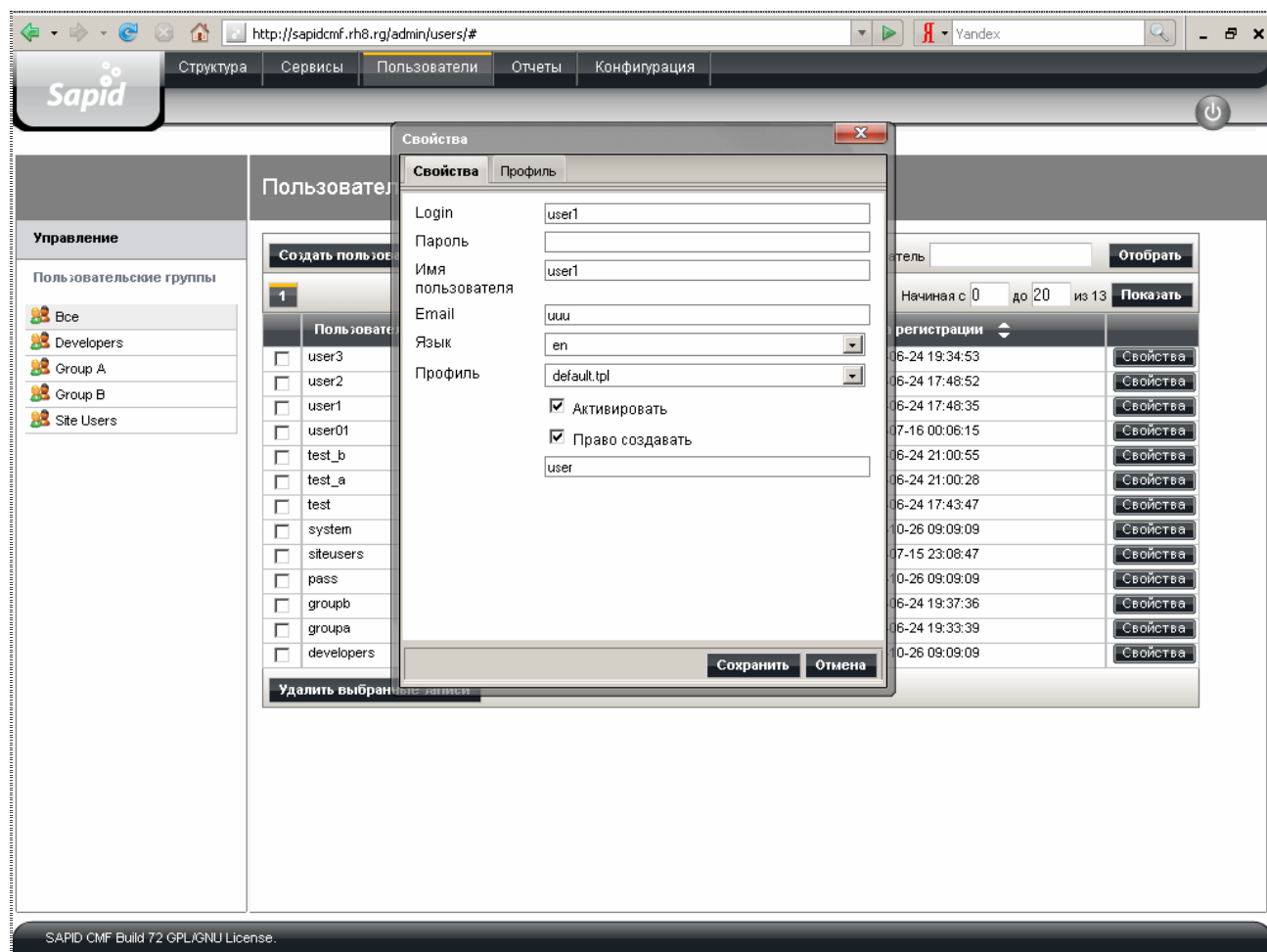
```
<script src = "&http_path.value;vendors/yui/yahoo/yahoo.js" ></script>
<script src = "&http_path.value;vendors/yui/connection/connection-min.js"
></script>
<script src = "&http_path.value;views/delivery/js/thesaurus.js" ></script>
<script>
    THConfig.parseElements = ['PageContent', 'Footer'];
    <sapi:apply exp="isequal('&state.value;', 'edit', '///','')" />
    parseContent();
</script>
```

После этого все термины из словаря по мере их наличия на страницах веб-проекта будут преобразованы в ссылки. При наведении курсора мыши на такую ссылку будет появляться определение сервиса (tooltip).



Пользователи

По адресу `/admin/users/` расположено приложение для управления пользователями. В левой части экрана расположен фильтр по группам, в центре таблица пользователей. По умолчанию таблица содержит все учетные записи платформы – всех пользователей и группы. Над таблицей расположены кнопки для добавления в платформу нового пользователя и для создания новой группы. Напротив каждой записи таблицы имеется кнопка для изменения свойств пользователя или группы. Для удаления учетных записей следует выбрать (чекбоксы) записи и нажать кнопку «Удалить выбранные записи» под списком.



Отчеты

По адресу `/admin/reports/` расположено приложение для управления отчетами. В настоящий момент он содержит лишь отчет о сеансах администрирования. Однако вы можете добавлять в него собственные отчеты.

The screenshot shows the SAPID CMF web interface. The browser address bar displays <http://sapdcmf.rh8.rg/admin/reports/>. The navigation bar includes links for 'Структура', 'Сервисы', 'Пользователи', 'Отчеты', and 'Конфигурация'. The 'Отчеты' section is active, showing a sidebar labeled 'Протокол' and a main area with a date filter, a table of reports, and pagination controls.

Дата: Отобразить

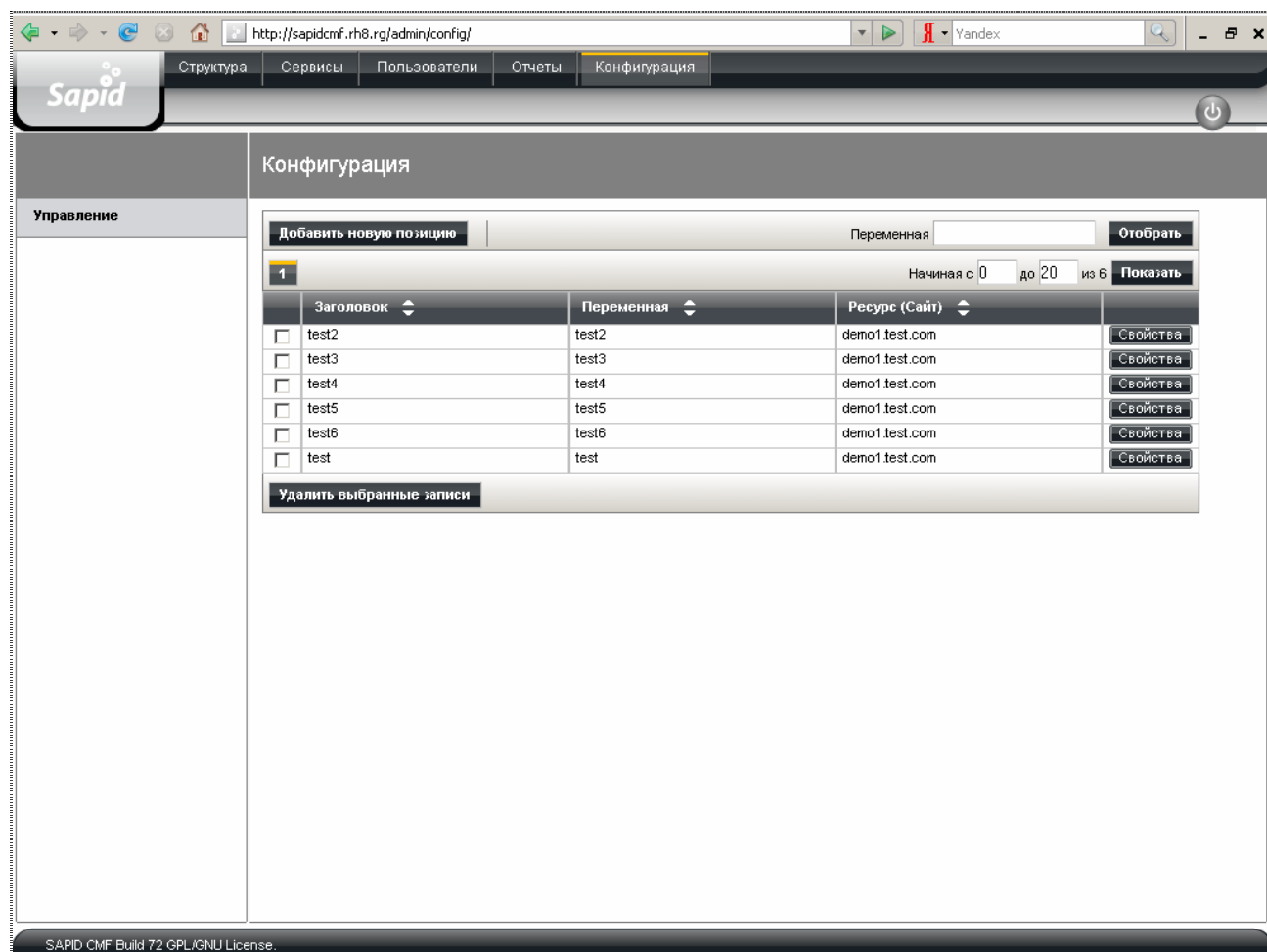
1 2 > Начиная с 0 до 20 из 34 Показать

Дата	Описание	Пользователь
2007-12-09 17:39:59	Обновлен документ	pass
2007-12-09 17:45:54	Обновлен документ	pass
2007-12-09 17:46:56	Обновлен документ	pass
2007-12-09 17:47:27	Обновлен документ	pass
2007-12-09 17:57:32	Обновлен документ	pass
2007-12-09 18:00:35	Обновлен документ	pass

SAPID CMF Build 72 GPL/GNU License.

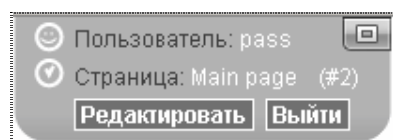
Конфигурация

Приложение «Конфигурация» позволяет определить дополнительные конфигурационные переменные и регулировать их значения. Эти переменные будут доступны в шаблонах как `config_переменная.value`.

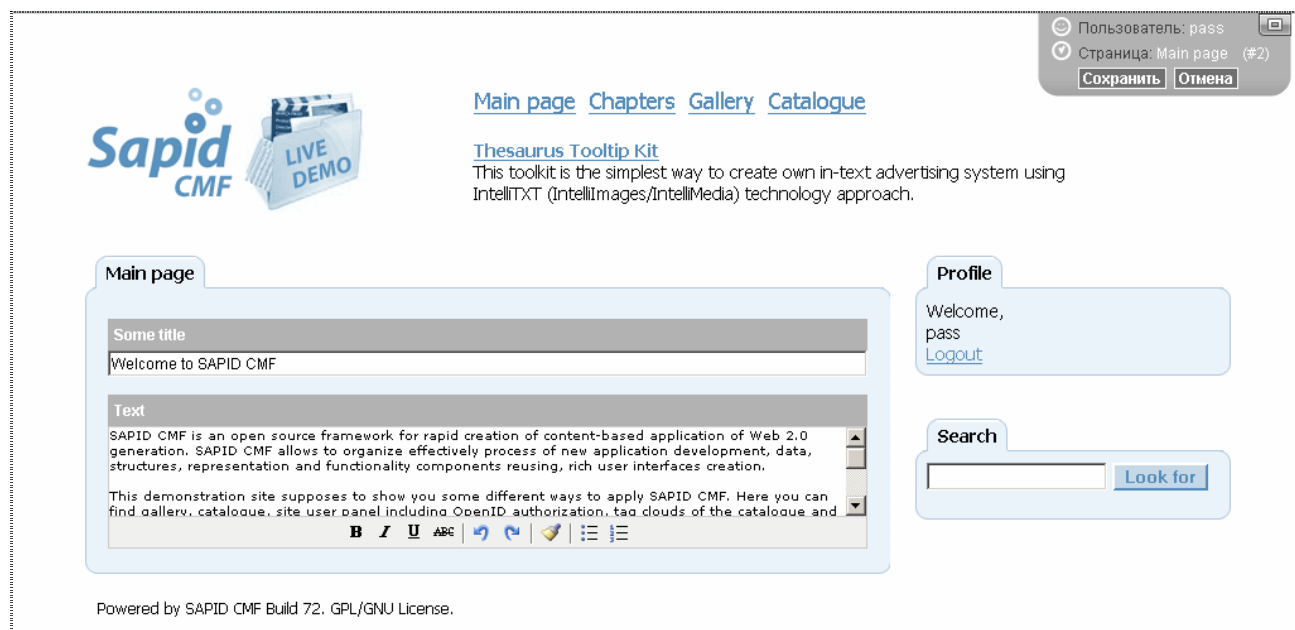


Администрирование непосредственно на сайте

Содержанием проекта также можно управлять в режиме INLINE. Если вы авторизованы в проекте как администратор (пользователь группы developers), то на страницах сайта вы обнаружите контрольную панель



Это панель содержит имя авторизованного пользователя, название и ID просматриваемой страницы. Кнопка «Редактировать» позволит перейти в режим редактирования текущей страницы. В этом режиме на месте полей содержания будут показаны запросы содержания.



После заполнения запросов следует нажать кнопку «Сохранить».

Если страница содержит список записей и DDC списка содержит элементы INLINE (см. /views/delivery/ddcs/infochannel.xml), а пользователь авторизован как администратор, над списком появится кнопка «Добавить», в списке будут представлены кнопки «Изменить» и «Удалить» напротив каждой записи.



[Main page](#) [Chapters](#) [Gallery](#) [Catalogue](#)

[AntiSpam Feedback Toolkit](#)
This free package can be used to prevent receiving of SPAM messages from your feedback form.

Catalogue

Главная > Catalogue

Добавить

SAPID CMF



SAPID CMF is a free framework for rapid creation of content-based application of Web 2.0 generation. SAPID CMF allows to organize effectively process of new application development, data, structures, representation and functionality components reusing, rich user interfaces creation.

Изменить Удалить

YACS



Yet Another Community System (YACS) is an open source, open-standards content management system based on PHP and MySQL with a focus on code quality and best practices. The YACS product has a straightforward and modular component architecture that covers most common needs of demanding web masters. By design, YACS makes an efficient usage of computing and networking resources, and it allows for several kinds of extensions.

Изменить Удалить

XOOPS



XOOPS is a program that allows administrators to easily create

Изменить Удалить

Profile

Welcome,
pass
[Logout](#)

Tag Cloud

[adodb](#) [blogs](#) [comments](#) [enterprise](#) [faq](#) [fast](#)
[flexible](#) [flexibility](#) [forums](#) [framework](#)
[groupware](#) [light-weight](#) [news](#) [phpbb](#)
[popular](#) [portals](#) [smarty](#) [wcag](#)
[workFlow](#)

Widget

You can use this code to get catalogue/gallery RSS widget within your blog

```
<OBJECT  
classid="clsid:D27CDB6E-AE6D-11cf-96B8  
codebase="http://download.macromedia  
WIDTH="400" HEIGHT="400"  
id="RssReader" ALIGN=""> <PARAM  
NAME=movie  
VALUE="http://sapidcmf.rh8.ru/widgets/  
<PARAM NAME=quality VALUE=high>
```

При нажатии «Добавить» появится окно со свойствами новой записи, аналогичное тому, что мы можем видеть в административной панели при добавлении новой записи. Когда запись будет добавлена, она появится в списке, после чего можно будет нажать «Изменить» для внесения содержания.

Отладка информационной системы

Для включения основного режима отладки назначьте значение константе DEBUG 1. В случае возникновения ошибок в платформе вы получите сообщение о них в специальном окне.

При возникновении ошибок платформа помещает их в отчет в файле TMP/error.log.

Пример:

```
function getDDCCache($obj) {
    if(!isset($obj["AppParams"])) return false;
    if(!isset($obj["AppParams"]["name"])) return false;
    if($obj["AppParams"]["name"]!="test") return false;

    if(file_exists(ROOT_PATH."tmp/ddcs/test.tmp")) {
        $saved =
unserialize(file_get_contents(ROOT_PATH."tmp/ddcs/test.tmp"));
        $obj["AppParams"]    = $saved["AppParams"];
        $obj["DDCParams"]    = $saved["DDCParams"];
        $obj["EnumAttrs"]    = $saved["EnumAttrs"];
        $obj["Results"]      = $saved["Results"];
        return true;
    } else return false;
}

function putDDCCache($obj) {
    if(!isset($obj["AppParams"])) return false;
    if(!isset($obj["AppParams"]["name"])) return false;
    if($obj["AppParams"]["name"]!="test") return false;
    toLog(serialize($obj),"a","ddcs/test.tmp");
    return true;
}

$cache = new Aspect();
$pc = $cache->pointcut("call *::*");
$pc->_event("getDDCCache", "getDDCCache(\$obj);");
$pc->_event("putDDCCache", "putDDCCache(\$obj);");
$pc->destroy();
Aspect::apply($cache);
```

Определенные в системе события

The_program_is_started – платформа стартовала, основные библиотеки не загружены;

All_libraries_are_loaded – платформа стартовала, загружены все библиотеки;

Error403Found – возникла 403 ошибка;

Error404Found – возникла 404 ошибка;

ModalWindowSchemasDefined – определены схемы модальных окон административной панели;

Программный код адаптации

В программном коде адаптации доступны среда окружения в структуре \$env и функции общего назначения.

Структура \$env

\$env→SiteID – содержит ID текущего ресурса;

\$env→Document – содержит массив с данными о документе, где

Template – имя файла шаблона документа;

Data – массив с содержанием документа;

ID – ID документа;

PointerID – ID зеркалируемого документа;

DataPointerID – ID документа, с которого зеркалируются данные;

Redirect – адрес для перенаправления;

\$env→URL – содержит информацию о запросе пользователя из адресной строки;

\$env→User – содержит информацию о текущем пользователе, где

ID – ID пользователя;

Groups – группы пользователя;

Profile – профиль пользователя;

CreationPermission – права на создание документов;

SysAdminPermission – права сисадмина;

ProfileID – ID профиля пользователя;

\$env→Values – содержит переменные среды, где

args_length – число элементов командной строки;

extraargs_length – число дополнительных аргументов;

DirectAccessArgs_length – вложенность канала;

document_url – адрес в формате branch1/digital_cams/;

`document_url_without_slash` - адрес в формате `branch1/digital_cams`;

`args.x` – аргумент номер `x`;

`extraargs.x` – аргумент номер `x`;

`argsstring` – адрес в формате `/branch1/digital_cams/`;

`argsstringwithoutslash` – адрес в формате `branch1/digital_cams`;

`argsstringwithoutdatasection` – адрес в формате `branch1/digital_cams`;

`argsstringwithoutnavigationsection` – адрес в формате `branch1/digital_cams`;

`admin_webroot_http_path` – адрес `http://сайт/webroot/admin/`;

`delivery_webroot_http_path` – адрес `http://сайт/webroot/delivery/`;

`http_path` – адрес `http://сайт/`;

`root_path` – корневой адрес;

`plugins_root_path` – корневой к папке плагинов;

`admin_http_path` – адрес административного интерфейса;

`system_configuration_date` – дата конфигурации;

`system_version` – версия платформы;

`user_id` – ID авторизованного пользователя;

`site_id` – ID сайта по умолчанию;

`this_record_id` – ID записи, если текущий документ запись;

`licenseno` – номер лицензии;

`document_атрибут` – значение атрибута текущего документа;

`this_level` – текущий уровень вложенности;

`this_name` – заголовок текущего объекта (документ/запись);

`this_parent_id` – ID родительского объекта к текущему;

`config_переменная` – значение переменной конфигурации;

`document_data_переменнаяданных` – данные документа, если это документ (например, `document_data_content`);

`record_data_переменнаяданных` – данные записи, если это запись (например, `record_data_recontent`).


```
<head>
    <title>FCKEditor - Sample</title>
    <meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
    <link href="../../../sample.css" rel="stylesheet" type="text/css"
/>
    <script type="text/javascript"
src="../../../fckeditor.js"></script>
</head>
<body bgcolor="#D9D6CF">
<form>
    <script type="text/javascript">
function saveContent() {
    var oEditor = FCKEditorAPI.GetInstance('FCKEditor1') ;

    window.opener.document.getElementById('<?=$_GET["id"]?>').value =
oEditor.GetXHTML( true );
    window.close();
}

    var sBasePath =
document.location.pathname.substring(0,document.location.pathname.
lastIndexOf('_samples')) ;
    var oFCKEditor = new FCKEditor( 'FCKEditor1' ) ;
    oFCKEditor.BasePath = sBasePath ;
    oFCKEditor.Height = 500 ;
    oFCKEditor.Config['HTTP_ADMINPATH'] = '<? print
HTTP_PATH.ADMINAREA_PREFIX."/"; ?>';
    oFCKEditor.Config['AutoDetectLanguage']= false;
    oFCKEditor.Config['DefaultLanguage']= '<? print DEFAULT_LANUAGE;
?>' ;
    oFCKEditor.Config['SkinPath'] = sBasePath + 'editor/skins/silver/'
;
    oFCKEditor.Value =
window.opener.document.getElementById('<?=$_GET["id"]?>').value;
    oFCKEditor.Create() ;
    //-->
</script>
<input type="button" value="<? print lang('Changes saving'); ?>"
onclick="saveContent();">
    </form>
</body>
</html>
```

2) Перенаправляем вызовы файлового менеджера на встроенный менеджер SAPID CMF. Следует внести следующие строки в файл `/vendors/fckeditor/editor/dialog/common/fck_dialog_common.js` сразу после открытия функции `OpenFileBrowser()` (предположительно строка 96):

```

window.open(FCKConfig.HTTP_ADMINPATH+'structure/popup_files/?group=img&ca
llback_id=txtUrl', 'popup',
'scrollbars=yes,status=no,resizable=0,toolbar=no,width=790,height=600');
return false;

```

Адаптация административной панели

AJAX фреймворк темы DEFAULT административной панели позволяет задавать образ модальных окон посредством схем. Схемы расположены в скрипте `/views/default/js/config.js.php` и имеют следующий синтаксис:

```
$ModalWindowSchemas['ShowDocModalWindow'] =
'var Schema_ShowDocModalWindow = [
{"ctrl":"admin/structure/doc/docproperties.ctrl.php",
"title":"' .lang("Properties").'"},
{"ctrl":"admin/structure/doc/doccontent.ctrl.php","title":"' .lang("Content").'"},
{"ctrl":"admin/structure/doc/security.ctrl.php",
"title":"' .lang("Security").'"}];';

$ModalWindowSchemas['ShowNewDocModalWindow'] = 'var
Schema_ShowNewDocModalWindow =
[{"ctrl":"admin/structure/doc/blank_docproperties.ctrl.php",
"title":"' .lang("Properties").'"}];';

$ModalWindowSchemas['ShowSiteModalWindow'] = 'var
Schema_ShowSiteModalWindow =
[{"ctrl":"admin/structure/doc/siteproperties.ctrl.php",
"title":"' .lang("Properties").'"}];';
```

Переменная `Schema_xxx` содержит массив, определяющий закладки модального окна `{"ctrl":"адрес контроллера", "title":"заголовок"}, ...`

Для адаптации модального окна заданного типа следует переопределить его схему в соответствующем плагине.

Пример плагина адаптации:

Plugins/index.php

```
// Sample of administrative panel modal window customization
include(ROOT_PATH."plugins/ModalWindowCustomization.inc.php");
$ModalWindowCustomization = new Aspect();
$pc = $ModalWindowCustomization->pointcut("call *::*");
$pc->_event("ModalWindowSchemasDefined",
"customizeModalWindow(&\$obj);" );
```

```
$pc->destroy();  
Aspect::apply($ModalWindowCustomization);
```

Plugins/ ModalWindowCustomization.inc.php

```
<?  
function customizeModalWindow($ModalWindowSchemas) {  
    $ModalWindowSchemas['ShowDocModalWindow'] = 'var Schema_ShowDocModalWindow  
= [{"ctrl":"admin/structure/doc/docproperties.ctrl.php",  
    "title":"' . lang("Properties") . '"},  
    {"ctrl":"admin/structure/doc/doccontent.ctrl.php","title":"Something  
new"},  
    {"ctrl":"admin/structure/doc/security.ctrl.php",  
    "title":"' . lang("Security") . '"}]';  
}  
?>
```

Разработка тем оформления административного интерфейса

В конфигурационном файле `config/rc.conf.php` задается текущая папка текущей темы оформления

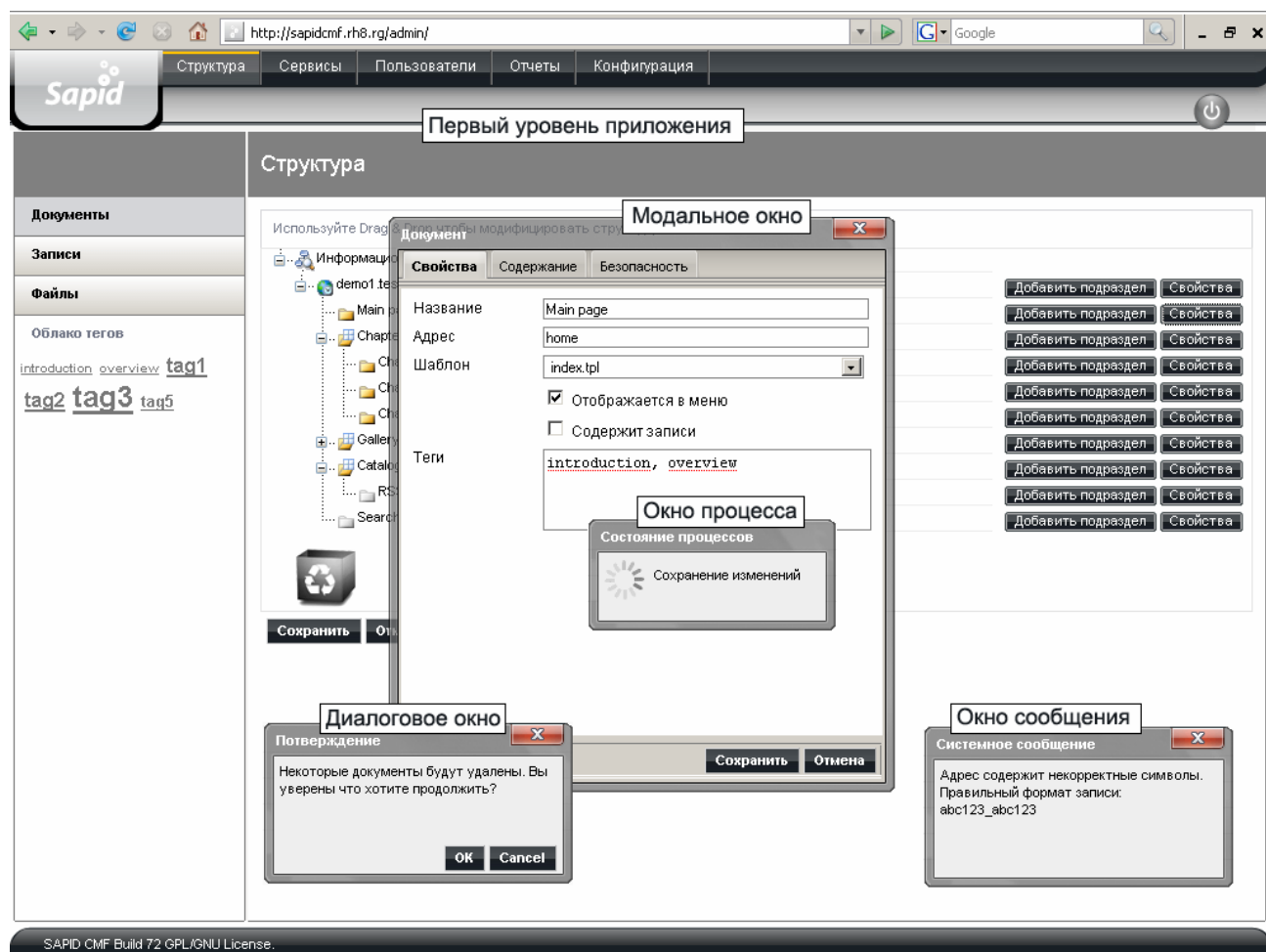
```
define("THEME", "default");
```

В данной папке расположены шаблоны и вспомогательные файлы административного интерфейса

Создание административных приложений

Состав приложения

Любой пользовательский интерфейс административной панели в SAPID CMF содержит первый уровень, которому отведена рабочая область панели.



Для изменения атрибутов и содержания элементов интерфейса первого уровня могут быть использованы модальные окна. Для любых операций на стороне сервера, исключая запросы страниц, используются асинхронные вызовы контроллеров. Если контроллеру требуется сообщить о чем-либо назад в интерфейс используется окно сообщения.

Синтаксис JS:

```
showSystemMessage("Сообщение");
```

Если платформа занята каким-либо процессом это будет показано посредством окна процесса.

Синтаксис JS:

```
showProcessMessage('Сообщение');  
hideProcessMessage();
```

Также существует разновидность окна сообщения с кнопками «ОК» и «Отмена». Такие диалоговые окна используются для запроса подтверждения перед выполнением операции.

Синтаксис JS:

```
showConfirmationMessage('Сообщение', okActionFunction);
```

Создание приложения

Для создания интерфейса первого уровня, прежде всего, следует определить его секцию в горизонтальном меню административной панели. Скажем, если мы желаем добавить новый отчет следует выбрать секцию «Отчеты». Итак, заходим в папку /app/admin/reports и находим там один уже имеющийся интерфейс logs. Создаем свою папку по аналогии, скажем, ecommerce. Добавляем в нее файлы index.layout.php, app.db, ecommerce.ctrl.php.

Список доступа app.db может быть следующим:

```
<?xml version="1.0" encoding="utf-8" ?>  
<treeitem titlekey="Ecommerce" sortkey="2">  
  <acl>  
    <group name="developers" permissions="7" />  
  </acl>  
</treeitem>
```

Где значение атрибута titlekey – название интерфейса (если в языковом массиве xx.lang.php не будет перевода для данного ключа, в левом меню интерфейсов секции появиться пункт Ecommerce). Значение sortkey указывает на положение интерфейса в этом меню.

Содержание index.layout.php может быть таким

```
<?php  
$env->Values["window_interface_title"] = lang("Logging");  
$tpl = factory("view","template");  
$output = $tpl->parseTemplate("#ecommerce.tpl");  
?>
```

Где мы задаем название интерфейса для отображения в рабочей области, а также передаем управление шаблону административной панели ecommerce.tpl

Что касается контроллера ecommerce.ctrl.php, его содержание следует строить по следующей схеме:

```
<?PHP  
class Controller {  
  var $json
```



```
function Controller() {  
    global $json;  
    $this->json = &$json;  
    return $this;  
}  
  
function remoteProcedure1() {  
    //...  
    $this->json->Body = "...";  
    $this->json->ErrorMsg = "...";  
    $this->json->ActionCode = 1;  
}  
}  
$ctrl = new Controller();  
if(isset($_POST["call"]))  
    $ctrl->$_POST["call"]();  
?>
```

Запрос на этот контроллер, содержащий в переменной call имя удаленной процедуры (например, remoteProcedure1) инициализирует класс Controller и выполнит запрашиваемую процедуру. В процедуре может быть определен ответ сервера интерфейсу. Если в процедуре будет назначено значение переменной \$this->json->Body, это значение затем будет отображено в окне сообщения. Значение \$this->json->ErrorMsg будет отображено в окне сообщения об ошибке. Значение \$this->json->ActionCode может быть обработано в JS особым образом.

Теперь давайте создадим шаблон, на который уже ссылается наш index.layout.php. Итак, создаем файл /views/default/templates/ecommerce.tpl.

```
<sapi:include href="#subtemplates/header.tpl" />  
<link rel="STYLESHEET" type="text/css" href="/views/default/css/grid.css" />  
<script src="/views/default/js/json_grid.js" type="text/javascript"></script>  
<body onload="initInterface()">  
<sapi:include href="#subtemplates/toppanel.tpl" />  
    <div class="smf_caption">&nbsp;</div>  
    <sapi:apply name="ddc.#submenu.value">  
        <sapi:param name="root">&args.0.value;</sapi:param>  
    </sapi:apply>  
    <div class="smf_docinfo">  
    </div>  
<sapi:include href="#subtemplates/middle.tpl"/>  
  
    <div class="smf_place">  
        Код рабочей области
```

```
</div>  
<sapi:include href="#subtemplates/footer.tpl"/>
```

В блоке код рабочей области может быть представлена форма, дерево или, что чаще всего, таблица. В случае формы мы вносим в эту часть код элементов формы. Для кнопок определяем события и описываем их в JS. Например,

```
function ourEventAction(param1, param2) {  
    showProcessMessage(lang("Please wait"));  
    serverRequest("admin/reports/ecommerce/ecommerce.ctrl.php", "call==  
remoteProcedure1&param1="+param1+"&param2="+param2, processDataRestoring,  
false);  
}
```

И соответственно кнопка типа `<button onclick=" ourEventAction(1,1)">Выполнить</button>`

С интерфейсом дерева сложнее. С примером такого интерфейса можно ознакомиться в шаблоне `tree.tpl`. Но чаще всего используются интерфейсы с таблицами.

Создание приложений со списками

В случае интерфейса со списком код рабочей области шаблона может быть таким:

```
<sapi:apply name="ddc.#gridnav.value">  
    <sapi:param name="grid_id">ecommerce</sapi:param>  
    <sapi:param name="filters">  
        <sapi:apply exp="lang('Date')"/> <input class="filter"  
type="text" name="edate" value="" />  
    </sapi:param>  
</sapi:apply>  
  
<script>  
    EGConfig.sUrl = "admin/reports/ecommerce/ecommerce.ctrl.php";  
    settings(columns, '{"field":"edate","title":lang("Date"),  
"width":"25%"}');  
    settings(columns, '{"field":"edesc","title":lang("Transaction"),  
"width":"70%"}');  
    settings(columns, '{"field":"user","title":lang("User"),  
"width":"15%"}');  
    makeRequest('');  
</script>
```

В параметре `filters` DDC `gridnav` задаются дополнительные фильтры таблицы. В JS описывается конфигурация таблицы.

EGConfig.sUrl – сообщает платформе местоположения контроллера грида

Settings – задает структуру списка, где каждый столбец описывается следующим образом:

```
settings(columns, '{"field":"имя поля","title":lang("Заголовок поля"), "width":"ширина колонки списка"}');
```

Команда makeRequest(""); запрашивает у сервера список.

В данном случае грид будет содержать три столбца и его содержание будет запрашиваться у контроллера ecommerce.ctrl.php. Однако наш контроллер пока не умеет возвращать содержание грида. Нам следует его научить. Давайте перепишем контроллер, так чтобы он лучше служил нашим новым целям.

```
<?PHP
registerLib("view","grid");

class Controller extends Grid {
    var $rec;
    function Controller() {
        $this->init();
        $this->rec = factory("model","model");
        return $this;
    }
    function getData() {
        global $db;

        // Восстановить последнее состояние грида
        if(!isset($_POST["call"]) AND !isset($_POST["offset"]) AND
        !isset($_POST["limit"]) AND !isset($_POST["filter_field"])) {
            $settings = $this->rec->getUserSettings("ecommerce");
            if($settings) list($this->Offset, $this->Limit, $this->Filter) = $settings;
        }

        // Узнать число записей таблицы отчета ecommerce
        $info = $db->one("SELECT count(*) as counter FROM
ecommerce");
        $this->Length = $info["counter"];
        if($this->Offset<0) $this->Offset=0;
        if($this->Offset>$this->Length) $this->Offset=0;
        if($this->Limit>$this->Length) $this->Limit = $this->Length;
        // Применить фильтры если пришло требование от пользователя
        $filter_str = "";
        if($this->Filter) $filter_str = " WHERE ".$this->Filter["field"]." LIKE '{ $this->Filter["value"] }%' ";
        // Построить SQL для выборки нашего отчета
        $sql = "SELECT * FROM ".LOGGINGTABLE." ".$filter_str."
".$this->OrderBy?" ORDER by ".$this->OrderBy["field"]." ".$this->OrderBy["direction"].":"). " LIMIT { $this->Offset }, { $this->Limit }";
```

```
$res = $db->all($sql);
$key = 0;
foreach ($res as $line) {
    foreach($this->Fields as $Index => $Field) {
        if($Field=="edesc") $line[$Field] =
lang($line[$Field]);
        $this->GridContent[$key][$Field]=$line[$Field];
    }
    $key++;
}

// Сохранить последнее состояние грида
$this->rec->putUserSettings("ecommerce", array($this->Offset,
$this->Limit, $this->Filter));

}
}

$ctrl = new Controller();
$ctrl->getData();
$json->Body = $ctrl->generateRespond();
?>
```

Теперь при переходе в наш интерфейс мы получим грид с данными из таблицы ecommerce.

Создание интерфейсов с управляемыми списками

В предыдущем примере мы рассмотрели интерфейс отчета, который содержит список, но не включает элементов управления записями списка. Следующим примером будет интерфейс управления почтовыми рассылками, имеющий кнопки добавления записей, кнопки для изменения свойств записей, кнопку удаления выбранных записей.

Аналогично первому примеру мы создаем в секции «Сервисы» (/app/admin/services/) папку maillist и размещаем в ней файлы index.layout.php, app.db, maillist.ctrl.php.

Шаблон интерфейса (maillist.tpl) будет помещен, соответственно, в папку /views/default/templates/. И его содержание будет следующим:

```
<sapi:include href="#subtemplates/header.tpl" />
<link rel="STYLESHEET" type="text/css" href="/views/default/css/grid.css" />
<script src="/views/default/js/json_grid.js" type="text/javascript"></script>

<script type="text/javascript">

var Schema_ShowItemModalWindow =
```

```
[{"ctrl":"admin/services/maillist/properties.model.php","title":"Свойства"}];

// Show modal window with document properties
// -- You should not change this function name. It's used in json_grid.js --
function showItemModalWindow(id){
    ModalWindowSaveAfterAction = "refresh grid & close";
    ModalWindowDataAssignment = "updateRecord";
    showModalWindow("admin/services/maillist/properties.model.php",
"Свойства", id, "admin/services/maillist/maillist.ctrl.php",
Schema_ShowItemModalWindow);
    return false;
}

// Show new modal window with document properties
function showNewItemModalWindow(parent_id){
    ModalWindowSaveAfterAction = "refresh grid & close";
    ModalWindowDataAssignment = "createRecord";
    showModalWindow("admin/services/maillist/properties.model.php",
"Добавить рассылку", parent_id, "admin/services/maillist/maillist.ctrl.php",
Schema_ShowItemModalWindow);
    return false;
}

// Processing when system gets data with AJAX
var processModalWindowContent= function(obj, empty) {
    if(respondStructure = responseAnalyze(obj)) {
        $("modal_window_body").innerHTML = respondStructure.Body;
    }
}
</script>

<body onload="initInterface()">
<sapi:include href="#subtemplates/toppanel.tpl" />

    <div class="smf_caption">&nbsp;</div>
    <sapi:apply name="ddc.#submenu.value">
        <sapi:param name="root">&args.0.value;</sapi:param>
    </sapi:apply>
    <div class="smf_docinfo">
    </div>

<sapi:include href="#subtemplates/middle.tpl"/>

    <div class="smf_place">

<!-- Tool panel -->
```

```
<sapi:apply name="ddc.#gridnav.value">
    <sapi:param name="grid_id">maillist</sapi:param>
    <sapi:param name="filters">
        <input class="smf_btn" type="button" name="add"
onclick="showNewItemModalWindow(false)" value="<sapi:apply exp="Назначить
рассылку" />" />&nbsp;
    </sapi:param>
</sapi:apply>

<script>
    EGConfig.sUrl = "admin/services/maillist/maillist.ctrl.php";
    EGConfig.ContextMenu = "on";
    EGConfig.ContextMovingMenu="on";
    settings(columns, '{"field":"sdate","title":"Дата рассылки",
"width":"60%"}');
    settings(columns, '{"field":"receivers","title":"Число получателей",
"width":"40%"}');
    makeRequest('');
</script>

</div>

<sapi:include href="#subtemplates/footer.tpl"/>
```

Как мы видим, здесь определены JS-структура Schema_ShowItemModalWindow и функции showItemModalWindow(), showNewItemModalWindow() и processModalWindowContent().

Schema_ShowItemModalWindow содержит адрес контроллера и заголовок кнопки свойств записей. В данном случае форму со свойствами записи формирует контроллер properties.model.php. Вы можете изучить его содержание в файле /app/admin/services/maillist/properties.model.php. Оно сводится к следующей схеме:

```
<?
    $env->Document->Data["itemattr_имяполя"] = 'значение';
    $output = parseXMLSapiens('
    <div class="smf_qc_area">
    <sapi:include href="#mlproperties.xml" parse="fieldset"
state="default" /></div>');
    $json->ActionCode = 1;
    $json->ErrorMsg = "";
    $json->Body = '""'.preg_replace("/[\r\n]/", "",
addslashes($output)).'""';
?>
```

В массив \$env->Document->Data помещаются сохраненные значения полей. Затем набор полей mlproperties.xml передается процессору XML Sapiens на обработку. Результат возвращается как содержание модального окна. Содержание набора полей mlproperties.xml может быть следующим:

```
<?xml version="1.0"?>
<sapi version="2.0" xmlns:sapi="http://www.xmlsapiens.org/spec/sapi.dtd">
<sapi:fieldset state="default" title="">
    <sapi:apply name="qc.itemattr_name.value" type="#inputtext"
title="Name" />
...
</sapi:fieldset>
</sapi>
```

Здесь попросту перечисляются все запросы формы свойств записи. Они будут содержать значения из \$env->Document->Data.

Функция showItemModalWindow() определяет, что будет происходить при нажатии кнопки «свойства» напротив записи. В данном случае будет открыто модальное окно, в которое направит форму контроллер properties.model.php. О том, как он ее сформирует, мы только что говорили. В заголовке окна будет "Свойства". Контроллеру окна будет передан ID записи и окно будет связано также с контроллером maillist.ctrl.php. Каким образом? В переменной ModalWindowSaveAfterAction задано что после сохранения изменений в модельном окне, связанный грид должен быть обновлен, а само модальное окно закрыто.

Функция showNewItemModalWindow() аналогичным образом описывает модальное окно для кнопки «Назначить рассылку» («Добавить запись»).

Далее мы видим, что в коде фильтра DDC gridnav помещена кнопка назначения рассылки (в общем случае это кнопка добавления записи в список) ей в событии onclick назначена описанная нами функция showNewItemModalWindow().

Еще одно изменение по отношению к шаблону неуправляемого списка – дополнительные параметры грида:

EGConfig.ContextMenu = "on"; - включение контекстного меню для списка (меню по нажатию правой кнопки мыши)

EGConfig.ContextMovingMenu="on"; - возможность изменения позиции для элементов списка посредством контекстного меню

Теперь мы получили список, но в нем по-прежнему нет кнопки свойств напротив записей. Это из-за того, что мы использовали в контроллере грида код из предыдущего примера. Давайте в методе getData изменим перечисление Foreach:

```
foreach($this->Fields as $Index => $Field) {
    if($Field=="actions") $line[$Field] =
        '<div style="margin-bottom: 3px;"><span class="grid_as">&nbsp;</span><a
href="#" class="grid_action"
onclick="showItemModalWindow('.$line["id"].')">'.lang('Properties').'/></a><span
class="grid_af">&nbsp;</span></div>';
    $this->GridContent[$key][$Field]=$line[$Field];
}
```

Теперь у нас есть кнопка добавления записи («Назначить рассылку»), кнопки свойств, но нет кнопки для удаления записей.

Как же получить возможность удаления элементов списка? Добавляем в конфигурацию списка в шаблон maillist.tpl

```
EGConfig.DeleteButton = true;
```

Теперь следует добавить и в контроллер maillist.ctrl.php методы обработки удаления записей и перемещения записей:

```
function liftRecord() {
    $info = $this->db->one("SELECT sort_id FROM ".MAILLISTTABLE."
WHERE id={$_POST["id"]}");
    $prev_info = $this->db->one("SELECT sort_id, id FROM
".MAILLISTTABLE." WHERE sort_id>{$info["sort_id"]} ORDER by sort_id ASC");
    $this->db->StartTransaction();
    $this->db->update("UPDATE ".MAILLISTTABLE." SET
sort_id={$prev_info["sort_id"]} WHERE id={$_POST["id"]}");
    $this->db->update("UPDATE ".MAILLISTTABLE." SET
sort_id={$info["sort_id"]} WHERE id={$prev_info["id"]}");
    $this->db->CompleteTransaction();
    return true;
}

function lowerRecord() {
    $info = $this->db->one("SELECT sort_id FROM ".MAILLISTTABLE."
WHERE id={$_POST["id"]}");
    $next_info = $this->db->one("SELECT sort_id, id FROM
".MAILLISTTABLE." WHERE sort_id<{$info["sort_id"]} ORDER by sort_id
DESC");
    $this->db->StartTransaction();
    $this->db->update("UPDATE ".MAILLISTTABLE." SET
sort_id={$next_info["sort_id"]} WHERE id={$_POST["id"]}");
    $this->db->update("UPDATE ".MAILLISTTABLE." SET
sort_id={$info["sort_id"]} WHERE id={$next_info["id"]}");
    $this->db->CompleteTransaction();
    return true;
}

function deleteSelected() {
```



```
        $this->db->StartTransaction();  
        foreach ($_POST as $field => $value) {  
            if(preg_match("/^EGRowDelete_/is", $field)) {  
                $rec_id = preg_replace("/^EGRowDelete_/is", "",  
$field);  
                $this->db->update("DELETE FROM ".MAILLISTTABLE."  
WHERE id={$rec_id}");  
            }  
        }  
        $this->db->CompleteTransaction();  
        return true;  
    }  
}
```

Можно себя поздравить – мы получили рабочий интерфейс с управляемым списком.

API платформы SAPID CMF

ADO API (Интерфейс управления БД)

AII()

Описание: получить данные выборки списка рядов из БД.

```
$db->all("..");
```

One()

Описание: получить массив данных заданной записи.

```
$db->one (".");
```

Update()

Описание: выполнить SQL запрос для модификации БД.

```
$db->update ("..");
```

lastNumRows()

Описание: получить число рядов, отвечающих условиям выборки.

```
$db->query("SELECT * FROM table")
$db->lastNumRows()
```

lastAffected()

Описание: получить число рядов, затронутых предыдущим SQL запросом.

```
$db->query("SELECT * FROM table")
$db->lastAffected()
```

lastInsertId()

Описание: получить ID последнего добавленного ряда.

```
$db->update("INSERT INTO table(id, text) VALUES('','text')");
$db->lastInsertId()
```

StartTransaction()/CompleteTransaction()

Описание: Указать точки начала и конца блока небезопасных операций над БД.

```
$db->StartTransaction()  
...  
$db->CompleteTransaction()
```

prepare()

Описание: выполнить экранирование строки, если это необходимо.

```
$db->prepare ("..");
```

DM API (интерфейс управления документами)

Для работы с интерфейсом следует инициализировать его класс:

```
$doc = new Doc($env);
```

Определение документа

Документ SAPID CMF – информационный объект, представленный в структурном дереве. Документы описываются атрибутами, содержанием и списком доступа. Набор полей содержания документа определяется шаблоном.

Атрибуты документа

doc_id	Идентификатор
name	Заголовок
type	Тип документа item, extlink, dalink, itlink
var	Переменная документа (элемент URL)
level	Уровень вложенности документа
site_id	Идентификатор сайта
cdate	Время создания документа
mdate	Время модификации документа
template	Файл шаблона
sort_id	Флаг сортировки
owner	Владелец
everyone	Права everyone
parent_id	Идентификатор родительского раздела


```
$PostData = array("name"=>"Item Example 1", "var"=>"item1",  
"parent_id"=>$env->SiteID);  
$id1 = $doc->add($PostData);
```

update()

Описание: изменение данных учетной записи документа.

Синтаксис: update(ID, array Attributs)

Возвращаемое значение: код успешной операции.

```
$PostData = array("name" => "New", "everyone"=>0);  
$doc->update(124, $PostData);
```

delete()

Описание: удаление документа.

Синтаксис: delete(int ID)

Возвращаемое значение: код успешной операции.

```
$doc->delete(102);
```

updateData()

Описание: изменение содержания документа.

Синтаксис: `updateData(ID, array Data)`

Возвращаемое значение: код успешной операции.

Входные параметры: ID, массив, содержащий последовательность "имя_QC"=>"значение"...

```
$PostData = array("title"=>"new", "message"=>"text");
$doc->updateData(124, $PostData);
```

copy()

Описание: копирование документа в новую позицию в структуре.

Синтаксис: copy(int ID, int TargetParentID)

Возвращаемое значение: код успешной операции.

```
$doc->copy(102, 43);
```

get()

Описание: получить атрибуты и содержание документа.

Синтаксис: `get(int ID, constant WITHOUTDATA/WITHDATA)`

Возвращаемое значение: массив данных.

```
$data = $doc->get(102);
```

или

```
$data = $doc->get(102, WITHOUTDATA);
```

getList()

Описание: получить список документов.

getSiteID()

Описание: получить ID первого ресурса/сайта информационной системы.

Синтаксис: `getSiteID()`

Возвращаемое значение: ID.

```
$ID = $doc->getSiteID();
```

identify()

Описание: определить выбранный по URL документ и принять его атрибуты и содержание.

Синтаксис: identify()

Возвращаемое значение: код успеха операции.

```
$doc->identify();
```

setPermission()

Описание: делегирование прав доступа к документу для конкретного пользователя или группы.

Синтаксис: setPermission(char(1) AccessPermissions, int ID, string User)

Возвращаемое значение: код успешной операции.

```
// Закрыть общий доступ
$PostData = array("unified_id" => $id11, "everyone"=>0);
$doc->update($PostData);
// Делегировать все права для пользователя Sheiko
$doc->setPermission(READING|MODIFYING|EXECUTING, $id11, "sheiko");
```


API (интерфейс управления записями)

Для работы с интерфейсом следует инициализировать его класс:

```
$rec = new Rec($env);
```

Определение записи

Запись SAPID CMF – информационный объект, представленный в линейном списке. Записи описываются атрибутами, содержанием и списком доступа. Набор полей содержания записи определяется XML-файлом fieldset.

Документ может включать в себя список записей. Каждая запись набора может, в свою очередь, включать в себя список записей.

Атрибуты записи

rec_id	Идентификатор
name	Заголовок
type	Тип записи (record или recset)
level	Уровень вложенности записи
site_id	Идентификатор сайта
cdate	Время создания записи
mdate	Время модификации записи
fieldset	Файл набора полей
sort_id	Флаг сортировки
owner	Владелец
everyone	Права everyone
parent_id	Идентификатор родительской записи
parent_type	Тип родительского элемента
document_id	Идентификатор родительского документа
parent_hierarchy	Список родительских элементов


```
$PostData = array("entry_name"=>"Modified Entry 1");
$rec->update(124, $PostData);
```

delete()

Описание: удаление записи.

Синтаксис: delete(int ID)

Возвращаемое значение: код успешной операции.

```
$rec->delete(1020);
```

updateData()

Описание: изменение содержания записи.

Синтаксис: `updateData(ID, array Data)`

Возвращаемое значение: код успешной операции

Входные параметры: ID, массив, содержащий последовательность "имя_QC"=>"значение"...

```
$PostData = array("title"=>"Title", "content"=>"Text");  
$res = $rec->updateData(124, $PostData);
```

get()

Описание: получить атрибуты и содержание записи.

Синтаксис: get(int ID, constant WITHOUTDATA/WITHDATA)

Возвращаемое значение: массив данных.

```
$data = $rec->get(1020, WITHOUTDATA);
```

getList()

Описание: получить список записей.

Синтаксис: `getList()`

См. CMS-приложение `get_infochannel()`

Возвращаемое значение: массив данных.

getByCondition()

Описание: получить атрибуты и содержание записи.

Синтаксис: `getByCondition(array/string Condition, constant WITHOUTDATA/WITHDATA)`

Возвращаемое значение: массив данных.

```
$data = $rec->getByCondition(array("url", "folder1/folder2/"),  
WITHOUTDATA);
```

getData()

Описание: получить содержание записи.

Синтаксис: `get(int ID)`

Возвращаемое значение: массив данных.

```
$data = $rec->getData(1020, WITHOUTDATA);
```

copy()

Описание: копировать запись в новую позицию.

Синтаксис: copy(int ID, int TargetParentID)

Возвращаемое значение: код успеха операции.

```
$rec->copy(1020, 34);
```

copyList()

Описание: копировать список записей в новую позицию.

Синтаксис: copyList(int ParentID, int TargetParentID)

Возвращаемое значение: код успеха операции.

```
$rec->copyList(33, 34);
```

addIndex()

Описание: добавить индекс в таблицу набора записей.

Синтаксис: addIndex(sting IndexFieldName, int ParentID, [string DBFieldType])

Возвращаемое значение: код успеха операции.

```
$rec->addIndex("price", 33, "VARCHAR( 32 )");
```

deleteIndex()

Описание: удалить индекс во всех таблицах.

Синтаксис: addIndex(sting IndexFieldName)

Возвращаемое значение: код успеха операции.

```
$rec->deleteIndex("price");
```

setPermission()

Описание: делегирование прав доступа к записи для конкретного пользователя или группы.

Синтаксис: setPermission(char(1) AccessPermissions, int ID, string User)

Возвращаемое значение: код успешной операции.

```
// Закрыть общий доступ
$PostData = array("everyone"=>0);
$rec->update(1020, $PostData);
// Делегировать все права для пользователя Sheiko
$rec->setPermission(READING|MODIFYING|EXECUTING, 1020, "pass");
```

backup()

Описание: создание резервной копии документа. При наличии более ранних резервных копий данной записи будет произведено смещение по списку в соответствии со значением констант DOCCOPIESMAXNUMBER и DOCCOPIESMAXNUMBERPERUSER, назначаемых в файле конфигурации.

Синтаксис: backup(int ID, int ParentID)

Возвращаемое значение: код успешной операции.

```
$res = $rec->backup(1020, 33);
```

УМ API (интерфейс управления пользователями)

Для работы с интерфейсом следует инициализировать его класс:

```
$uapi = new User($env);
```

get ()

Описание: получить атрибуты пользователя/группы.

Синтаксис: get (string User, constant WITHOUTDATA/WITHDATA)

Возвращаемое значение: массив.

```
$attrs = $uapi->get ("sheiko");
```

update ()

Описание: изменить атрибуты пользователя/группы.

Синтаксис: update (string User, array Data)

Возвращаемое значение: код успеха операции.

```
$uapi->update("sheiko", array("login"=>"newlogin");
```

delete ()

Описание: удалить пользователя/группу.

Синтаксис: delete(string User)

Возвращаемое значение: код успеха операции.

```
$uapi->delete("sheiko");
```

getData ()

Описание: получить данные пользователя/группы.

Синтаксис: getData(string User)

Возвращаемое значение: массив.

```
$data = $uapi->getData("sheiko");
```

updateData ()

Описание: изменить данные пользователя/группы.

Синтаксис: updateData(string User, array Data)

Возвращаемое значение: код успеха операции.

```
$uapi->updateData("sheiko", "description"=>"something");
```

deleteData ()

Описание: удалить данные пользователя/группы.

Синтаксис: deleteData(string User)

Возвращаемое значение: код успеха операции.

```
$uapi->deleteData("sheiko");
```

getByLogin()

Описание: получить информацию о пользователе по заданным логин/пароль.

Синтаксис: getByLogin(string Login, string Password)

Возвращаемое значение: массив.

```
$data = $uapi->getByLogin("trial","trial");
```

RD API (интерфейс адресов сервисов)

Для работы с интерфейсом следует инициализировать его класс:

```
$newURI = new InterfaceURI();
```

Пример назначения:

```
$newURI = new InterfaceURI();  
$newURI->clean();  
$newURI->set(51, "limit", "0,10");  
$newURI->create();
```

clean ()

Описание: очистить массив параметров.

Возвращаемое значение: нет.

```
$newURI->clean ();
```

set()

Описание: устанавливает значение параметра для заданного источника.

Возвращаемое значение: нет.

```
$newURI->set(51, "limit", "0-10");  
$newURI->set(51, "order", "FieldA DESC");  
$newURI->set(51, "filter", "FiledB >2 ");
```

create ()

Описание: будет сформирован URL, на базе указанных параметров.

Возвращаемое значение: URL.

```
$newURI->create ();
```

Справочник разработчика проекта

Переменные среды

Переменные среды доступны в шаблонах и DDC посредством вызова:

```
<sapi:apply name="переменная.value" />
```

или же:

```
&переменная.value;
```

Инструкции &переменная.value; обрабатываются процессором ранее прочих и могут быть использованы как параметры в выражений и вызовов XML Sapiens.

В платформе доступны следующие переменные:

args_length – число аргументов адресной строки (например, в этом случае /arg1/arg2/ переменная вернет 2);

extraargs_length – число вспомогательных аргументов адресной строки (например, в этом случае /arg1/arg2/data/extrarg1/ переменная вернет 1);

DirectAccessArgs_length – число аргументов при запросе записи (например, в этом случае /arg1/arg2/0000192/0000193/ переменная вернет 2);

document_url – адрес екущего документа (например, gallery/);

args.0 – первый аргумент адресной строки (например, gallery). Любой следующий аргумент доступен в переменных типа args.индекс_аргумента;

extraargs.0 – первый аргумент вспомогательной адресной строки (например, something). Любой следующий аргумент доступен в переменных типа extraargs.индекс_аргумента;

argsstring – экранированная строка аргументов адресной строки (например, gallery/);

argsstringwithoutslash – тоже, но без слешей (например, gallery);

admin_view_http_path – адрес корневой папки шаблонов административной панели (например, <http://sapidcmf.rh8.rg/views/default/>);

Parent – nameID родительского SELECT;

Display – none если данные не следует отображать при доставке (например, meta tags).

Следующие типы запросов:

inputtext – запрос строки;

article – запрос с WYSIWYG;

file – файл;

image – изображение;

date – дата;

select – выпадающий список;

checkbox – да/нет.

radio – да/нет.

Выражения

Выражения используются в условиях DDC, а также непосредственно в шаблонах посредством конструкции

A screenshot of a code editor with a light gray background. On the left, there is a vertical sidebar with a dark background and some faint, illegible text. The main area of the editor shows a single line of code: `<sapi:apply exp="выражение()" />`. The code is in a monospaced font, and the opening and closing quotes around the expression are double quotes.

Вы можете написать в plugin и использовать собственные выражения или воспользоваться выражениями, имеющимися в платформе:

showselectvalue() – значение индекса SELECT;

showthumb(var, width, height, alt, extra, extraparams) – создать/показать иконку;

showthumb_background – то же с задним фоном;

showimage(var,alt,extra) – показать изображение в списке;

enlarge(var,v1) – увеличить значение;

set(var,v1) – назначить значение переменной среды;

isequal(var,v1,v2) – если var верно, то v1;

&this.enum.list_length; – длина выборки;

&this.enum.pages_number; – число страниц;

&this.enum.range; – число записей на страницу;

&this.enum.page_next_href; – ссылка на следующую страницу;

&this.enum.page_prev_href; – ссылка на предыдущую страницу;

&this.enum.page_prev_number – номер предыдущей страницы;

&this.enum.page_next_number – номер следующей страницы;

&this.enum.page_first_href; – ссылка на первую страницу;

&this.enum.page_first_number; – номер первой страницы;

&this.enum.page_last_href; – ссылка на последнюю страницу;

&this.enum.page_last_number; – номер последней страницы;

&this.enum.page_first_range; – заголовок первой страницы типа 10-20;

&this.enum.page_last_range; – заголовок последней страницы типа 10-20.

get_tree()

Описание: возвращает документную структуру

```
<sapi:for-each select="get_tree()" name="enum" title="Get tree">
  <sapi:params>
    <sapi:param name="параметр_1">значение_1</sapi:param>
    <sapi:param name="параметр_N">значение_N</sapi:param>
  </sapi:params>
  <sapi:ifempty>Nothing found</sapi:ifempty>
  <sapi:fallback>CMS-application error</sapi:fallback>
```

Параметры:

limit {A,B} – ограничение числа выводимых узлов, где A – порядковый номер для начала списка (0..n), B – число выводимых узлов;

root {адрес_ветви} – указать адрес корневого документа для получения структуры ветви;

orderby {struct.fieldname ASC} - сортировать список;

withdata {yes} – принимать свойства и данные списка;

levellimitation {A-B} – ограничение по диапазону уровней;

Переменные перечисления:

this.lenght.value – число элементов перечисления;

this.this.id.value, this.this.doc_id.value – ID документа;

this.this.level.value – уровень вложенности;

this.this.type.value – тип документа (item/reset);

this.this.cdate.value – время создания в DATETIME;

this.this.cdatetime.value – время создания в UNIX TIMESTAMP;

this.this.mdate.value – время модификации в DATETIME;

this.this.href.value – адрес документа;

this.this.counter.value – счетчик записей;

this.this.branchlength.value – число элементов вложенной ветви;

this.this.resetlength.value – число элементов вложенного инфоканала;

this.this.document_id.value – ID документа, начинающего ветвь record set.

get_track()

Описание: возвращает маршрут к открытому документу



```
<sapi:for-each select="get_track()" name="enum" title="Get track">
```

Параметры:

нет.

Переменные перечисления:

this.this.item_id.value – ID документа;

this.this.item_level.value – уровень вложенности;

this.this.item_type.value – тип документа (item/reset);

this.this.var.value – переменная документа;

this.this.cdate.value – время создания в DATETIME;

Переменные перечисления:

`this.this.html_code.value` – HTML кода баннера;

`this.this.url.value` – адрес баннера;

`this.this.ad_id.value` – ID баннера

`this.this.banner.value` – файл баннера

`this.this.name.value` – название баннера

Приложения

Поддержка

Если Вы встретились с какими-либо проблемами при установке проекта или в ходе его эксплуатации, вы можете сообщить нам об этом.

Сайт сообщества: <http://www.sapid-club.com>