



HostCMS — удобство
управления сайтом в любой
точке мира.

Система управления сайтом HostCMS v. 5

Руководство по интеграции дизайна

Содержание

Содержание	2
Требования к квалификации пользователя HostCMS	4
<i>Требования к квалификации разработчиков сайтов на базе HostCMS</i>	4
Визуальный редактор	5
<i>Указание фона визуального редактора</i>	5
Подсветка синтаксиса	5
Верхняя панель клиентского раздела	6
<i>Отладочная информация</i>	6
<i>Просмотр XML в клиентском разделе</i>	7
<i>Просмотр SQL-запросов</i>	8
Основные принципы работы с XML/XSL	8
<i>Синтаксис XSL</i>	9
<i>Общая структура XSL-документа</i>	9
<i>Шаблоны <code>xsl:template match</code></i>	10
<i>Работа с атрибутами XML документа</i>	11
<i>Переменные (константы)</i>	12
<i>Сортировка <code>xsl:sort</code></i>	13
<i>Условный оператор <code>if</code></i>	14
<i>Инструкция <code>for-each</code></i>	16
<i>Инструкция <code>choose</code></i>	16
<i>Наиболее часто используемые XSLT-функции</i>	18
Пошаговое руководство по интеграции	19
<i>Структура макетов и шаблонов</i>	19
<i>Именованые XSL-шаблонов</i>	20
<i>Интеграция макета сайта</i>	20
Классы и объекты в HostCMS	21
Заголовка страницы	21
Описание страницы	21
Ключевые слова страницы	21
Кодировка страницы	21
Подключение CSS-стилей	22
Указание путей к изображениям и внешним файлам	22
Подключение файлов HostCMS в макете сайта	22
Общий вид блока <code><head></code> в макете сайта	23
Показ шаблона страницы в макете сайта	23
Вывод текущего года для строки Copyright	24
<i>Меню сайта</i>	24
Верхнее меню	24
Левое меню	26
<i>Интернет-магазин</i>	28
Список разделов каталога товаров	28
Горячие предложения	30
Краткая корзина	32
<i>Информационные системы</i>	34
Вывод анонсов новостей	34
Метки информационных систем	36
Поиск	38
<i>Форма поиска в макете сайта</i>	38
<i>Ограничение поиска по определенным источникам контента</i>	38
<i>Если ссылки из результатов поиска ведут на другой сайт</i>	39
Опросы	39
<i>Добавления опросов на сайт</i>	39
<i>Добавления блока опроса в макет сайта</i>	39
Структура сайта	41
<i>Работа с дополнительными свойствами структуры в XSL</i>	41
<i>Создание страницы для отображения ошибки 403/404/503</i>	42

Создание карты сайта.....	43
Информационные системы.....	44
Работа с дополнительными свойствами информационных элементов.....	44
Дополнительные свойства типа файл.....	44
Работа с дополнительными свойствами групп информационной системы.....	45
Внедрение информационной системы на сайт.....	45
Создание экспорта RSS 2.0 записей из информационных систем.....	47
Навигационная цепочка — «Хлебные крошки».....	49
Формы.....	52
Добавление формы на сайт.....	52
Внедрение формы в макет или шаблон страницы.....	53
Проблемы при отправке формы.....	53
Пользователи сайта.....	54
Публикация личного кабинета.....	54
Страница заказанных товаров.....	54
Страница восстановления пароля.....	54
Страница регистрации/редактирования анкетных данных.....	55
Страница почтовых рассылок, на которые подписан пользователь.....	55
Реклама.....	56
Размещение кода показа баннера на сайте.....	56
Защита от показа баннера поисковым ботам.....	56
Контекстный показ баннеров.....	56
Контекстный показ баннера в зависимости от содержания страницы.....	56
Контекстный показ баннера в зависимости от поискового запроса, по которому пользователь пришел на сайт.....	57
Создание страницы для учета нажатий на баннер.....	58
Интернет магазин.....	59
Работа с дополнительными свойствами товара в XSL.....	59
Работа с дополнительными свойствами группы товаров в XSL.....	59
Платежная система WebMoney.....	59
Платежная система RBK Money.....	61
Платежная система ASSIST.....	62
Автоматическое получение подтверждения оплаты.....	63
Платежная система Яндекс.Деньги.....	63
Прием платежей через ROBOKASSA.....	64
Обработчики платежных систем.....	65
Формирование страницы оплаты.....	68
Обработка уведомления платежной системы об оплате.....	69
Формирование страницы оплаты.....	70
Обработка уведомления платежной системы об оплате.....	71
Публикация интернет-магазина на сайте.....	72
Автоматическое построение прайс-листа для магазина.....	75
Вывод информации о продавцах.....	76
Сравнение товаров.....	76
Экспорт в Яндекс.Маркет.....	76
Кэширование.....	77
Управление блоками кэширования.....	77
Сохранение значения в кэше.....	77
Извлечение информации из кэша.....	78
Организация страниц для печати.....	79
Использование кода подтверждения для защиты от автоматического заполнения форм (Captcha).....	80
Импорт RSS-каналов.....	82
Восстановление пароля администратора в случае его утери.....	84
Очищение списка неудачных попыток авторизации.....	85
Предопределенные константы.....	85
Полезные константы для управления работой системы.....	86
Настройка файла main_classes.php.....	88

Требования к квалификации пользователя HostCMS

Пользователю системы управления для управления сайтом специальные знания не требуются, достаточно основных навыков работы с прикладным программным обеспечением.

Требования к квалификации разработчиков сайтов на базе HostCMS

Разработчику сайтов на базе системы управления HostCMS необходимо владеть базовыми знаниями HTML и PHP. Желательно обзорное знание XML/XSL технологий (более подробные требования к XSL-шаблонам размещены ниже), каждый модуль поставляется с демонстрационными шаблонами с комментариями в коде.

С внедрением в HostCMS 4.0 нового модуля «Типовые динамические страницы», требования к квалификации разработчиков сайтов на базе HostCMS значительно снижены.

Визуальный редактор

Система управления сайтом HostCMS для визуального редактирования информации использует редактор TinyMCE (подключаемый как внешний модуль), разработанный Moxiecode Systems AB и распространяемый под LGPL лицензией. Исходные коды визуального редактора TinyMCE размещены в директории /admin/wysiwyg/.

Информация с сайта производителя о редакторе (<http://tinymce.moxiecode.com/>):

TinyMCE is a platform independent web based Javascript HTML WYSIWYG editor control released as Open Source under LGPL by Moxiecode Systems AB. It has the ability to convert HTML TEXTAREA fields or other HTML elements to editor instances. TinyMCE is very easy to integrate into other CMS systems.

Указание фона визуального редактора

Визуальный редактор применяет фон и стиль текста из CSS-стилей макета. В некоторых случаях при особенностях верстки желательно указать цвет фона и стиль текста, для этого можно воспользоваться следующим стилем, который необходимо добавить в CSS для требуемого макета:

```
.mceContentBody {  
background: white;  
color: #000;  
}
```

Подсветка синтаксиса

Система управления сайтом HostCMS для подсветки редактируемого кода использует редактор CodePress (подключаемый как внешний модуль), распространяемый под LGPL лицензией.












Исходные коды CodePress размещены в директории /admin/js/codepress/.

Верхняя панель клиентского раздела

Панель выводится в клиентском разделе сайта для авторизованных в центре администрирования пользователей.




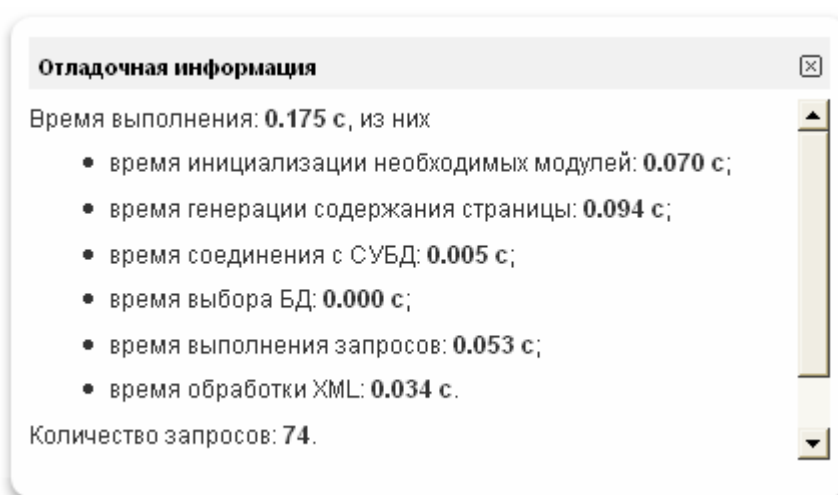
Верхняя панель позволяет выполнять быстрые действия из клиентского раздела и может содержать следующие кнопки:

-  редактировать структуру сайта;
-  редактировать макет;
-  редактировать шаблон страницы;
-  редактировать документ;
-  редактировать информационную систему;
-  редактировать магазин;
-  переход в раздел администрирования;
-  отладочная информация;
-  просмотр SQL-запросов;
-  отобразить XML;
-  выход из системы управления.

За отображение панели отвечает константа `ALLOW_PANEL`, для отключения панелей в клиентском разделе установите константе значение `false`.


Отладочная информация

Выводится с помощью кнопки «Отладочная информация»  верхней панели клиентского раздела и позволяет провести анализ общего быстродействия сайта и сервера, а также уровень интеграции.



Обратите внимание, время инициализации модулей, время выбора БД, время выполнения запросов и время обработки XML может частично или полностью входить во время генерации содержания страницы.

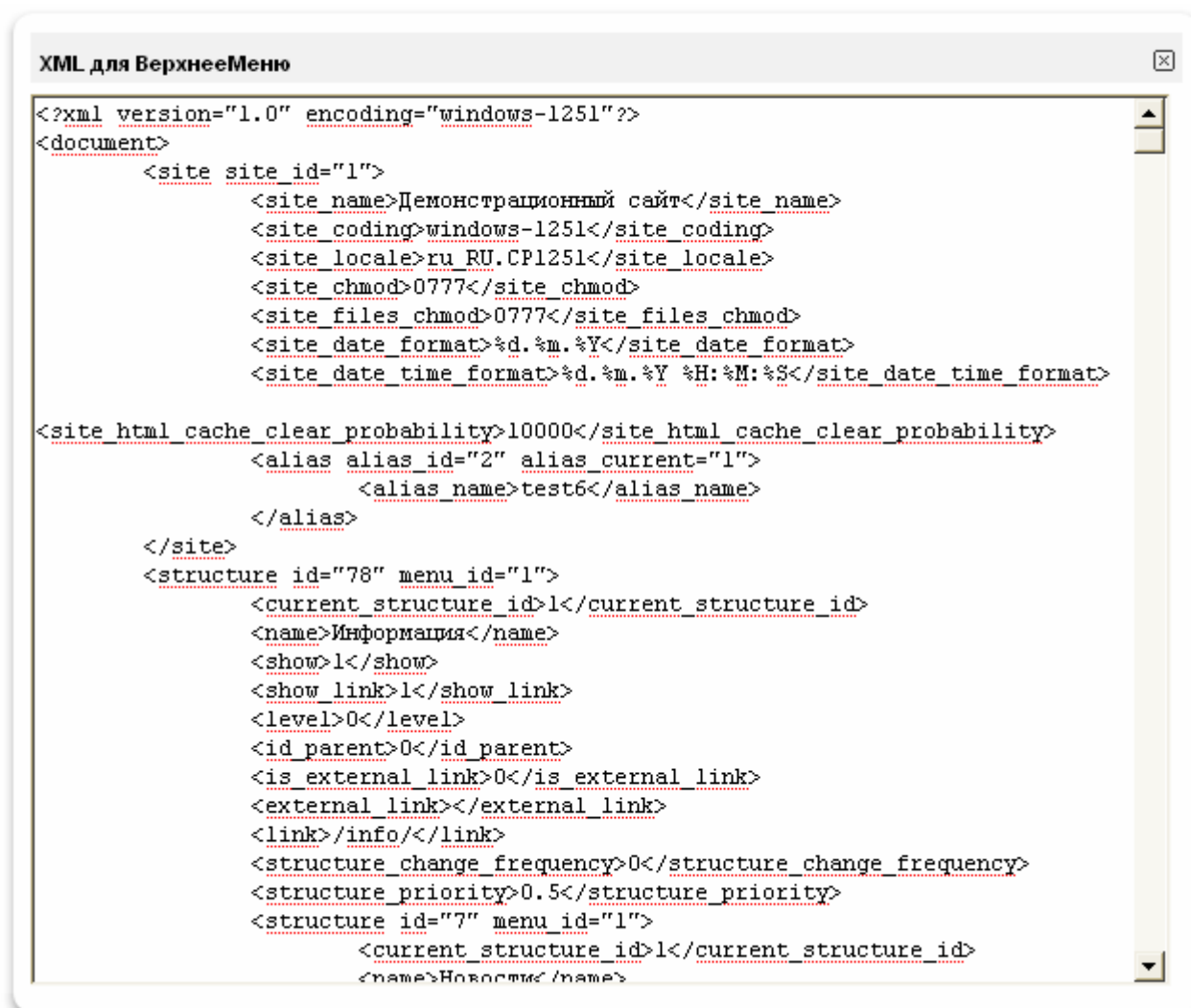
Просмотр XML в клиентском разделе

Для просмотра XML-данных, сгенерированных системой на странице сайта, необходимо в клиентском разделе на верхней панели нажать на кнопку  «Отобразить XML». Чтобы скрыть XML необходимо повторно нажать на кнопку на верхней панели.

В отчет по каждой генерации XML выводится наименование XSL-шаблона, с помощью которого было произведено XSLT-преобразование, ссылка на быстрое редактирование XSL-шаблона, ссылка на просмотр XML-данных, время обработки XML данных и размер XML в байтах:

XSL-шаблон  [ВерхнееМеню](#) 
 обработка  [XML](#)  0.009 с, размер XML 26 891 байт.


Пример окно с XML-данными:



```

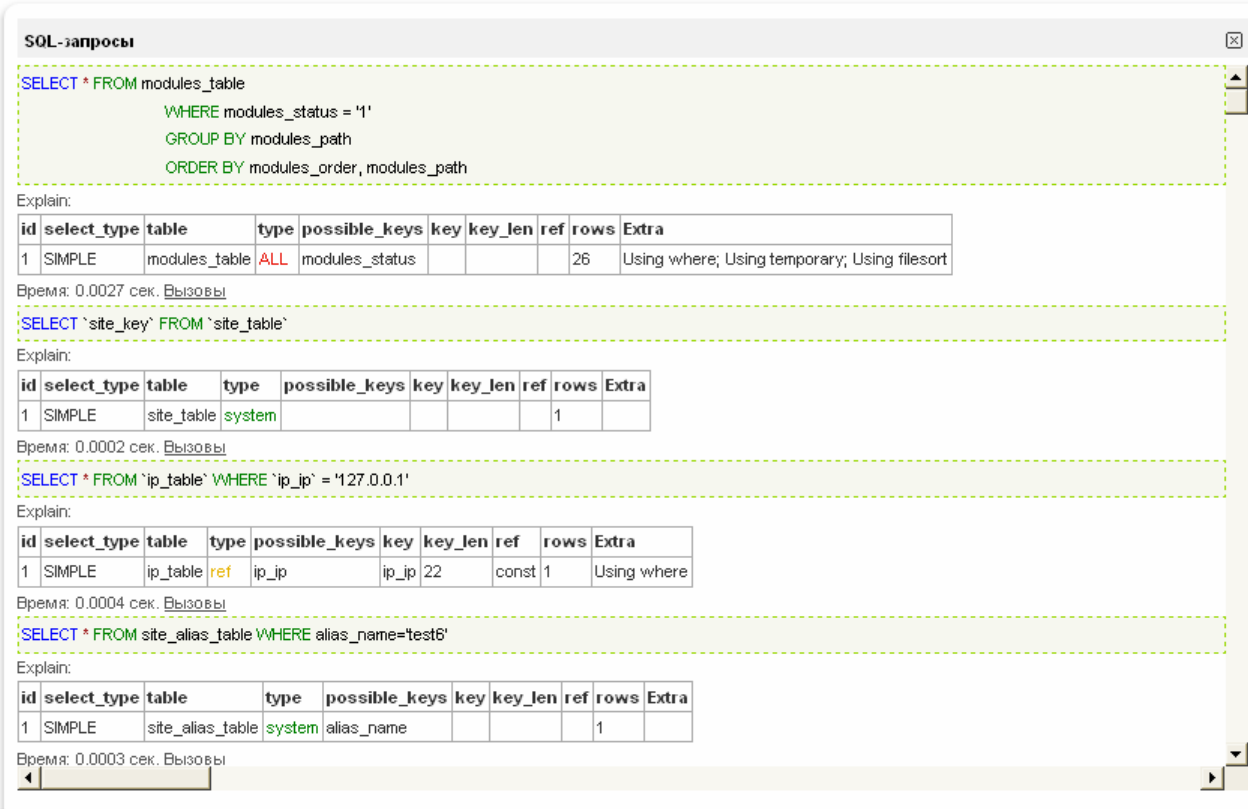
XML для ВерхнееМеню
<?xml version="1.0" encoding="windows-1251"?>
<document>
  <site site_id="1">
    <site_name>Демонстрационный сайт</site_name>
    <site_coding>windows-1251</site_coding>
    <site_locale>ru RU.CP1251</site_locale>
    <site_chmod>0777</site_chmod>
    <site_files_chmod>0777</site_files_chmod>
    <site_date_format>%d.%m.%Y</site_date_format>
    <site_date_time_format>%d.%m.%Y %H:%M:%S</site_date_time_format>
  </site>
  <site_html_cache_clear_probability>10000</site_html_cache_clear_probability>
  <alias alias_id="2" alias_current="1">
    <alias_name>test6</alias_name>
  </alias>
  </site>
  <structure id="78" menu_id="1">
    <current_structure_id>1</current_structure_id>
    <name>Информация</name>
    <show>1</show>
    <show_link>1</show_link>
    <level>0</level>
    <id_parent>0</id_parent>
    <is_external_link>0</is_external_link>
    <external_link></external_link>
    <link>/info</link>
    <structure_change_frequency>0</structure_change_frequency>
    <structure_priority>0.5</structure_priority>
    <structure id="7" menu_id="1">
      <current_structure_id>1</current_structure_id>
      <name>Новости</name>
    </structure>
  </structure>
  </document>
  
```

Просмотр SQL-запросов

Выводится с помощью кнопки «SQL-запросы»  верхней панели клиентского раздела и позволяет провести анализ SQL-запросов и их времени выполнения.

Кроме просмотра запросов возможен вывод анализа выполнения запроса.

Внимание! Включение вывода запросов и включение анализа запросов может увеличивать время генерации страницы, поэтому должно использоваться только на этапах интеграции сайта или отладки его работы.



The screenshot shows a window titled "SQL-запросы" with four query blocks, each containing a SQL query and its EXPLAIN output table.

Query 1:

```
SELECT * FROM modules_table
WHERE modules_status = '1'
GROUP BY modules_path
ORDER BY modules_order, modules_path
```

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	modules_table	ALL	modules_status				26	Using where; Using temporary; Using filesort

Время: 0.0027 сек. Вызовы

Query 2:

```
SELECT `site_key` FROM `site_table`
```

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	site_table	system					1	

Время: 0.0002 сек. Вызовы

Query 3:

```
SELECT * FROM `ip_table` WHERE `ip_ip` = '127.0.0.1'
```

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	ip_table	ref	ip_ip	ip_ip	22	const	1	Using where

Время: 0.0004 сек. Вызовы

Query 4:

```
SELECT * FROM site_alias_table WHERE alias_name='test6'
```

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	site_alias_table	system	alias_name				1	

Время: 0.0003 сек. Вызовы

За отображение кнопки просмотра SQL-запросов отвечает константа *ALLOW_SHOW_SQL*, для отображения кнопки в клиентском разделе установите константе значение *true*, за отображение анализа запросов с помощью EXPLAIN отвечает константа *ALLOW_EXPLAIN*, для отображения анализа запросов установите константе значение *true*.

Основные принципы работы с XML/XSL

Использование стилевых таблиц (XSL) позволяет обеспечить независимое от конкретного устройства вывода отображение XML-документов.

Использование XSL-шаблонов с одними и теми же XML-данными позволяет получать различные результаты, например, метод отображения структуры сайта с XSL-шаблоном "Меню" отобразит нужное меню, этот же метод отображения структуры с XSL-шаблоном "Карта сайта" отобразит полную карту сайта.

Синтаксис XSL

В общем случае содержание XSL-документа должно удовлетворять определенным требованиям:

- в заголовке документа помещается объявление XML, в котором указывается язык разметки документа, номер его версии и дополнительная информация:

```
<?xml version="1.0" encoding="windows-1251"?>
```

- каждый открывающий тэг, определяющий некоторую область данных в документе обязательно должен иметь закрывающий тэг;

Правильно:

```
<p>Абзац</p>

```

Неправильно:

```
<p> – непарный тег, нет закрывающего тега, либо отсутствует слэш в конце тега
 – Непарный тэг, отсутствует слэш в конце тега
```

- в XML учитывается регистр символов;
- все значения атрибутов, используемых в определении тэгов, должны быть заключены в кавычки:

Правильно:

```

```

Неправильно:

```
<img src=/hostcms/image1.gif alt="" border="0"/> - Значение параметра src не взято в кавычки
```

- вложенность тэгов в XML строго контролируется, поэтому необходимо следить за порядком следования открывающих и закрывающих тэгов:

Правильно:

```
<strong>Жирный текст <i>Текст курсивом</i></strong>
```

Неправильно:

```
<strong>Жирный текст <i>Текст курсивом</strong></i>
```

- вся информация, располагающаяся между начальным и конечным тэгами, рассматривается в XML как данные, и поэтому учитываются все символы форматирования;
- Символ неразрывного пробела * *; необходимо записывать как * * или * *;

Общая структура XSL-документа

В самом начале документа указывается стилистика XML и импорт пространства имен XML:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
...
```

```
</xsl:stylesheet>
```

Шаблоны xsl:template match

Основным элементом для оформления является xsl:template match.

xsl:template match является основным для оформления XML дерева и вызывается каждый раз при совпадении имени узла XML.

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<!-- Шаблон для корневого узла -->
<xsl:template match="/">
  <!--Обработка данного узла -->
</xsl:template>
</xsl:stylesheet>
```

Рассмотрим простейший XML документ:

```
<document>
<title>Заголовок 1</title>
<structure id="1">
<value>aaa</value>
</structure>
<structure id="2">
<value>bbb</value>
</structure>
</document>
```

Теперь напишем простой XSL-шаблон для форматированного отображения XML данных:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/document">
<html>
<head>
<title><xsl:value-of select="title"/></title>
</head>
<body>
  <!-- Выбираем узлы structure -->
  <xsl:apply-templates select="structure" />
</body>
</html>
</xsl:template>
<!-- Шаблон для отображения узлов /document/structure -->
<xsl:template match="structure">
  <p>Значение: <xsl:value-of disable-output-escaping="yes" select="value"/></p>
</xsl:template>
</xsl:stylesheet>
```

В результате получится HTML-документ:

```
<html>
<head>
<title>Заголовок 1</title>
</head>
<body>
<p>Значение: aaa</p>
<p>Значение: bbb</p>
</body>
</html>
```

Работа с атрибутами XML документа

Получение значений атрибутов производится через конструкцию конструкция @[имя_атрибута], например @id возвращает значение атрибута id.

Пример XSL-шаблона для XML из предыдущего пункта:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/document">
<html>
<head>
<title><xsl:value-of select="title"/></title>
</head>

<body>
      <!-- Выбираем узлы structure -->
      <xsl:apply-templates select="structure" />
</body>

</html>
</xsl:template>

<!-- Шаблон для отображения узлов /document/structure -->
<xsl:template match="structure">
      <p>Значение: <xsl:value-of disable-output-escaping="yes" select="value"/></p>
      <p>Значение параметра ID: <xsl:value-of select="@id"/></p>
</xsl:template>

</xsl:stylesheet>
```

Результирующий HTML-документ:

```
<html>
<head>
<title>Заголовок 1</title>
</head>
<body>
<p>Значение: aaa</p>
<p>Значение параметра ID: 1</p>
<p>Значение: bbb</p>
<p>Значение параметра ID: 2</p>
</body>
</html>
```

Переменные (константы)

Переменные в XSLT фактически являются константами.

Определение переменных осуществляется через `xsl:variable`, например:

```
<xsl:variable name="var1">значение</xsl:variable>
```

Обращение к переменной осуществляется через `$имя_переменной`.

Прототип:

```
<!-- Category: top-level-element -->
<!-- Category: instruction -->
<xsl:variable
  name = qname
  select = expression>
  <!-- Content: template -->
</xsl:variable>
```

Рассмотрим простейший XML документ:

```
<document>
<title>Заголовок 1</title>
<structure id="1">
<value>aaa</value>
</structure>
<structure id="2">
<value>bbb</value>
</structure>
</document>
```

XSL-шаблон для форматированного отображения XML данных:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/document">
<html>
<head>
<title><xsl:value-of select="title"/></title>
</head>
<body>
      <!-- Выбираем узлы structure -->
      <xsl:apply-templates select="structure" />
</body>
</html>
</xsl:template>
<!-- Шаблон для отображения узлов /document/structure -->
<xsl:template match="structure">
```

```

    <!-- Определяем переменную с адресом -->
    <xsl:variable name="path">http://localhost/</xsl:variable>

    <p>
    <a href="{ $path }{@id}/">Значение: <xsl:value-of disable-output-escaping="yes"
select="value"/></a>
    </p>

    <p>Значение параметра ID: <xsl:value-of disable-output-escaping="yes" select="@id"/></p>
</xsl:template>
</xsl:stylesheet>

```

В результате получится HTML-документ:

```

<html>
<head>
<title>Заголовок 1</title>
</head>
<body>
<p><a href="http://localhost/1/">Значение: aaa</a></p>
<p>Значение параметра ID: 1</p>
<p><a href="http://localhost/2/">Значение: bbb</a></p>
<p>Значение параметра ID: 2</p>
</body>
</html>

```

Переменной можно присвоить значение узла или значение атрибута. Пример присвоения значения узла value:

```
<xsl:variable name="value1" select="value"/>
```

Пример присвоения атрибута id узла value:

```
<xsl:variable name="value1" select="value/@id"/>
```

Использование переменной:

```
<xsl:value-of disable-output-escaping="yes" select="$value1"/>
```

Сортировка xsl:sort

Сортировка XML-тегов в XSLT выполняется с использованием элемента `<xsl:sort select=" attribute">`

Этот элемент должен размещаться внутри `xsl:apply-templates` или `xsl:for-each`. Сортировка может выполняться как по самим xml-тегам, так и по их атрибутам, порядок сортировки можно задавать по возрастанию или по убыванию.

Прототип:

```
<xsl:sort
  select = string-expression
  lang = { nmtoken }
  data-type = { "text" | "number" | QName-but-not-ncname }
  order = { "ascending" | "descending" }
  case-order = { "upper-first" | "lower-first" } />
```

XSL-шаблон для форматированного отображения XML данных (сортировка по параметру):

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/document">
<html>
<head>
<title><xsl:value-of select="title"/></title>
</head>

<body>
  <!-- Выбираем узлы structure -->
  <xsl:apply-templates select="structure">
    <xsl:sort select="@id" order="descending"/>
  </xsl:apply-templates>
</body>
</html>
</xsl:template>

<!-- Шаблон для отображения узлов /document/structure -->
<xsl:template match="structure">
  <p>Значение: <xsl:value-of disable-output-escaping="yes" select="value"/></p>
</xsl:template>

</xsl:stylesheet>
```

Сортировка по значению тега

```
<xsl:apply-templates select="structure">
  <xsl:sort select="value" order="descending"/>
</xsl:apply-templates>
```

В результате получится HTML-документ:

```
<html>
<head>
<title>Заголовок 1</title>
</head>
<body>
<p>Значение: bbb</p>
<p>Значение: aaa</p>
</body>
</html>
```

Условный оператор if

Элемент `xsl:if` используется для управления формированием результатов работы шаблонов.

Прототип:

```

<!-- Category: instruction -->
<xsl:if test = boolean-expression>
  <!-- Content: template -->
</xsl:if>

```

XSL-шаблон:

```

<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/document">
<html>
<head>
<title><xsl:value-of select="title"/></title>
</head>

<body>
  <!-- Выбираем узлы structure -->
  <xsl:apply-templates select="structure">
    <xsl:sort select="value" order="descending"/>
  </xsl:apply-templates>
</body>
</html>
</xsl:template>

<!-- Шаблон для отображения узлов /document/structure -->
<xsl:template match="structure">

  <p>
  <!-- Если значение равно 1 - выводим слово "Значение красным" -->
  <xsl:if test="@id=1">
  <font color="red">Значение:</font>
  </xsl:if>

  <!-- Если значение не равно 1-->
  <xsl:if test="not(@id=1)">
  Значение:
  </xsl:if>

  <xsl:value-of disable-output-escaping="yes" select="value"/>
  </p>
</xsl:template>

</xsl:stylesheet>

```

В результате получится HTML-документ:

```

<html>
<head>
<META http-equiv="Content-Type" content="text/html; charset=UTF-16">
<title>Заголовок 1</title>
</head>
<body>
<p>
  Значение:
  bbb</p>
<p>
<font color="red">Значение:</font>aaa</p>
</body>
</html>

```

Инструкция for-each

Вывод можно осуществлять как через `xsl:apply-templates` и `xsl:template match`, так и через `xsl:for-each`

Прототип:

```
<!-- Category: instruction -->
<xsl:for-each
  select = node-set-expression>
  <!-- Content: (xsl:sort*, template) -->
</xsl:for-each>
```

XSL-шаблон:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/document">
<html>
<head>
<title><xsl:value-of select="title"/></title>
</head>

<body>
  <!-- Выбираем узлы structure -->
  <xsl:for-each select="structure">
    <p>Значение: <xsl:value-of disable-output-escaping="yes" select="value"/></p>
  </xsl:for-each>

</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

В результате получится HTML-документ:

```
<html>
<head>
<META http-equiv="Content-Type" content="text/html; charset=UTF-16">
<title>Заголовок 1</title>
</head>
<body>
<p>Значение: aaa</p>
<p>Значение: bbb</p>
</body>
</html>
```

Инструкция choose

Среди нескольких возможных альтернатив элемент `xsl:choose` выбирает одну. Он состоит из последовательности элементов `xsl:when`, за которой следует необязательный элемент `xsl:otherwise`. Каждый элемент `xsl:when` имеет единственный атрибут `test`, который задает некое выражение. Содержимое элементов `xsl:when` и `xsl:otherwise` является шаблоном.

Если обрабатывается элемент `xsl:choose`, поочередно проверяются все элементы `xsl:when`. При этом обрабатывается соответствующее выражение, а полученный объект преобразуется в булевый тип как при вызове функции `boolean`. Обрабатывается содержимое первого, и только первого элемента `xsl:when`, при проверке которого было получено `true`. Если ни один из `xsl:when` не показал `true`,

подставляется значение элемента `xsl:otherwise`. Если ни один из `xsl:when` не показал `true`, а элемент `xsl:otherwise` отсутствует, то ничего не создается.

Первым должен идти элемент `xsl:choose`, а за ним дополнительные (один или несколько) элементы `xsl:when`, если требуется обрабатывать значение не подпадающее ни под одно из условий имеющихся элементов `xsl:when`, то вы можете добавить элемент `xsl:otherwise`.

Прототип:

```
<!-- Category: instruction -->
<xsl:choose>
  <!-- Content: (xsl:when+, xsl:otherwise?) -->
</xsl:choose>
<xsl:when
  test = boolean-expression>
  <!-- Content: template -->
</xsl:when>
<xsl:otherwise>
  <!-- Content: template -->
</xsl:otherwise>
```

XSL-шаблон:

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/document">
<html>
<head>
<title><xsl:value-of select="title"/></title>
</head>
<body>
  <!-- Выбираем узлы structure -->
  <xsl:for-each select="structure">

    <p>
      <!-- Проверка на параметр id -->
      <xsl:choose>
        <xsl:when test="@id = 1">
          <font color="red">Значение:</font>
        </xsl:when>
        <xsl:when test="@id > 1">
          Значение:
        </xsl:when>
        <xsl:otherwise>
          Значение:
        </xsl:otherwise>
      </xsl:choose>

        <xsl:value-of disable-output-escaping="yes" select="value"/>

    </p>
  </xsl:for-each>
</body>
</html>
</xsl:template>
</xsl:stylesheet>
```

В результате получится HTML-документ:

```

<html>
<head>
<META http-equiv="Content-Type" content="text/html; charset=UTF-16">
<title>Заголовок 1</title>
</head>
<body>
<p><font color="red">Значение:</font> aaa</p>
<p>Значение: bbb</p>
</body>
</html>

```

Наиболее часто используемые XSLT-функции

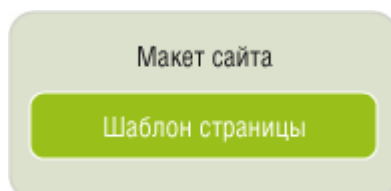
- boolean **boolean**(object) — Явным образом преобразует объект, который ей передается в булевый тип
- boolean **not**(boolean) — Выполняет логическое отрицание
- boolean **true**() — Возвращает «истину»
- boolean **false**() — Возвращает false, «ложь»
- boolean **lang**(string) — Возвращает истину, если идентификатор языка, который передан ей в виде строкового параметра, соответствует языковому контексту контекстного узла
- number **number**(object) — Явным образом конвертирует свой аргумент в числовой тип. Если аргумент опущен, то выполняется с множеством, состоящим из контекстного узла
- number **sum**(node-set) — Суммирует значения узлов из переданного ей множества
- number **floor**(number) — Округляет аргумент до ближайшего не большего целого
- number **ceiling**(number) — Округляет аргумент до ближайшего не меньшего целого
- number **round**(number) — Округляет аргумент до ближайшего целого значения
- string **string**(object) — Преобразует аргумент к строковому типу явным образом. Если аргумент опущен, то выполняется с множеством, состоящим из контекстного узла
- string **concat**(string, string, ...) — Возвращает конкатенацию аргументов
- boolean **starts-with**(string, string) — Принимает на вход два строковых аргумента и возвращает true, если первая строка начинается со второй и false в противном случае
- boolean **contains**(string, string) — Принимает на вход два строковых аргумента и возвращает true, если первая строка содержит вторую и false в противном случае
- string **substring-before**(string, string) — Принимает на вход два строковых аргумента, находит в первой строке вторую и возвращает подстроку, которая ей предшествует
- string **substring-after**(string, string) — Принимает на вход два строковых аргумента, находит в первой строке вторую и возвращает подстроку, которая за ней следует
- string **substring**(string, number, number) — Возвращает подстроку переданного ей строкового аргумента, которая начинается с позиции, указанной вторым аргументом и длиной, указанной третьим аргументом. Если третий аргумент не указан, то подстрока продолжается до конца строки
- number **string-length**(string) — Возвращает число символов строкового аргумента
- string **normalize-space**(string) — Производит со строковым аргументом нормализацию пробельного пространства. Если аргумент опущен, выполняется со строковым значением контекстного узла
- string **translate**(string, string, string) — Производит замену символов первого своего строкового аргумента, которые присутствуют во втором аргументе на соответствующие символы третьего аргумента
- number **last**() — Возвращает размер контекста вычисления выражения
- number **position** () — Возвращает позицию контекста вычисления выражения
- number **count**(node-set) — Возвращает число узлов, которое входит во множество, переданное ей в качестве аргумента
- string **local-name**(node-set) — Возвращает локальную часть имени первого в порядке просмотра документа узла множества, переданного в качестве аргумента или локальную часть имени контекстного узла, если аргумент отсутствует. Если аргумент опущен, то выполняется с множеством, состоящим из контекстного узла

- string **namespace-uri**(node-set) — Возвращает URI пространства имен первого в порядке просмотра документа узла множества, переданного в качестве аргумента или локальную часть имени контекстного узла, если аргумент отсутствует. Если аргумент опущен, то выполняется с множеством, состоящим из контекстного узла
- string **name**(node-set) — Возвращает в виде префикс:имя расширенное имя локальную часть имени первого в порядке просмотра документа узла множества, переданного в качестве аргумента или локальную часть имени контекстного узла, если аргумент отсутствует. Если аргумент опущен, то выполняется с множеством, состоящим из контекстного узла
- node-set **id**(object) — Возвращает множество узлов по уникальным идентификаторам
- node-set **key**(string, object) — По данному имени и значению ключа возвращает множество узлов, которые им обладают
- node-set **document**(object, node-set) — Позволяет обращаться к внешним документам по заданным URI. Первый узел необязательного параметра node-set принимается за точку отсчета для относительных URI
- node-set **current**() — Возвращает текущий узел преобразования
- string **generate-id**(node-set) — Возвращает уникальный строковый идентификатор первого узла переданного множества или контекстного узла, если аргумент опущен
- object **system-property**(string) — Возвращает значение свойства, имя которого передано как аргумент

Пошаговое руководство по интеграции

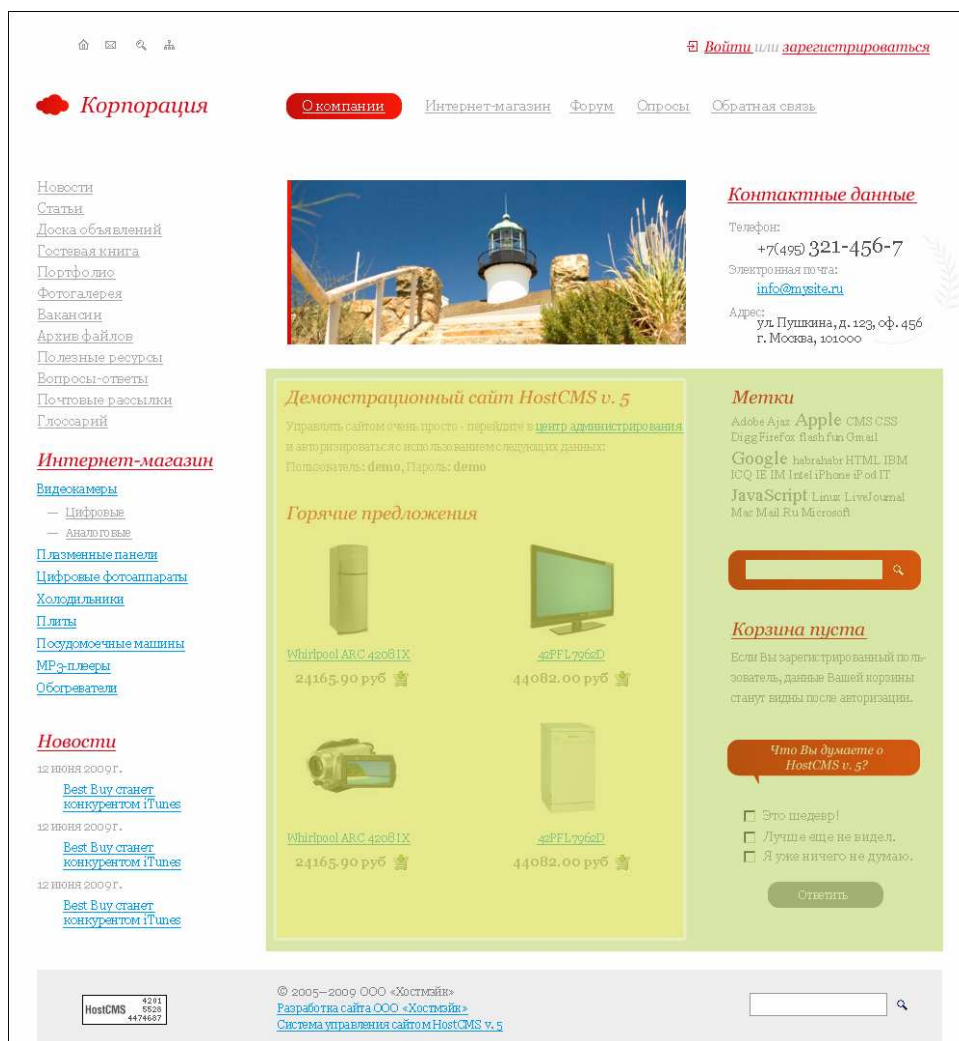
Структура макетов и шаблонов

Общее оформление страницы сайта размещается в макетах и шаблонах сайта. Шаблоны сайта включаются в макет сайта по принципу матрешки.



В макет сайта выделяется общая область, неизменная для большинства страниц сайта, а в шаблоны страниц выносятся отличающиеся фрагменты макета.

Рассмотрим пример выделения областей в макет сайта и шаблон страницы:



Белый блок — макет сайта, зеленый — шаблон страницы, желтый — область для вывода контента страницы сайта.

Именованние XSL-шаблонов

За формат отображения динамичных данных в HostCMS отвечают XSL-шаблоны, которые обрабатывают генерируемые системой управления XML-данные, в результате чего получается HTML-код.

В системе управления каждый XSL-шаблон имеет название, которое кратко характеризует его функционал, например «ВерхнееМеню».

Администратор может использовать и модифицировать поставляемые вместе с системой управления XSL-шаблоны или разрабатывать собственные.

Интеграция макета сайта

Выделяем область, которая будет вынесена в макет, сохраняем ее в макете сайта. CSS-стили сохраняются в соответствующем поле макета сайта в центре администрирования (см. руководство пользователя).

Редактирование макетов сайта и шаблонов страниц осуществляется исключительно через центр администрирования.

В исходном макете производится замена областей на вызовы различных методов HostCMS.

Классы и объекты в HostCMS

Каждый модуль имеет один или несколько классов, каждый класс в свою очередь имеет методы, которые используются в системе управления.

Описание методов дано в API системы управления, ознакомиться с которым можно на странице <http://www.hostcms.ru/documentation/>, в настоящем руководстве дана ознакомительная информация о методах системы управления без указания дополнительных параметров. С полным списком параметров и подробным описанием метода с примерами можно ознакомиться в API.

Чтобы воспользоваться методом определенного класса, необходимо создать объект этого класса. Создание объекта класса допустимо двумя способами:

1. С использованием шаблона программирования singleton, который позволяет создавать только один объект заданного класса, экономя используемую память:
`$Structure = & singleton('Structure');`
2. С использованием стандартного создания объекта:
`$Structure = new Structure();`

Объект `$kernel` всегда доступен в макете и шаблоне, поэтому не требует дополнительного создания.

Заголовок страницы

Метод `$kernel->show_title()` выводит заголовок страницы.

```
<title><?php $kernel->show_title()?></title>
```

Описание страницы

Метод `$kernel->show_description()` выводит описание страницы.

```
<meta name="description" content="<?php $kernel->show_description()?>"></meta>
```

Ключевые слова страницы

Метод `$kernel->show_keywords()` выводит ключевые слова, заданные для страницы.

```
<meta name="keywords" content="<?php $kernel->show_keywords()?>"></meta>
```

Кодировка страницы

```
<meta content="text/html; charset=<?php echo SITE_CODING?>" http-equiv="Content-Type"></meta>
```

Кодировка задается в атрибутах каждого сайта.

Подключение CSS-стилей

Метод `$kernel->show_CSS()` – выводит подключение CSS-стилей в макет. Необязательный параметр указывает способ подключения CSS-стиля и по умолчанию имеет значение `true` (ссылка на CSS-стиль), для вывода кода CSS-стилей в макет указывается `false`.

```
<?php $kernel->show_CSS()?>
```

Указание путей к изображениям и внешним файлам

Наиболее часто пользователи сталкиваются с проблемой искажения макета после вставки его в систему. Причиной такого поведения является некорректное указание путей к изображениям.

Пути необходимо указывать с ведущим слэшем (абсолютный путь относительно корня), например:

- `images/my.gif` – неправильно;
- `/images/my.gif` – правильно.

Подключение файлов HostCMS в макете сайта

Файлы, содержащие JavaScript инструкции:

```
<script type="text/javascript" src="/templates/template1/hostcms.js"></script>
<script type="text/javascript" src="/hostcmsfiles/ajax/JsHttpRequest.js"></script>
<script type="text/javascript" src="/hostcmsfiles/ajax/ajax.js"></script>
<script type="text/javascript" src="/hostcmsfiles/main.js"></script>
```

Системные стили HostCMS:

```
<link rel="stylesheet" type="text/css" href="/hostcmsfiles/style.css" />
```

Ссылка на RSS-источник для информационной системы «Новости»:

```
<link rel="alternate" type="application/rss+xml" title="HostCMS RSS Feed" href="/news/rss/" />
```

Предварительная загрузка изображения для AJAX-прелоадера:

```
<!-- Предварительная загрузка изображений -->
<script type="text/javascript" language="JavaScript">
var image1 = new Image(); image1.src = "/hostcmsfiles/images/shadow-b.png";
var image2 = new Image(); image2.src = "/hostcmsfiles/images/shadow-l.png";
var image3 = new Image(); image3.src = "/hostcmsfiles/images/shadow-lb.png";
var image4 = new Image(); image4.src = "/hostcmsfiles/images/shadow-lt.png";
var image5 = new Image(); image5.src = "/hostcmsfiles/images/shadow-r.png";
var image6 = new Image(); image6.src = "/hostcmsfiles/images/shadow-rb.png";
var image7 = new Image(); image7.src = "/hostcmsfiles/images/shadow-rt.png";
var image8 = new Image(); image8.src = "/hostcmsfiles/images/shadow-t.png";
var image9 = new Image(); image9.src = "/hostcmsfiles/images/ajax_loader.gif";

/* Изображения для верхнего меню */
var image10 = new Image(); image10.src = "/images/red_grad.gif";
var image11 = new Image(); image11.src = "/images/top_menu_1.gif";
```

```
var image12 = new Image(); image12.src = "/images/top_menu_r.gif";
</script>
```

Общий вид блока <head> в макете сайта

```
<head>
  <title><?php $kernel->show_title()?></title>
  <meta name="description" content="<?php $kernel->show_description()?>"></meta>
  <meta name="keywords" content="<?php $kernel->show_keywords()?>"></meta>
  <meta http-equiv="X-UA-Compatible" content="IE=EmulateIE7" />
  <meta content="text/html; charset=<?php echo SITE_CODING?>" http-equiv="Content-
Type"></meta>

  <?php $kernel->show_CSS()?>

  <script type="text/javascript" src="/templates/template1/hostcms.js"></script>
  <script type="text/javascript" src="/hostcmsfiles/ajax/JsHttpRequest.js"></script>
  <script type="text/javascript" src="/hostcmsfiles/ajax/ajax.js"></script>
  <script type="text/javascript" src="/hostcmsfiles/main.js"></script>

  <link rel="stylesheet" type="text/css" href="/hostcmsfiles/style.css" />

  <link rel="alternate" type="application/rss+xml" title="HostCMS RSS Feed"
href="/news/rss/" />

  <!-- Предварительная загрузка изображений -->
  <script type="text/javascript" language="JavaScript">
var image1 = new Image(); image1.src = "/hostcmsfiles/images/shadow-b.png";
var image2 = new Image(); image2.src = "/hostcmsfiles/images/shadow-l.png";
var image3 = new Image(); image3.src = "/hostcmsfiles/images/shadow-lb.png";
var image4 = new Image(); image4.src = "/hostcmsfiles/images/shadow-lt.png";
var image5 = new Image(); image5.src = "/hostcmsfiles/images/shadow-r.png";
var image6 = new Image(); image6.src = "/hostcmsfiles/images/shadow-rb.png";
var image7 = new Image(); image7.src = "/hostcmsfiles/images/shadow-rt.png";
var image8 = new Image(); image8.src = "/hostcmsfiles/images/shadow-t.png";
var image9 = new Image(); image9.src = "/hostcmsfiles/images/ajax_loader.gif";

/* Изображения для верхнего меню */
var image10 = new Image(); image10.src = "/images/red_grad.gif";
var image11 = new Image(); image11.src = "/images/top_menu_l.gif";
var image12 = new Image(); image12.src = "/images/top_menu_r.gif";
  </script>
</head>
```

Показ шаблона страницы в макете сайта

В макете сайта в нужное место вносится код вызова шаблона страницы. Шаблон страницы задается для каждого раздела сайта из выпадающего списка в структуре сайта.

```
<!-- Вызов шаблона для текущей страницы -->
<?php
$kernel->show_current_template();
?>
```

Код макета и шаблона страниц состоит из HTML-кода, PHP-кода и вызова API функций системы.

Система управления размещает макеты и шаблоны страниц в файловой системе сервере — макеты размещаются в директории `/templates/template{id_макета}/`, шаблоны страниц в директории `/data_templates/{id_шаблона}.htm`.

В шаблоне производится вызов статичной страницы, динамической страницы или типовой динамической страницы в соответствии с атрибутами узла структуры сайта.

Вывод текущего года для строки Copyright

Метод `$kernel->GetCurrentYear()` выводит текущий год. Данный метод полезен при указании года в строке «Copyright ©»:

```
&copy; 2005&mdash;<?php echo $kernel->GetCurrentYear()?> 000 &laquo;Хостмэйк&raquo;;
```

Меню сайта

Верхнее меню

Меню является отражением структуры сайта и выводится с использованием метода `ShowStructure()` класса модуля «Структура сайта».

Метод принимает несколько параметров, из них два являются основными:

1. `$menu_id` — идентификатор меню, получить его можно в разделе администрирования;
2. `$xsl` — имя XSL-шаблона для отображения меню.

```
<?php
$Structure = & singleton('Structure');
$Structure->ShowStructure(1, 'ВерхнееМеню');
?>
```

В приведенном примере осуществляется создание экземпляра объекта класса `Structure`, вызов метода `ShowStructure()` с передачей ему идентификатора меню (1) и названия XSL-шаблона для отображения меню ('ВерхнееМеню').

Меню сайта может быть представлено следующей структурой:

```
<ul class="top_menu gray_link">
  <li class="red_li"><div><div><a href="/about/" title="О компании">О
компаний</a></div></div></li>
  <li><div><div><a href="/shop/" title="Интернет-магазин">Интернет-
магазин</a></div></div></li>
  <li><div><div><a href="/forum/" title="Форум">Форум</a></div></div></li>
  <li><div><div><a href="/poll/" title="Опросы">Опросы</a></div></div></li>
  <li><div><div><a href="/feedback/" title="Обратная связь">Обратная
связь</a></div></div></li>
</ul>
```

Как мы видим, меню обрамлено в блок `<ul class="top_menu gray_link"> ... `, внутри меню каждый раздел представляет собой блок ` ... `.

Текущий активный раздел меню выделяется с помощью применения класса «`red_li`» к узлу ``.

Просмотреть создаваемый системой XML можно в клиентском разделе с помощью кнопки на верхней панели, более подробно о просмотре XML в соответствующем разделе настоящего руководства.

Общая структура XML-документа структуры сайта:

```
<documents>
...
<structure id="ID" menu_id="ID">
данные об узле структуры
<propertys>
<property type="TYPE" id="ID" name="ИМЯ_XML_ТЕГА">
конкретное свойство
</property>
... множество свойств каждого элемента структуры
</propertys>
вложенные структуры (неограниченный уровень)
</structure>
<structure id="ID" menu_id="ID">
...
</structure>
... множество узлов структуры
</documents>
```

XSL-шаблон для отображения меню представленной структуры:

```
<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE xsl:stylesheet>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output xmlns="http://www.w3.org/TR/xhtml1/strict" doctype-public="-//W3C//DTD XHTML
1.0 Strict//EN" encoding="Windows-1251" indent="yes" method="html" omit-xml-declaration="no"
version="1.0" media-type="text/xml"/>

  <xsl:template match="/document">
    <ul class="top_menu_gray_link">
      <!-- Выбираем узлы структуры первого уровня -->
      <xsl:apply-templates select="structure[show=1]"/>
    </ul>
  </xsl:template>

  <xsl:template match="structure">
    <!-- Запишем в константу ID структуры, данные для которой будут выводиться
пользователю -->
    <xsl:variable name="current_structure_id"
select="/document/structure/current_structure_id"/>

    <li>
      <!--
Выделяем текущую страницу добавлением к li класса red_li,
если это текущая страница, либо у нее есть ребенок с атрибутом id, равным
текущей uehggт.
-->
      <xsl:if test="current_structure_id = @id or
count(../structure[@id=$current_structure_id]) = 1">
        <xsl:attribute name="class">red_li</xsl:attribute>
      </xsl:if>

      <div>
        <div>
          <!-- Показывать ссылку, или нет -->
          <xsl:choose>
```

```

                                <xsl:when test="show_link = 1">
                                    <!-- Определяем адрес ссылки -->
                                    <xsl:variable name="link">
                                        <xsl:choose>
                                            <!-- Если внешняя ссылка -->
                                            <xsl:when test="is_external_link
= 1">
                                                <xsl:value-of disable-
output-escaping="yes" select="external_link"/>
                                            </xsl:when>
                                            <!-- Иначе если внутренняя
ссылка -->
                                            <xsl:otherwise>
                                                <xsl:value-of disable-
output-escaping="yes" select="link"/>
                                            </xsl:otherwise>
                                        </xsl:choose>
                                    </xsl:variable>
                                    <!-- Ссылка на пункт меню -->
                                    <a href="{\$link}" title="{name}"><xsl:value-of
disable-output-escaping="yes" select="name"/></a>
                                </xsl:when>
                                <!-- Если не показывать ссылке - выводим просто имя
ссылки -->
                                <xsl:otherwise>
                                    <xsl:value-of disable-output-escaping="yes"
select="name"/>
                                </xsl:otherwise>
                            </xsl:choose>
                        </div>
                    </div>
                </li>
            </xsl:template>
        </xsl:stylesheet>

```

Левое меню

Левое меню имеет код 2 и представлено следующей структурой:

```

<ul class="left_menu mp gray_link">
    <li><a href="/news/" title="Новости">Новости</a></li>
    <li><a href="/articles/" title="Статьи">Статьи</a></li>
</ul>

```

PHP-код для показа левого меню с кодом 2:

```

<?php
$Structure = & singleton('Structure');
$Structure->ShowStructure(2, 'ЛевоеМеню');
?>

```

XSL-шаблон для отображения левого меню представленной структуры:

```

<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE xsl:stylesheet>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

```

```

<xsl:output xmlns="http://www.w3.org/TR/xhtml1/strict" doctype-public="-//W3C//DTD XHTML
1.0 Strict//EN" encoding="Windows-1251" indent="yes" method="html" omit-xml-declaration="no"
version="1.0" media-type="text/xml"/>

<xsl:template match="/document">
  <ul class="left_menu mp gray_link">
    <!-- Выбираем узлы структуры первого уровня -->
    <xsl:apply-templates select="structure[show=1]"/>
  </ul>
</xsl:template>

<xsl:template match="structure">

  <!-- Запишем в константу ID структуры, данные для которой будут выводиться
пользователю -->
  <xsl:variable name="current_structure_id"
select="/document/structure/current_structure_id"/>

  <li>
    <!-- Показывать ссылку, или нет -->
    <xsl:choose>
      <xsl:when test="show_link = 1">
        <!-- Определяем адрес ссылки -->
        <xsl:variable name="link">
          <xsl:choose>
            <!-- Если внешняя ссылка -->
            <xsl:when test="is_external_link = 1">
              <xsl:value-of disable-output-
escaping="yes" select="external_link"/>
            </xsl:when>
            <!-- Иначе если внутренняя ссылка -->
            <xsl:otherwise>
              <xsl:value-of disable-output-
escaping="yes" select="link"/>
            </xsl:otherwise>
          </xsl:choose>
        </xsl:variable>
        <!-- Ссылка на пункт меню -->
        <a href="{\$link}" title="{name}">
          <!--
Выделяем текущую страницу добавлением к <a> стиля
если это текущая страница, либо у нее есть ребенок с
атрибутом id, равным текущей uheggt.
-->
          <xsl:if test="current_structure_id = @id or
count(../structure[@id=\$current_structure_id]) = 1">
            <xsl:attribute name="style">font-weight:
bold</xsl:attribute>
          </xsl:if>
          <xsl:value-of disable-output-escaping="yes"
select="name"/>
        </a>
      </xsl:when>
      <!-- Если не показывать ссылку - выводим просто имя ссылки -->
      <xsl:otherwise>
        <xsl:value-of disable-output-escaping="yes" select="name"/>
      </xsl:otherwise>
    </xsl:choose>
  </li>
</xsl:template>
</xsl:stylesheet>

```

Интернет-магазин

Список разделов каталога товаров

Ввод списка разделов каталога является частным случаем показа магазина, при котором сами товары не выбираются и не выводятся.

Показ каталога товаров осуществляется с использованием метода ShowShop() класса модуля «Интернет-магазин».

Метод принимает несколько параметров:

1. \$shop_id — идентификатор магазина, получить его можно в разделе администрирования;
2. \$xsl — имя XSL-шаблона для отображения меню;
3. \$param — массив дополнительных параметров (полный список параметров см. в API).
 - a. \$param['items_on_page'] — число товаров, отображаемых на странице;
 - b. \$param['xml_show_group_type'] — тип генерации XML для групп, может принимать значения (по умолчанию 'tree'):
 - i. all - все группы всех уровней;
 - ii. current - группы только текущего уровня;
 - iii. tree - группы, находящиеся выше по дереву;
 - iv. none - не выбирать группы.

```
<?php
// Проверяем, существует ли класс Интернет-магазина
if (class_exists('shop'))
{
    $shop = & singleton('shop');
    $shop_id = 1;

    $param = array();
    $param['items_on_page'] = 0;
    $param['xml_show_group_type'] = 'all';

    $shop->ShowShop($shop_id, 'МагазинГруппыТоваровНаГлавной', $param);
}
?>
```

Структура XML-документа для показа магазина содержит множество данных, посмотреть которые можно самостоятельно.

Пример структуры ссылок на разделы:

```
<h2><a href="/shop/" title="Интернет-магазин">Интернет-магазин</a></h2>
<ul class="left_menu">
  <li><a href="/shop/video/" title="Видеокамеры">Видеокамеры</a>
    <ul class="left_menu gray_link gray">
      <li>&mdash; <a href="/shop/video/1/" title="Цифровые">Цифровые</a></li>
      <li>&mdash; <a href="/shop/video/2/" title="Аналоговые">Аналоговые</a></li>
    </ul>
  </li>
  <li><a href="#" title="Плазменные панели">Плазменные панели</a></li>
</ul>
```

XSL-шаблон для отображения ссылок на разделы каталога:

```

<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE xsl:stylesheet>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output xmlns="http://www.w3.org/TR/xhtml1/strict" doctype-public="-//W3C//DTD XHTML
1.0 Strict//EN" encoding="Windows-1251" indent="yes" method="html" omit-xml-declaration="no"
version="1.0" media-type="text/xml"/>

  <!-- МагазинГруппыТоваровНаГлавной -->

  <xsl:template match="/">
    <xsl:apply-templates select="/shop"/>
  </xsl:template>

  <!-- Шаблон для магазина -->
  <xsl:template match="/shop">
    <SCRIPT>
      <xsl:comment>
        <xsl:text disable-output-escaping="yes">
          <![CDATA[
            function show_hide_menu(id)
            {
              if (obj = document.getElementById(id)) {
                obj.style.display == 'none' ? obj.style.display
= 'block' : obj.style.display = 'none';
              }
              return false; }
            return true;
          }
        ]]>
        </xsl:text>
      </xsl:comment>
    </SCRIPT>

    <h2><a href="{path}" title="Интернет-магазин">Интернет-магазин</a></h2>

    <ul class="left_menu" id="shop_menu">
      <xsl:apply-templates select="group"/>
    </ul>
  </xsl:template>

  <!-- Шаблон для групп товара -->
  <xsl:template match="group">
    <li>
      <!-- Если это подпункт (родитель не равен 0) - выводим слева от него тире -
-->
      <xsl:if test="@parent != 0">
        &#8212;
      </xsl:if>

      <a href="{/shop/path}{fullpath}" onClick="return show_hide_menu('{@id}');">
        <xsl:value-of disable-output-escaping="yes" select="name"/>
      </a>

      <!-- Если есть подгруппы -->
      <xsl:if test="group">
        <ul class="left_menu gray_link gray" id="{@id}" style="display:
none;">
          <xsl:apply-templates select="group"/>
        </ul>
      </xsl:if>
    </li>
  </xsl:template>
</xsl:stylesheet>

```

Горячие предложения

Горячие предложения также выводятся с помощью метода ShowShop(), при этом в метод через внешние параметры передается список скидок на товары, что позволяет выбрать только товары, имеющие скидки.

```
<?php
if (class_exists('shop'))
{
    $shop = & singleton('shop');

    $shop_id = 1;

    $param = array();

    // Товары выбираем из всех групп
    $param['current_group_id'] = false;

    // Выводим по 4 товара в блока
    $param['items_on_page'] = 4;

    // Используем случайный вывод товара
    $param['items_order'] = 'rand';

    // Получаем все скидки магазина
    $AllDiscount = $shop->GetAllDiscounts($shop_id);

    // Скидки есть
    if ($AllDiscount)
    {
        $param['select_discount'] = array();

        // Цикл по полученным скидкам
        while ($row = mysql_fetch_assoc($AllDiscount))
        {
            // В массив добавляем идентификаторы скидок
            $param['select_discount'][] = $row['shop_discount_id'];
        }
    }

    $shop->ShowShop($shop_id, 'МагазинКаталогТоваровНаГлавнойСпецПред', $param);
}
?>
```

XSL-шаблон для отображения горячих предложений:

```
<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE xsl:stylesheet>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output xmlns="http://www.w3.org/TR/xhtml1/strict" doctype-public="-//W3C//DTD XHTML
1.0 Strict//EN" encoding="Windows-1251" indent="yes" method="html" omit-xml-declaration="no"
version="1.0" media-type="text/xml"/>

    <!-- МагазинКаталогТоваровНаГлавнойСпецПред -->

    <xsl:template match="/">
        <xsl:apply-templates select="/shop"/>
    </xsl:template>

    <!-- Шаблон для магазина -->
    <xsl:template match="/shop">
```

```

        <!-- Есть товары -->
        <xsl:if test="item">
            <h2>Горячие предложения</h2>

            <!-- Выводим товары магазина -->
            <xsl:apply-templates select="item"/>
        </xsl:if>
    </xsl:template>

    <!-- Шаблон для товара -->
    <xsl:template match="item">
        <div class="shop_item">
            <div class="left" style="text-align: center;">
                <!-- Указана малое изображение -->
                <xsl:if test="small_image != ''">
                    <a href="{/shop/path}{fullpath}{path}/">
                        
                    </a>
                </xsl:if>

                <!-- Ссылка на товар -->
                <p>
                    <a href="{/shop/path}{fullpath}{path}/" title="{name}"
                        <xsl:value-of disable-output-escaping="yes"
select="name"/>
                    </a>
                </p>

                <p>
                    <!-- Цена -->
                    <b>
                        <xsl:value-of disable-output-escaping="yes"
select="price"/>&#xA0;<xsl:value-of disable-output-escaping="yes" select="currency"/>&#xA0;
                    </b>

                    <!-- Ссылку на добавление в корзины выводим, если:
                    type = 0 - простой тип товара
                    type = 1 - электронный товар, при этом остаток на складе
                    больше 0 или -1,
                    что означает неограниченное количество -->
                    <xsl:if test="type = 0 or (type = 1 and (eitem_count > 0 or
eitem_count = -1))">
                        <a
href="{/shop/path}cart/?action=add&item_id={@id}" onclick="return
AddIntoCart('{/shop/path}', {@id}, 1)"
                        
                    </a>
                </xsl:if>
                </p>
            </div>
        </div>

        <xsl:if test="position() mod 2 = 0">
            <div class="clearing"></div>
        </xsl:if>
    </xsl:template>
</xsl:stylesheet>

```

Обратите внимание на конструкцию, которая используется для вывода по два товара в строке, ее работа основана на функции получения остатка от деления — как только у текущей позиции (счет ведется с 1) остаток от деления на 2 станет равным 0, тогда происходит вставка разрыва:

```
<xsl:if test="position() mod 2 = 0">
  <div class="clearing"></div>
</xsl:if>
```

Для приведенного кода такое событие произойдет для позиции 2, 4, 6, 8, 10, 12 и т.д.

Краткая корзина

Краткая корзина предназначена для отображения пользователю общего состояния его корзины с возможностью перехода на страницу корзины с подробной информацией.

Отображение корзины осуществляется с использованием метода ShowCart() класса модуля «Интернет-магазина».

```
<!-- Краткая корзина -->
<?php
if (class_exists('shop'))
{
    $shop = & singleton('shop');
    $shop_id = 1;
    $shop->ShowCart($shop_id, false, "МагазинКорзинаКраткая");
}
?>
```

XSL-шаблон для отображения краткой корзины:

```
<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE xsl:stylesheet>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output xmlns="http://www.w3.org/TR/xhtml1/strict" doctype-public="-//W3C//DTD XHTML
1.0 Strict//EN" encoding="Windows-1251" indent="yes" method="html" omit-xml-declaration="no"
version="1.0" media-type="text/xml"/>

  <!-- МагазинКорзинаКраткая -->
  <xsl:template match="/cart">

    <div id="little_cart">
      <xsl:choose>
        <!-- Магазин не найден -->
        <xsl:when test="error_not_isset_shop/node()">
          <p>
            <b>Ошибка! Магазин с указанным идентификатором не
            найден!</b>
            <br/>
            Укажите правильный код магазина!
          </p>
        </xsl:when>
        <!-- Магазин найден -->
        <xsl:otherwise>
          <xsl:choose>
            <!-- В корзине нет ни одного элемента -->
            <xsl:when test="totalquantity = 0">
              <h2><a href="{/cart/shop/path}cart/">Корзина
            </h2>
            <xsl:choose>
              <xsl:when test="site_users_class_exists
= 1 and user_id = 0">
```



```

        <p>Если Вы зарегистрированный
пользователь, данные Вашей корзины станут видны после авторизации.</p>
        </xsl:when>
        <xsl:otherwise>Перейдите в каталог,
выберите требуемый товар и добавьте его в корзину.</xsl:otherwise>
        </xsl:choose>
    </xsl:when>
    <xsl:otherwise>
        <h2>Моя корзина</h2>

        <!-- Вывод общих количества, веса и стоимости
товаров -->
        <p>В корзине <b><xsl:value-of
select="totalquantity"/></b>&#xA0;<xsl:call-template name="declension">
        <xsl:with-param name="number"
select="totalquantity"/></xsl:call-template>
        <br/>на сумму <b><xsl:value-of
select="totalsum"/>&#xA0;<xsl:value-of disable-output-escaping="yes"
select="shop/shop_currency/shop_currency_name"/></b></p>

        <xsl:if test="totalweight > 0">
        <p>Общий вес заказа <b><xsl:value-of
select="totalweight"/>&#xA0;<xsl:value-of
select="shop/shop_mesures/shop_mesures_name"/></b>.</p>
        </xsl:if>

        <p>
        <a href="{/cart/shop/path}cart/">Перейти
в корзину &#8594;</a>
        </p>
    </xsl:otherwise>
</xsl:choose>
</xsl:otherwise>
</xsl:choose>
</div>
</xsl:template>

<!-- Склонение после числительных -->
<xsl:template name="declension">

    <xsl:param name="number" select="number"/>

    <!-- Именительный падеж -->
    <xsl:variable name="nominative">
        <xsl:text>товар</xsl:text>
    </xsl:variable>

    <!-- Родительный падеж, единственное число -->
    <xsl:variable name="genitive_singular">
        <xsl:text>товара</xsl:text>
    </xsl:variable>

    <xsl:variable name="genitive_plural">
        <xsl:text>товаров</xsl:text>
    </xsl:variable>

    <xsl:variable name="last_digit">
        <xsl:value-of select="$number mod 10"/>
    </xsl:variable>

    <xsl:variable name="last_two_digits">
        <xsl:value-of select="$number mod 100"/>
    </xsl:variable>

```

```

        <xsl:choose>
            <xsl:when test="\$last_digit = 1 and \$last_two_digits != 11">
                <xsl:value-of select="\$nominative"/>
            </xsl:when>
            <xsl:when test="\$last_digit = 2 and \$last_two_digits != 12 or
\$last_digit = 3 and \$last_two_digits != 13 or \$last_digit = 4 and \$last_two_digits !=
14">
                <xsl:value-of select="\$genitive_singular"/>
            </xsl:when>
            <xsl:otherwise>
                <xsl:value-of select="\$genitive_plural"/>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>
</xsl:stylesheet>

```

Информационные системы

Вывод анонсов новостей

Новости являются информационной системой, показ осуществляется с использованием метода ShowInformationSystem () модуля «Информационные системы».

Метод принимает несколько обязательных параметров:

1. \$InformationSystemIdArray — массив идентификаторов или идентификатор информационной системы;
2. \$information_groups_id — идентификатор информационной группы, подгруппы и элементы которой необходимо показать. Для выбора элементов из всех групп указывается false.
3. Название XSL-шаблона для вывода;
4. \$xsl_name — имя XSL шаблона для отображения групп и элементов информационной системы;
5. \$items_on_page — число информационных элементов, отображаемых на странице;
6. \$items_begin — номер, начиная с которого выводить информационные элементы.

```

<!-- Новости -->
<?php
// Вывод информационной системы
$InformationSystem = & singleton('InformationSystem');

// Количество выводимых элементов
$item_count = 3;

// Код информационной системы
$InformationSystemId = 1;

// Код отображаемой группы инфосистем
$InformationGroupId = false; // false - из всех групп, 0 - из корневой группы

$InformationSystem->ShowInformationSystem($InformationSystemId, $InformationGroupId,
'СписокНовостейНаГлавной', $item_count, 0);
?>

```

Пример структуры блока новостей:

```

<h2><a href="/news/" title="Новости">Новости</a></h2>
<dl class="news_list">

```

```

    <dt>12 июня 2009г.</dt>
    <dd><a href="/news/1/" title="Best Buy станет конкурентом iTunes">Best Buy станет
конкурентом iTunes</a></dd>
    <dt>12 июня 2009г.</dt>
    <dd><a href="/news/2/" title="Best Buy станет конкурентом iTunes">Best Buy станет
конкурентом iTunes</a></dd>
    <dt>12 июня 2009г.</dt>
    <dd><a href="/news/3/" title="Best Buy станет конкурентом iTunes">Best Buy станет
конкурентом iTunes</a></dd>
</dl>

```

XSL-шаблон для отображения анонса новостей:

```

<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE xsl:stylesheet>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:output xmlns="http://www.w3.org/TR/xhtml1/strict" doctype-public="-//W3C//DTD XHTML
1.0 Strict//EN" encoding="Windows-1251" indent="yes" method="html" omit-xml-declaration="no"
version="1.0" media-type="text/xml"/>

  <!-- Список новостей на Главной -->

  <xsl:template match="/">
    <xsl:apply-templates select="/document"/>
  </xsl:template>

  <xsl:template match="/document">
    <!-- Выводим название информационной системы -->
    <h2>
      <a href="{blocks/url}" title="{blocks/name}"><xsl:value-of disable-output-
escaping="yes" select="blocks/name"/></a>
    </h2>

    <!-- Отображение записи информационной системы -->
    <xsl:if test="blocks/items/item[item_status=1]">
      <dl class="news_list">
        <xsl:apply-templates select="blocks/items/item[item_status=1]"/>
      </dl>
    </xsl:if>
  </xsl:template>

  <!-- Шаблон вывода информационного элемента -->
  <xsl:template match="item">
    <!-- Дата время -->
    <dt>
      <xsl:value-of disable-output-escaping="yes" select="substring-
before(item_date, '.')"/>
      <xsl:variable name="month_year" select="substring-after(item_date, '.')"/>
      <xsl:variable name="month" select="substring-before($month_year, '.')"/>
      <xsl:choose>
        <xsl:when test="$month = 1"> января </xsl:when>
        <xsl:when test="$month = 2"> февраля </xsl:when>
        <xsl:when test="$month = 3"> марта </xsl:when>
        <xsl:when test="$month = 4"> апреля </xsl:when>
        <xsl:when test="$month = 5"> мая </xsl:when>
        <xsl:when test="$month = 6"> июня </xsl:when>
        <xsl:when test="$month = 7"> июля </xsl:when>
        <xsl:when test="$month = 8"> августа </xsl:when>
        <xsl:when test="$month = 9"> сентября </xsl:when>
        <xsl:when test="$month = 10"> октября </xsl:when>
        <xsl:when test="$month = 11"> ноября </xsl:when>
        <xsl:otherwise> декабря </xsl:otherwise>
      </xsl:choose>
    </dt>
  </xsl:template>

```

```

        <xsl:value-of disable-output-escaping="yes" select="substring-
after($month_year, '.')"/>г.
    </dt>

    <!-- Название -->
    <dd>
        <a href="{item_path}" title="{item_name}">
            <xsl:value-of disable-output-escaping="yes" select="item_name"/>
        </a>
    </dd>
</xsl:template>
</xsl:stylesheet>

```

Метки информационных систем

Показ облака тегов для меток информационных систем осуществляется с использованием метода ShowTagsCloud () модуля «Информационные системы».

Метод принимает несколько обязательных параметров:

1. \$InformationSystemId —идентификатор информационной системы;
2. \$xsl_name — имя XSL шаблона для отображения групп и элементов информационной системы.

```

<?php
// Вывод меток информационной системы
$InformationSystem = & singleton('InformationSystem');

// Код информационной системы
$InformationSystemId = 1;

$InformationSystem->ShowTagsCloud($InformationSystemId, 'ОблакоТэговИнформационнойСистемы');
?>

```

Пример структуры облака тегов для меток информационных систем:

```

<h2>Метки</h2>
<div>
    <ul class="tag_gray_link">
        <li><a href="/news/tag/ajax/" style="font-size: 10pt">ajax</a></li>
        <li><a href="/news/tag/hostcms/" style="font-size: 14pt">hostcms</a></li>
    </ul>
</div>

```

XSL-шаблон для отображения облака тегов для меток информационных систем:

```

<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE xsl:stylesheet>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:math="http://exslt.org/math">
    <xsl:output xmlns="http://www.w3.org/TR/xhtml1/strict" doctype-public="-//W3C//DTD XHTML
1.0 Strict//EN" encoding="Windows-1251" indent="yes" method="html" omit-xml-declaration="no"
version="1.0" media-type="text/xml"/>

    <!-- ОблакоТэговИнформационнойСистемы -->

    <xsl:template match="/">

```

```

        <xsl:apply-templates select="/document"/>
    </xsl:template>

    <xsl:template match="/document">

        <xsl:if test="count(InformationSystem/tags/tag) != 0">
            <div>
                <h2>Метки</h2>

                <xsl:variable name="max_tag_count"
select="/document/InformationSystem/tags/tag/count[not(. &lt;
/document/InformationSystem/tags/tag/count)])[1] - 1"/>

                <xsl:variable name="max_size" select="16"/>
                <xsl:variable name="min_size" select="9"/>

                <xsl:variable name="coeff_size">
                    <xsl:choose>
                        <xsl:when test="$max_tag_count &gt; 0">
                            <xsl:value-of select="($max_size - $min_size) div
$max_tag_count"/>
                        </xsl:when>
                        <xsl:otherwise>0</xsl:otherwise>
                    </xsl:choose>
                </xsl:variable>

                <div>
                    <ul class="tag_gray_link">
                        <xsl:apply-templates select="InformationSystem/tags/tag">
                            <xsl:with-param name="min_size" select="$min_size"/>
                            <xsl:with-param name="coeff_size"
select="$coeff_size"/>
                        </xsl:apply-templates>
                    </ul>
                </div>
            </div>
        </xsl:if>

    </xsl:template>

    <!-- Облако из групп -->
    <xsl:template match="tag">

        <xsl:param name="min_size"/>
        <xsl:param name="coeff_size" select="10"/>

        <!-- Нужный размер шрифта вычисляется по формуле $min_size + количество *
$coeff_size -->
        <xsl:variable name="size" select="round($min_size + ((count - 1) * $coeff_size))"/>

        <li>
            <a href="{/document/InformationSystem/url}tag/{tag_path_name}/"
style="font-size: {$size}pt">
                <xsl:value-of select="tag_name"/>
            </a>
        </li>

        <xsl:text> </xsl:text>

    </xsl:template>
</xsl:stylesheet>

```

Поиск

Форма поиска в макете сайта

В макете сайта форма поиска может быть размещена в виде HTML-кода с формой, в которой поле поиска имеет имя «text», например:

```
<form action="/search/" method="GET">
  <input type="text" class="text" name="text" />
  <input type="image" src="/images/search_white.gif" title="Поиск"/>
</form>
```

Ограничение поиска по определенным источникам контента

Начиная с версии 5.0 система управления имеет возможность поиска по определенной области источника контента. Для этого в метод GoSearch класса Search передается необязательный атрибут \$property:

- \$property['current_page'] — номер отображаемой страницы
- \$property['items_on_page'] — количество записей на странице
- \$property['len']=200 — максимальная длина поискового запроса;
- \$property['site_id'] — идентификатор сайта, по которому производится поиск;
- \$property['search_page_module'] массив модулей, ключами которого являются номера модулей, а значениями — массив идентификаторов элементов.
Номера модулей:
 - 0 — Структура сайта;
 - 1 — Информационные системы;
 - 2 — Форум;
 - 3 — Интернет-магазин;
 - 4 — HelpDesk.

Произведем, например, поиск по информационной системе с номером 5 и 7, а также по магазину с номером 17.

```
$Search = new Search();

$property['search_page_module'] = array(
  1 => array (5, 7),
  3 => array (17)
);

$Search->GoSearch($words, to_str($GLOBALS['LA']['xsl']), $property);
```

Пример поиска по информационной системе с номером 5 и 7 (с дополнительным условием поиска только по информационным элементам), а также по магазину с номером 17.

```
$Search = new Search();

$property['search_page_module'] = array(
  1 => array (5,
             array('search_page_module_entity_id' => 7, 'search_page_module_value_type' => 2)),
  3 => array (17));
```

```
$Search->GoSearch($words, to_str($GLOBALS['LA']['xsl']), $property);
```

При указании массива с дополнительными условиями он может принимать следующие аргументы:

- **search_page_module_entity_id** — целое число, ID сущности, например, магазин с кодом 7
- **search_page_module_value_type** — целое число или массив, ID типа, например, 1 - группа, 2 - элемент (или товар)
- **search_page_module_value_id** — целое число или массив, ID сущности указанного типа (например, ID товара или группы) при поиске только по ним.

Если ссылки из результатов поиска ведут на другой сайт

Такая ситуация может возникнуть при неправильном указании основного домена для сайта. Для исправления ситуации перейдите в раздел «Сайты», выберите пиктограмму «Домены» нужного сайта. В открывшемся окне установите основной домен. Переиндексируйте сайт.

Опросы

Добавления опросов на сайт

Создайте группу опросов и опрос (см. руководство пользователя). Перейдите в раздел «Структура сайта», выберите ссылку «Добавить узел». Заполните основные параметры страницы, в поле «Название раздела» внесите значение polls, выберите тип раздела «Типовая динамическая страница», «Раздел» — Опросы и «Страница» — Опросы.

В появившемся блоке выберите требуемую группу опросов; «XSL-шаблон для отображения результатов опросов» — «Опросы» → «ОтображениеРезультатовОпроса»; «XSL-шаблон для отображения списка опросов» — «Опросы» → «ОтображениеОпросаБезРезультатов».

Раздел	Опросы
Страница	Опросы
Группа опросов	Группа опросов1 [1]
Число опросов, выводимых в списке	1
Порядковый номер опроса, с которого начинать показ списка опросов	0
Направление сортировки	В произвольном порядке
Показать результаты до голосования	<input checked="" type="checkbox"/> Да
XSL-шаблон для отображения результатов опросов	Опросы ОтображениеРезультатовОпроса
XSL-шаблон для отображения списка опросов	Опросы ОтображениеОпросаБезРезультатов
<input type="button" value="Сохранить"/> <input type="button" value="Применить"/>	

Добавления блока опроса в макет сайта

В шаблон страницы или макет добавьте следующий код:

```

<?php
if (class_exists('polls'))
{
    $polls = & singleton('polls');

    // Идентификатор группы опросов
    $PollsGroupID = 1;

    $polls->ShowPollsGroup($PollsGroupID, 'ОтображениеОпросаБезРезультатов');
}
?>

```

XSL-шаблон для отображения опроса без результатов:

```

<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE xsl:stylesheet>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output xmlns="http://www.w3.org/TR/xhtml1/strict" doctype-public="-//W3C//DTD XHTML
1.0 Strict//EN" encoding="Windows-1251" indent="yes" method="html" omit-xml-declaration="no"
version="1.0" media-type="text/xml"/>

    <!-- ОтображениеОпросаБезРезультатов -->

    <xsl:template match="/">
        <xsl:apply-templates select="/polls/polls_group/poll"/>
    </xsl:template>

    <xsl:template match="poll">
        <div class="survey_block">
            <div class="red_block survey">
                <div class="t1"></div>
                <div class="tr"></div>
                <div class="br"></div>
                <div class="b1"></div>
                <div class="tail"></div>
                <div>
                    <xsl:value-of select="title"/>
                </div>
            </div>

            <br/>

            <form action="{/polls/polls_group/poll_group_url}poll-{@id}/"
method="post">

                <!-- Выводим варианты ответа -->
                <xsl:apply-templates select="replys/reply" />

                <xsl:if test="public=1">
                    <p>
                        <a href="{/polls/polls_group/poll_group_url}poll-
{@id}/">Результаты голосования &#8594;</a>
                    </p>
                </xsl:if>

                <div class="gray_button">
                    <div>
                        <input type="submit" name="vote" value="Ответить"/>
                    </div>
                </div>

                <div class="clearing"></div>

                <input type="hidden" name="polls_id" value="{@id}"/>

```



```

        <input type="hidden" name="public_vote" value="{public}"/>
      </form>
    </div>
  </xsl:template>

  <xsl:template match="replies/reply">
    <xsl:variable name="current_polls_group" select="current_polls_group"/>

    <p>
      <xsl:if test="../../type = 0">
        <input type="radio" class="input_radio" name="poll_reply_id"
id="poll_reply_id{@id}" value="{@id}">
        <!-- Для первого варианта отмечаем, что он выбран -->
        <xsl:if test="position() = 1">
          <xsl:attribute name="checked"></xsl:attribute>
        </xsl:if>
        </input>
      </xsl:if>
      <xsl:if test="../../type = 1">
        <input type="checkbox" name="poll_reply_id{@id}"
id="poll_reply_id{@id}" value="{@id}"></input>
      </xsl:if>

      <label for="poll_reply_id{@id}" accesskey="">
        <xsl:value-of select="name"/>
      </label>
    </p>
  </xsl:template>
</xsl:stylesheet>

```

Структура сайта

Работа с дополнительными свойствами структуры в XSL

Свойства узла структуры могут быть использованы для различных решений – например изображения графического меню, alt (или title) подписи и так далее. Значения свойств получаются в XSL-шаблонах вывода структуры.

Для внедрения в XSL значений свойств используются конструкции, значение поля:

```
<xsl:value-of select="propertys/property[@name = 'FIELD_NAME']/value"/>
```

Вывод большой картинки:

```

```

Вывод малой картинки:

```

```

Вместо FIELD_NAME необходимо указать имя поля.

Создание страницы для отображения ошибки 403/404/503

Система обрабатывает ошибки 403 (доступ к ресурсу запрещен) и 404 (файл не найден).

Для указания страницы, которая должна быть показана при ошибке, в раздел «Сайты» отредактируйте атрибуты сайта и в закладке «Ошибки» из выпадающего списка выберите разделы структуры, который должен отображаться пользователю.

Если страница для ошибки 404 не указана, то производится редирект на главную страницу сайта.

Примечание: Для организации перенаправления пользователя (редиректа) на какую либо другую страницу сайта (или внешнюю страницу), укажите узлу структуры для 404 ошибки в качестве параметра «Ссылка на другой файл» адрес страницы-перенаправления.

Создание карты сайта

Карта сайта организуется с использованием модуля Структура. Создайте новую страницу «Карта сайта» в разделе «Структура сайта», укажите тип страницы – «Типовая динамическая страница».

Тип раздела	
<input type="radio"/>	Страница
<input checked="" type="radio"/>	Типовая динамическая страница
<input type="radio"/>	Динамическая страница
Макет	
Основной макет сайта	
Раздел	
Карта сайта	
Страница	
Карта сайта	
XSL шаблон	Карта сайта
	КартаСайта
Отображать группы информационных систем	<input checked="" type="checkbox"/> Да
Отображать элементы информационных систем	<input checked="" type="checkbox"/> Да
Отображать группы магазина	<input checked="" type="checkbox"/> Да
Отображать товары магазина	<input checked="" type="checkbox"/> Да
Родительский узел	0

Выберите раздел «Карта сайта» и страницу «Карта сайта».

Укажите требуемый XSL-шаблон для отображения карты сайта (по умолчанию устанавливается шаблон «Карта сайта»).

В карту сайта автоматически могут попадать группы и элементы информационных систем. Указать необходимость их публикации в карте сайта Вы можете с помощью соответствующих флажков.

Информационные системы

Работа с дополнительными свойствами информационных элементов

Для получения дополнительного свойства текстового/числового типа используется следующая конструкция:

```
<xsl:value-of disable-output-escaping="yes"
select="item_propertys/item_property[@xml_name='XML_ИМЯ_СВОЙСТВА']/value"/>
```

Внимание! Имеется устаревший вариант, оставленный для совместимости:

```
<xsl:value-of disable-output-escaping="yes"
select="item_propertys/item_property/XML_ИМЯ_СВОЙСТВА"/>
```

Пример проверки значения свойства, если поле не заполнено, выводится «Расположение: страна не указана»:

```
Расположение:
<xsl:choose>
<xsl:when test="item_propertys/item_property[@xml_name='XML_ИМЯ_СВОЙСТВА']/value != ''">
  <xsl:value-of disable-output-escaping="yes"
select="item_propertys/item_property[@xml_name='XML_ИМЯ_СВОЙСТВА']/value"/>
</xsl:when>
<xsl:otherwise>
  страна не указана
</xsl:otherwise>
</xsl:choose>
```

Пример проверки значения свойства условным оператором, в котором значение свойства проверяется на неравенство пустой строке (два апострофа, используются именно апострофы, а не кавычки, т.к. условие уже внутри кавычек):

```
<xsl:if test="item_propertys/item_property[@xml_name='XML_ИМЯ_СВОЙСТВА']/value != ''">
Расположение: <xsl:value-of disable-output-escaping="yes"
select="item_propertys/item_property[@xml_name='XML_ИМЯ_СВОЙСТВА']/value"/>
</xsl:if>
```

Дополнительные свойства типа файл

Дополнительные свойства файлового типа представляются в XML, генерируемом системой в следующем виде:

```
<item_property type="File" xml_name="myfile1" parent_id="0" id="4154">
  <property_id>1814</property_id>
  <property_xml_name>myfile1</property_xml_name>
  <property_name>Мой файл</property_name>
  <myfile1>Мой_новый_файл.jpg</myfile1>
  <property_order>0</property_order>
  <value>Мой_новый_файл.jpg</value>
```

```

    <property_file_path width="10" height="3"
size="696">/upload/information_system_1/2/2/2/item_22222234/Мой_новый_файл.jpg</property_file_pa
th>
    <small_image>
        <value>Мой_новый_файл.jpg</value>
        <property_file_path width="10" height="3"
size="696">/upload/information_system_1/2/2/2/item_22222234/small_ Мой_новый_файл.jpg
</property_file_path>
    </small_image>
</item_property>

```

Узлы имеют следующие значения:

- XML-имя свойства: <property_xml_name>myfile1</property_xml_name>
- Имя свойства: <property_name>Мой файл</property_name>
- Устаревший теги имя загруженного файла (оригинальное): <myfile1>Мой_новый_файл.jpg</myfile1>
- Порядок сортировки свойства: <property_order>0</property_order>
- Имя загруженного файла (оригинальное): <value>Мой_новый_файл.jpg</value>
- Ссылка на файл:

```

<property_file_path width="10" height="3"
size="696">/upload/information_system_1/2/2/2/item_22222234/Мой_новый_файл.jpg</property_file_pa
th>

```

- Данные о малом изображении содержатся в <small_image>

Фрагмент XSL-шаблона для вставки ссылки на загруженный в дополнительное свойство файл с указанием оригинального имени файла

```

<a href="{item_propertys/item_property[@xml_name='myfile1']/property_file_path}">
<xsl:value-of select="item_propertys/item_property[@xml_name='myfile1']/value"/>
</a>

```

Вместо *myfile1* необходимо указать XML-имя соответствующего дополнительного свойства. Обратите внимание на необходимость указания ведущего слэша перед значением адреса свойства.

Работа с дополнительными свойствами групп информационной системы

Для получения в XSL-шаблоне дополнительного свойства текстового/числового типа используется следующая конструкция:

```

<xsl:value-of disable-output-escaping="yes" select="
//group[@id=$parent_group_id]/propertys/property[@xml_name='XML_ИМЯ_СВОЙСТВА']/value" />

```

Константа \$parent_group_id должна содержать идентификатор группы.

Внедрение информационной системы на сайт

Порядок добавления информационной системы систему подробно описан в руководстве по наполнению сайта.

Создание экспорта RSS 2.0 записей из информационных систем

RSS экспорт чаще всего размещают под корневым узлом вывода информационного элемента.

Например, если новости размещены по адресу `/news/`, то RSS поток чаще всего организую по адресу `/news/rss/`

Структура сайта

Раздел [Меню сайта](#)

[Список узлов структуры](#) → [Новости](#)

	Код ↑↓	Название ↑↓	Путь ↑↓	Активность	Индексируется	Сортировка ↑↓	
<input type="checkbox"/>				—	—	—	
<input type="checkbox"/>	20	RSS	/news/rss/			<input type="text" value="0"/>	
<input type="checkbox"/>	72	Импорт новостей	/news/import/			<input type="text" value="0"/>	

[Применить](#) [Удалить](#)

Для публикации RSS-канала, источником данных которого является информационная система, создайте узел структуры, заполните обязательные атрибуты, тип раздела укажите «Типовая динамическая страница», выберите раздел «RSS» и страницу «RSS канал для информационной системы».

Раздел

Страница

Код информационной системы

Число выводимых элементов в ленте

Позиция, с которой начинается вывод элементов

Код выводимой группы

Удалять теги из RSS Да

Показывать в формате Yandex:full-text Да

Заголовок RSS-канала

Описание RSS-канала

URL RSS-канала

Изображение для RSS-канала (URL)

Отображать изображение для информационного элемента в RSS Да

В открывшемся блоке для атрибута «Код информационной системы» укажите из выпадающего списка информационную систему, элементы которой должны публиковаться в RSS-канале, заполните другие атрибуты. Заголовок RSS-канала, описание RSS-канала и URL RSS-канала могут быть не заданы, в таком случае они генерируются системой на основании данных информационной системы.

Навигационная цепочка — «Хлебные крошки»

«Хлебные крошки» являются элементом навигации по сайту и предназначены для указания пользователю текущего месторасположения на сайте.

Для вывода навигационной цепочки можно использовать в макете или шаблоне страницы следующий код:

```
<?php
// Вывод строки навигации
$InformationSystem = & singleton('InformationSystem');

// Определяем информацию об информационной системе, связанной с текущим узлом
$InformationSystem_id = $InformationSystem-
>GetInformationSystemByStructureId(CURRENT_STRUCTURE_ID);

$params = array();
$params['xml_show_structure_property'] = false;

$external_property = array();

// Если с узлом структуры связана ИС
if ($InformationSystem_id)
{
    $result = $InformationSystem->GetInformationFromPath($InformationSystem_id, '', false);

    // определяем id информационного элемента
    if ($result['item'])
    {
        $item_id = $InformationSystem->GetIdInformationItem($result['item'],
$result['group'], $InformationSystem_id);

        if ($item_id != 0)
        {
            $external_property['item'] = $item_id;
            $params['show_groups'] = array($result['group']);
            $params['show_items'] = array($item_id);
        }
    }
    else
    {
        $item_id = false;
    }

    if (is_array($result) && $result['group'] > 0)
    {
        $external_property['group'] = $result['group'];
        $params['show_groups'] = array($result['group']);
    }

    // Данные для хлебных крошек заполняем только для текущей ИС
    $params['show_information_systems'][] = $InformationSystem_id;
}
elseif (class_exists('shop'))
{
    $shop = & singleton('shop');

    // Определяем информацию о магазине, связанным с магазином
    $shop_row = $shop->GetShopWhithStructureId(CURRENT_STRUCTURE_ID, CURRENT_SITE);

    if ($shop_row)
    {
```

```

$result = $shop->GetItemPath($shop_row['shop_shops_id'], '', false);

// Если товар
if ($result['item'])
{
    $item_id = $result['item'];

    // Получаем данные о товаре
    $item_row = $shop->GetItem($item_id);

    if ($item_row)
    {
        // Если товар является модификация, подставляем в ID товара
        if ($item_row['shop_items_catalog_modification_id'] != 0)
        {
            $item_id = $item_row['shop_items_catalog_modification_id'];
        }

        if ($item_id != 0)
        {
            $external_propertys['item'] = $item_id;
            $param['show_shop_groups'] = array($result['group']);
            $param['show_shop_items'] = array($item_id);
        }
    }
}
else
{
    $item_id = false;
}

// Если группа
if (is_array($result) && $result['group'] > 0)
{
    $external_propertys['group'] = $result['group'];
    $param['show_shop_groups'] = array($result['group']);
}
}

}

$structure = & singleton('Structure');
$structure->ShowStructure(false, 'Хлебные крошки', $param, $external_propertys);

// /Хлебные крошки
?>

```

XSL-шаблон «Хлебные крошки»:

```

<?xml version="1.0" encoding="windows-1251"?>
<!DOCTYPE xsl:stylesheet>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:output xmlns="http://www.w3.org/TR/xhtml1/strict" doctype-public="-//W3C//DTD XHTML
1.0 Strict//EN" encoding="Windows-1251" indent="yes" method="html" omit-xml-declaration="no"
version="1.0" media-type="text/xml"/>

    <!-- Хлебные крошки -->

    <xsl:template match="/document">

        <!-- Запишем в константу ID структуры, данные для которой будут выводиться
пользователю -->
        <xsl:variable name="current_structure_id">

```

```

        <xsl:choose>
            <xsl:when test="item/node()">item_<xsl:value-of
select="item"/></xsl:when>
            <xsl:when test="group/node()">group_<xsl:value-of
select="group"/></xsl:when>
            <xsl:otherwise>
                <xsl:value-of select="structure/current_structure_id"/>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:variable>

    <xsl:if test="count(//structure[@id = $current_structure_id]) > 0">
        <a href="/"><xsl:value-of disable-output-escaping="yes"
select="structure[link = '/']/name"/></a>
        <xsl:apply-templates select="//structure[@id = $current_structure_id]">
            <xsl:with-param name="is_last_item" select="1"/>
        </xsl:apply-templates>
    </xsl:if>
</xsl:template>

<xsl:template match="structure">

    <xsl:param name="is_last_item" select="0"/>

    <!-- Выбираем все узлы структуры всех уровней вложенности -->
    <xsl:apply-templates select="parent::structure"/>

    <!-- Определяем адрес ссылки -->
    <xsl:variable name="link">
        <xsl:choose>
            <!-- Если внешняя ссылка -->
            <xsl:when test="is_external_link=1">
                <xsl:value-of disable-output-escaping="yes"
select="is_external_link"/>
            </xsl:when>
            <!-- Иначе если внутренняя ссылка -->
            <xsl:otherwise>
                <xsl:value-of disable-output-escaping="yes" select="link"/>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:variable>

    <span class="path_arrow">&#x2192;</span>

    <!-- Показывать ссылку, или нет -->
    <xsl:choose>
        <xsl:when test="show_link = 1 and $is_last_item = 0">
            <a href="{ $link }">
                <xsl:value-of disable-output-escaping="yes" select="name"/>
            </a>
        </xsl:when>
        <!-- Если не показывать ссылку - выводим просто имя ссылки -->
        <xsl:otherwise>
            <xsl:value-of disable-output-escaping="yes" select="name"/>
        </xsl:otherwise>
    </xsl:choose>
</xsl:template>
</xsl:stylesheet>

```

Формы

Добавление формы на сайт

До добавления формы на сайт создайте форму в разделе администрирования, добавьте поля формы.

Перейдите в раздел «Структура сайта», создайте страницу, на которой будет располагаться форма.

Укажите тип раздела "Типовая динамическая страница", в поле "Раздел" выберите "Формы", в поле "Страница" выберите "Отображение формы".

В открывшемся блоке заполните параметры публикации формы на странице:

Поле	Описание
Код формы	Форма, с которой ведется работа. Указывается из выпадающего списка.
Тип отправляемого письма	Отправляемые письма могут отправляться в HTML или текстовом формате. Поставляемые с системой XSL-шаблоны предназначены для отправки писем в текстовом формате. При изменении типа отправляемого письма необходимо внести изменения в XSL-шаблоны. Внимание! При внесении изменения в для отправки в HTML формате не забудьте изменить в заголовке XSL-шаблона: <i>method="text"</i> на <i>method="html"</i>
Имя поля, содержащего электронную почту отправителя	Указывается имя поля формы, в которое посетитель сайта вносит свой электронный адрес. При указании данного поля в отправляемых письмах с заполненной формой будет указываться обратный адрес, указанный в поле с указанным именем.
XSL-шаблон для обработки отправки формы	Шаблон с именем "ОбработкаОтправкиФормы"
XSL-шаблон для письма куратору формы	Шаблон с именем "ПисьмоКураторуФормы"
XSL-шаблон для отображения формы	Шаблон с именем "ОтобразитьФорму"

Пример заполнения блока настроек типовой динамической страницы

Код формы	Тестовая форма [1] ▼
Тип отправляемого письма	Текст ▼
Имя поля, содержащего эл. почту отправителя	email
XSL шаблон для обработки отправки формы	Формы ▼ ОбработкаОтправкиФормы ▼
XSL шаблон для письма куратору формы	Формы ▼ ПисьмоКураторуФормы ▼
XSL шаблон для отображения формы	Формы ▼ ОтобразитьФорму ▼
<input type="button" value="Сохранить"/> <input type="button" value="Применить"/>	

Внедрение формы в макет или шаблон страницы

Часто возникает необходимость опубликовать форму (например, форма обратной связи) в макете или шаблоне страницы для отображения на нескольких страницах. Для этого в нужном месте макета или шаблона страницы добавим отображение формы:

```
<?php
$Forms = new Forms();

/* Идентификатор формы, замените на идентификатор нужной формы */
$form_id = 1;

/* Отображаем форму */
$Forms->ShowForm($form_id, 'ОтобразитьФормуНаписатьПисьмо');
?>
```

В коде отображения формы мы указали новый XSL-шаблон "ОтобразитьФормуНаписатьПисьмо", который был клонирован с XSL-шаблона ОтобразитьФорму. Название "ОтобразитьФормуНаписатьПисьмо" условно и может быть заменено на другое значение.

В клонированный XSL-шаблон вносим небольшие изменения, вместо

```
<!-- Параметр action формы должен быть "./", если обработчик на этой же странице, либо "/form/",
если обработчик на другой странице, например /form/ -->
<form name="form{form_id}" action="." method="post" ENCTYPE="multipart/form-data">
```

указываем явно путь к странице, на которой расположена форма с обработчиком, например /feedback/.

```
<!-- Параметр action формы должен быть "./", если обработчик на этой же странице, либо "/form/",
если обработчик на другой странице, например /form/ -->
<form name="form{form_id}" action="/feedback/" method="post" ENCTYPE="multipart/form-data">
```

Проблемы при отправке формы

Пользователи, перешедшие с версии 3.2.3 на новые версии 3.2.4 с поддержкой CAPTCHA могут столкнуться с проблемой отправки формы, когда форма отображается, однако после заполнения не отправляется куратору формы. Для решения данной проблемы необходимо обновить XSL-шаблоны для форм, а также код динамической страницы. Если у Вас есть в структуре сайта, под узлом с формой, раздел «Обработка формы» — его необходимо удалить.

Пользователи сайта

Публикация личного кабинета

Необходимо создать узел структуры сайта, указать путь «users» и тип "Типовая динамическая страница":

Раздел	Пользователи сайта
Страница	Пользователи сайта
XSL личного кабинета	Пользователи сайта ЛичныйКабинетПользователя
<input type="button" value="Сохранить"/> <input type="button" value="Применить"/>	

Затем, необходимо создать следующие динамические страницы, указав в качестве родительского раздела приведенную выше страницу.

Страница заказанных товаров

Страница имеет путь "order", родительский узел "Личный кабинет", полный путь к странице "/users/order/", тип страницы "Типовая динамическая страница".

Раздел	Пользователи сайта
Страница	Заказы
XSL заказов	Интернет-магазин СписокЗаказов
<input type="button" value="Сохранить"/> <input type="button" value="Применить"/>	

Страница восстановления пароля

Страница имеет путь "restore_password", родительский узел "Личный кабинет", полный путь к странице "/users/restore_password/", тип страницы "Типовая динамическая страница".

Раздел	Пользователи сайта
Страница	Восстановление пароля
XSL письма восстановления пароля	Пользователи сайта ПисьмоВосстановлениеПароля
<input type="button" value="Сохранить"/> <input type="button" value="Применить"/>	

Страница регистрации/редактирования анкетных данных

Страница имеет путь "registration", родительский узел "Личный кабинет", полный путь к странице "/users/registration/", тип страницы "Типовая динамическая страница".

Раздел	Пользователи сайта	
Страница	Регистрация	
XSL письма подтверждения	Пользователи сайта	ПисьмоПодтверждениеРегистрации
XSL регистрации пользователя	Пользователи сайта	РегистрацияПользователя
<input type="button" value="Сохранить"/> <input type="button" value="Применить"/>		

Страница почтовых рассылок, на которые подписан пользователь

Страница имеет путь "maillist", родительский узел "Личный кабинет", полный путь к странице "/users/maillist/", тип страницы "Типовая динамическая страница".

Раздел	Пользователи сайта	
Страница	Почтовые рассылки	
XSL списка рассылок	Почтовые рассылки	СписокРассылок
<input type="button" value="Сохранить"/> <input type="button" value="Применить"/>		

Реклама

Размещение кода показа баннера на сайте

Для показа баннера встройте в код макета(ов) или шаблона страниц (в зависимости от структуры страниц) в соответствующие места показ баннеров группы:

```
<?php
    $Advertisement = new Advertisement();
    $banner_group = 3;
    $Advertisement->ShowBannerGroup($banner_group);
?>
```

для переменной \$banner_group установите соответствующий идентификатор группы баннеров, которую Вы планируете показывать в данном месте.

Показ баннеров возможен с использованием XSL-шаблона, в таком случае можно воспользоваться следующим методом:

```
<?php
$Advertisement = new Advertisement();

$advertisement_group_id = 3;
$xml_name = 'ОтображениеБаннера';

$Advertisement->ShowAdvertisementGroup($advertisement_group_id, $xml_name);
?>
```

Защита от показа баннера поисковым ботам

При наличии модуля статистики посещаемости код баннера можно располагать внутри условия:

```
<?php
$counter = new counter();
$is_bot = $counter->CheckBot(to_str($_SERVER['HTTP_USER_AGENT']));

if (!$is_bot)
{
    // Здесь вывод рекламы
}
?>
```

Контекстный показ баннеров

Контекстный показ баннеров может использоваться с разными источниками контекста — текстом страницы, текстом поискового запроса и т.п.

Контекстный показ баннера в зависимости от содержания страницы

Для показа баннера в зависимости от содержания страницы получим текст страницы следующим образом:


```

<?php
ob_start();

// здесь показ контента, в зависимости от которого необходимо выводить баннер

// Записываем текст в параметр и выводим в поток
$params['contextual_words'] = ob_get_clean();

// Указываем на необходимость показа только баннеров, найденных по контексту. Если ни одного
// контекстного баннера не будет найдено – ничего не будет показано.
$params['show_only_context'] = true;
?>

```

Далее остается передать этот \$params в метод ShowAdvertisementGroup() или ShowBannerGroup():

```

<?php
$advertisement = new Advertisement();

$advertisement_group_id = 3;
$xml_name = 'ОтображениеБаннера';

$advertisement->ShowAdvertisementGroup($advertisement_group_id, $xml_name, $params);
?>

```

Контекстный показ баннера в зависимости от поискового запроса, по которому пользователь пришел на сайт

Добавим в начало каждого макета код, который будет определять поисковый запрос, по которому пришел пользователь и сохранять этот запрос в cookie. При определении поискового запроса используется модуль «Статистика посещаемости». Если модуль «Статистика посещаемости» отключен или отсутствует, нижеприведенный код работать не будет. **Добавлять код необходимо в самое начало макета.**

```

<?php
if (class_exists('counter'))
{
    // Получим ссылающуюся страницу
    $HTTP_REFERER = to_str($_SERVER["HTTP_REFERER"]);

    $counter = new counter();
    $search_query_array = $counter->IsSearchSystem($HTTP_REFERER);

    $search_query = trim(to_str($search_query_array['search_query']));

    // Ограничим максимальную длину запроса (при необходимости лимит можете увеличить)
    if (strlen($search_query) <= 255)
    {
        // Сохраним cookie с поисковым запросом.
        setcookie('hostcms_search_query', $search_query, time() + 31536000, '/');
        $_COOKIE['hostcms_search_query'] = $search_query;
    }
}
?>

```

Далее остается передать текст поискового запроса в метод ShowAdvertisementGroup() или ShowBannerGroup():

```
<?php
$Advertisement = new Advertisement();

$advertisement_group_id = 3;
$xml_name = 'ОтображениеБаннера';

// Передадим в качестве контекстных запросов содержимое cookie
$params['contextual_words'] = to_str($_COOKIE['hostcms_search_query']);

// Указываем на необходимость показа только баннеров, найденных по контексту. Если ни одного
// контекстного баннера не будет найдено – ничего не будет показано.
$params['show_only_context'] = true;

$Advertisement->ShowAdvertisementGroup($advertisement_group_id, $xml_name, $params);
?>
```

Создание страницы для учета нажатий на баннер

В структуре сайта создайте страницу с именем в меню «Переход по ссылке баннера» и названием раздела «*showbanner*», отключите отображение данной страницы в меню. Выберите тип страницы «Типовая динамическая страница», из выпадающего списка «Раздел» выберите «Реклама», в выпадающем списке «Страница» выберите «Переход по ссылке баннера».

Если при нажатии на баннер Вы получите сообщение Apache, например «OK. The document has moved here», необходимо заменить серверный редирект на JavaScript, для этого в разделе «Страница» выберите «Переход по ссылке баннера (JavaScript)».

Интернет магазин

Работа с дополнительными свойствами товара в XSL

Для внедрения в XSL значений свойств используются следующие конструкции:

Значение поля:

```
<xsl:value-of select="property[@xml_name = 'FIELD_NAME']/value"/>
```

Для большой картинки:

```

```

Для малой картинки:

```

```

Вместо FIELD_NAME необходимо указать имя поля.

Работа с дополнительными свойствами группы товаров в XSL

Для внедрения в XSL значений свойств группы товаров используются следующие конструкции:

Значение поля:

```
<xsl:value-of select="propertys/property[@xml_name = 'FIELD_NAME']/value"/>
```

Для большой картинки:

```

```

Для малой картинки:

```

```

Вместо FIELD_NAME необходимо указать имя поля.

Платежная система WebMoney

Зарегистрируйтесь в системе, запомните WMID. Для приема платежей необходимо провести настройки. Перейдите по адресу <https://merchant.webmoney.ru/conf/purses.asp> и пройдите авторизацию. Обратите внимание, что необходимо иметь аттестат не ниже «Персональный аттестат».

В разделе «Список кошельков» поочередно выберите ссылку «настроить» для тех кошельков, на которые Вы планируете принимать оплату. В поле «Secret Key» внесите секретный ключ для данного

типа кошелька и запомните его, установите флажок «Позволять использовать URL, передаваемые в форме», установить «Метод формирования контрольной подписи» значение «MD5», установить «Активность» в «Вкл.», в поле «Тестовый/Рабочий режимы» выберите «Рабочий», если требуется перевод WM, а не проверка работы системы.

В настройках торгового кошелька на merchant.webmoney.ru указать в полях:

- Result URL: <http://www.site.ru/shop/cart/>
- Success URL: <http://www.site.ru/shop/cart/?payment=success>
- Fail URL: <http://www.site.ru/shop/cart/?payment=fail>

Метод передачи данных выберите POST.

merchant.webmoney.ru / Настройки / Web Merchant Interface

Настройки торгового кошелька

Кошелек:

Торговое имя: - отображается на странице при оплате

Secret Key: Высылать Secret Key на Result URL, если Result URL обеспечивает секретность

Result URL: Передавать параметры в предварительном запросе

Success URL: метод вызова Success URL

Fail URL: метод вызова Fail URL

Позволять использовать URL, передаваемые в форме:

Высылать оповещение об ошибке платежа на кипер:

Метод формирования контрольной подписи:

Тестовый/Рабочий режимы: выкл. тестовый рабочий Прием WM на кошелек ВКЛЮЧЕН.

Прием чеков Ruymeg.com (ВМ-карт) или WM-нот: Прием ВМ-карт, чеков Ruymeg или WM-нот на кошелек отключен. [Подробнее ...](#)

Прием платежей с телефонов Telepat.ru: Прием платежей с телефонов Telepat.ru на кошелек ВКЛЮЧЕН. [Подробнее ...](#)

[Список разрешенных/запрещенных стран](#)

[перейти к списку кошельков](#)

Отредактируйте обработчик платежной системы WebMoney и внесите изменения в следующие параметры:

```

/* Кошельки */
var $wmr = 'R123456789123';
var $wmz = 'Z123456789123';

/* Определяем валюты для WMR и WMZ */
var $wmr_currency_id = 2;
var $wmz_currency_id = 15;

/* Определяем коэффициенты перерасчета для WMR и WMZ */
var $wmr_coefficient_id = 1;
var $wmz_coefficient_id = 1;

/* Секретные ключи для кошельков, должны совпадать с настройками
на https://merchant.webmoney.ru/conf/purses.asp */
var $wmr_secret_key = 'hostmake';
var $wmz_secret_key = 'hostmake';

```

Обработчик уведомления платежной системы о совершении платежа размещается на типовой динамической странице «Интернет-магазин корзина»:

```
// Обработка уведомления об оплате от WebMoney
if (isset($_POST['LMI_PAYEE_PURSE']))
{
    // Получаем ID заказа
    $order_id = to_int($_POST['LMI_PAYMENT_NO']);

    $order_row = $GLOBALS['shops']->GetOrder($order_id);

    if ($order_row)
    {
        // Вызов обработчика платежной системы
        $GLOBALS['shops']->ExecSystemsOfPayHandler( $order_row['shop_system_of_pay_id']);
    }
}
```

Внимание! Код обработчика уведомления об оплате необходимо вносить только при его отсутствии в указанной типовой динамической странице.

Платежная система RBK Money

Адрес платежной системы: www.rbkmoney.ru, зарегистрироваться и войти в систему.

Заполнить поля:

- Название сайта — название сайта осуществляющего прием платежей.
- Адрес сайта — URL сайта осуществляющего прием платежей.
- Описание сайта.
- Оповещение о платеже — URL (на веб-сайте продавца), на который система RBK Money посылает HTTP POST оповещение о совершении платежа с его реквизитами. Для магазина на базе HostCMS необходимо указывать путь по следующей схеме: `http://{адрес_сайта}/{путь к магазину}/{страница корзины}/`, например, `http://www.site.ru/shop/cart/`
- Секретный ключ (32 символа) — строка символов, добавляемая к реквизитам платежа, высылаемым продавцу вместе с оповещением. Эта строка используется для повышения надежности идентификации высылаемого оповещения. Содержание строки известно только системе RBK Money и продавцу!
- Категория — выбрать из выпадающего списка.
- Способы оплаты — выберите способы оплаты, которые будет использоваться на сайте продавца.

Отредактируйте обработчик платежной системы RBK Money и внесите изменения в следующие параметры:

```
/* Идентификатор сайта в системе RBK Money, например 12345 */
var $rbkmoney_id = '12345';

/* Идентификатор валюты, в которой будет производиться платеж.
Сумма к оплате будет пересчитана из валюты магазина в указанную валюту */
var $rbkmoney_currency_id = 1;

/* Секретные ключи для кошельков, должны совпадать с настройками сайта на www.rbkmoney.ru
*/
var $rbkmoney_secret_key = 'hostmake';
```

Обработчик уведомления платежной системы о совершении платежа размещается на типовой динамической странице «Интернет-магазин корзина»:

```
// Обработка уведомления об оплате от RBK Money
if (isset($_POST['paymentStatus']))
{
    // Получаем ID заказа
    $order_id = to_int($_POST['orderId']);

    $order_row = $shop->GetOrder($order_id);

    if ($order_row)
    {
        // Вызов обработчика платежной системы
        $shop->ExecSystemsOfPayHandler($order_row['shop_system_of_pay_id']);
    }
}
```

Внимание! Код обработчика уведомления об оплате необходимо вносить только при его отсутствии в указанной типовой динамической странице.

Платежная система ASSIST

ASSIST — это мультибанковская система платежей по пластиковым и виртуальным картам через интернет, позволяющая в реальном времени производить авторизацию и обработку транзакций.

Адрес платежной системы: <http://www.assist.ru/>.

Отредактируйте обработчик платежной системы ASSIST и внесите изменения в следующие параметры:

```
/* Идентификатор сайта (Shop_IDP) в системе ASSIST, например 123456 */
var $Shop_IDP = '123456';

/* Идентификатор валюты.
Указывается ID валюты рубли (RUR)
*/
var $assist_currency_id = 2;

/*
Режим работы магазина:
true - тестовый режим;
false - боевой режим (не забудьте изменить режим магазина в системе ASSIST).
*/
var $test_mode = true;
```

При переводе магазина в боевой режим не забудьте изменить режим в системе ASSIST и в обработчике платежной системы.

В «Настройки магазина» на сайте <https://secure.assist.ru/members/> заполнить поля:

- Режим работы — «Рабочий» или «Тестовый».
- Получение результатов — «Получать результаты в базовой валюте магазина» или «Получать результаты в оригинальной валюте.»
- URL для возврата покупателя в магазин при успешной авторизации — URL (на веб-сайте продавца), на который система Assist перенаправляет пользователя после успешной оплаты. Для магазина на базе HostCMS необходимо указывать путь по следующей схеме:
http://{адрес_сайта}/{путь_к_магазину}/{страница_корзины}/, например,
<http://www.site.ru/shop/cart/>

- URL для возврата покупателя в магазин во всех остальных случаях — URL (на веб-сайте продавца), на который система Assist перенаправляет пользователя в случае других статусов оплаты.
Для магазина на базе HostCMS необходимо указывать путь по следующей схеме:
`http://{адрес_сайта}/{путь к магазину}/{страница корзины}/`, например,
<http://www.site.ru/shop/cart/>
- Укажите адреса эл. почты для уведомлений.

Автоматическое получение подтверждения оплаты

Для установки статусов оплаты заказов, оплаченных через систему ASSIST, необходимо использовать файл `assist_orders.php`. Для автоматизации запросов добавьте вызов файла `cron/assist_orders.php` в планировщик задач.

Информацию о планировщике задач можно получить на странице <http://www.hostcms.ru/documentation/crontab/>

Платежная система Яндекс.Деньги

Зарегистрируйтесь в системе, запомните номер кошелька.

Отредактируйте обработчик платежной системы Яндекс.Деньги и внесите изменения в следующие параметры. Данные для оплаты через «Интернет.Кошелек» используются при прямой оплате, данные при оплате через «Яндекс.Деньги» используются при платеже через сайт Яндекс.Деньги.

Данные, вводимые для оплаты через «Яндекс.Деньги» пользователь получает после подписания официального договора с системой электронных платежей.

```
// -----
// Для оплаты через "Интернет.Кошелек"
// -----

// Идентификатор валюты. Указывается ID валюты рубли (RUR)
var $ym_currency_id = 1;

// Коэффициент увеличения цены при оплате Яндекс.Деньгами
var $ym_coefficient = 1;

/* Идентификатор счета получателя для оплаты на кошелек */
var $ym_account_id = "1234567891234";

/* Адрес магазина*/
var $ym_shop_address = 'admin@localhost.ru';

var $ym_shop_key_id = '';
var $ym_encryption_key = '';

// -----
// Для оплаты через "Яндекс.Деньги"
// -----

// код валюты - рубли
//var $TargetCurrency = 643;
var $TargetCurrency = 10643;

// код валюты - рубли
//var $currency = 643;
var $currency = 10643;
```

```

// идентификатор процессингового центра платежной системы
//var $BankID = 100;
var $BankID = 1003;

// идентификатор процессингового центра платежной системы
//var $TargetBankID = 1001;
var $TargetBankID = 1003;

// тип платежа: по технологии PayCash
var $PaymentTypeCD = 'PC';

/* Идентификатор магазина в ЦПП - уникальное значение,
присваивается Магазину платежной системой после заключения договора */
var $ym_shop_id = 0;

// Адрес платежной системы
//var $action = 'http://money.yandex.ru/select-wallet.xml';
var $action = 'http://demomoney.yandex.ru/select-wallet.xml';

```

При переводе магазина в боевой режим не забудьте изменить соответствующие идентификаторы и адрес платежной системы.

Прием платежей через ROBOKASSA

Адрес системы: <http://www.robokassa.ru/>, зарегистрируйтесь и войдите в систему. Создайте аккаунт магазина, заполните секретные коды.

В качестве страниц для уведомления о совершении платежа ResultURL, SuccessURL и FailURL укажите метод передачи POST и адрес: http://www.ваш_сайт.ru/shop/cart/

Через центр администрирования HostCMS в справочнике «Платежные системы» добавьте новую платежную систему «ROBOKASSA» или отредактируйте, если она уже существует.

В обработчике платежной системы ROBOKASSA внесите изменения в следующие параметры:

```

/*>Login в системе ROBOKASSA */
var $mrh_login = "mylogin";

/* Пароль #1 - Используется интерфейсом инициализации оплаты */
var $mrh_pass1 = "1234567a";

// регистрационная информация (пароль #2)
// registration info (password #2)
var $mrh_pass2 = "98765432a";

```

Обработчик уведомления платежной системы о совершении платежа размещается в настройках типовой динамической странице «Интернет-магазин корзина»:

```

// -----
// Обработка уведомления об оплате от ROBOKASSA
// должно быть только в настройках типовой дин. страницы
// -----
if (isset($_REQUEST['SignatureValue'])
// для отличия от SuccessURL/FailURL
&& !isset($_REQUEST['Culture']))
{
    // Получаем ID заказа

```



```

$order_id = to_int($_REQUEST['InvId']);

$order_row = $shop->GetOrder($order_id);

if ($order_row)
{
    // Вызов обработчика платежной системы
    $shop->ExecSystemsOfPayHandler($order_row['shop_system_of_pay_id']);
}

exit();
}
// -----

```

Обработчик уведомления пользователя о совершении платежа размещается на типовой динамической странице «Интернет-магазин корзина»:

```

// -----
// Обработка уведомления об оплате от ROBOKASSA
// -----
if (isset($_REQUEST['SignatureValue']) && isset($_REQUEST['Culture']))
{
    // Получаем ID заказа
    $order_id = to_int($_REQUEST['InvId']);

    $order_row = $shop->GetOrder($order_id);

    if ($order_row)
    {
        // Вызов обработчика платежной системы
        $shop->ExecSystemsOfPayHandler($order_row['shop_system_of_pay_id']);
    }
}
// -----

```

Внимание! Код обработчика уведомления об оплате необходимо вносить только при его отсутствии в указанной типовой динамической странице.

Обработчики платежных систем

Система управления имеет открытый интерфейс для внедрения обработчиков различных платежных систем. Обработчик представлен классом `system_of_pay_handler`, имеющим следующую структуру:

```

<?php
class system_of_pay_handler
{
    function ProcessResult()
    {
        // Обработчик уведомления платежной системы о совершении платежа
    }

    /**
     * Отображает стартовую страницу для оплаты *
     */
    function ShowPurseRequest()
    {
        $shop_id = to_int($GLOBALS['shop_id']);
    }
}

```

```

/* Получаем id текущего пользователя сайта */
if (class_exists('SiteUsers'))
{
    /* Получаем id текущего пользователя сайта */
    $SiteUsers = new SiteUsers();
    $site_users_id = $SiteUsers->GetCurrentSiteUser();
}
else
{
    $site_users_id = false;
}

/* ID платежной системы берем из сессии */
$system_of_pay_id = to_int($_SESSION['system_of_pay_id']);

// статус платежа, по умолчанию 0
$order_row['status_of_pay'] = 0 ;

// дата платежа, по умолчанию пустая строка
$order_row['date_of_pay'] = '';

// описание и системная информация, по умолчанию пустая строка
$order_row['description'] = '';

/* Оформляем заказ */
$order_id = $GLOBALS['shops']->ProcessOrder($shop_id, $site_users_id,
$system_of_pay_id, $order_row);

if ($order_id > 0)
{
    if (!class_exists('SiteUsers'))
    {
        /* Класс пользователей сайта не существует, дописываем информацию о
заказчике
в поле shop_order_description из текущей сессии */
        if ($order_row)
        {
            /* E-Mail заказчика */
            $user_email = trim(to_str($_SESSION['site_users_email']));

            /* Описание заказчика */
            $description = "Информация о заказчике:\n"
            ."Имя: ".to_str($_SESSION['site_users_name'])."\n"
            ."Фамилия: ".to_str($_SESSION['site_users_surname'])."\n"
            ."Отчество: ".to_str($_SESSION['site_users_patronymic'])."\n"
            ."E-Mail: ".to_str($_SESSION['site_users_email'])."\n"
            ."Телефон: ".to_str($_SESSION['site_users_phone'])."\n"
            ."Факс: ".to_str($_SESSION['site_users_fax'])."\n"
            ."Адрес: ".to_str($_SESSION['site_users_address'])."\n";

            $order_row['description'] = to_str($description);

            /* Обязательно добавляем идентификатор! */
            $order_row['id'] = $order_id;

            $GLOBALS['shops']->InsertOrder($order_row);
        }
    }
    else
    {
        $user_email = false;
    }

    if ($order_row)
    {

```

```

        $shop_row = $GLOBALS['shops']->GetShop($GLOBALS['shop_id']);

        /* Информация об алиасе сайта */
        $site = new site();
        $site_alias = $site->GetCurrentAlias($shop_row['site_id']);

        /* Получаем путь к магазину */
        $Structure = new Structure();
        $shop_path = "/" . $Structure-
>GetStructurePath($shop_row['structure_id'], 0);

        $handler_url = 'http://' . $site_alias . $shop_path . 'cart/';

        ?>
        // Здесь выводится форма для начала оплаты
        <?php

    }

    /* Отправляем письмо заказчику */
    $GLOBALS['shops']->SendMailAboutOrder($shop_id, $order_id, $site_users_id,
    $GLOBALS['LA']['xsl_letter_to_admin'], $GLOBALS['LA']['xsl_letter_to_user'], $user_email,
    array('admin-content-type' => 'html', 'user-content-type' => 'html'));
    }
    else
    {
        switch ($order_id)
        {
            case -1:
                {
                    echo "Ошибка вставки заказа в базу данных. Обратитесь
к администратору.";
                    break;
                }
            case -2:
                {
                    echo "Ошибка - не найден магазин. Обратитесь к
администратору.";
                    break;
                }
        }
    }
}

/**
 * Метод, запускающий выполнение обработчика
 */
function Execute()
{
    /* Пришло подтверждение оплаты, обрабатываем его */
    // Условие необходимо заменить на соответствующее внедряемой платежной системе
    if (isset($_POST['LMI_PAYEE_PURSE']))
    {
        $this->ProcessResult();
        return true;
    }

    /* Иначе оформляем заказ и отображаем стартовую страницу для оплаты через WebMoney
*/
    $this->ShowPurseRequest();
}

/**
 * Метод для отображения формы заказа для печати.
 *
 * @param int $order_id идентификатор заказа

```

```

    */
    function PrintOrder($order_id)
    {
    }
}
?>

```

Рассмотрим класс более подробно, метод `Execute()` вызывается при вызове оплаты через платежную систему. В методе необходимо определить, получили ли мы подтверждение от платежной системы об осуществленной оплате или только хотим произвести оплату.

Необходимо ознакомиться с API по внедрению платежной системы и выяснить, какие данные и каким методом она пересылает в Интернет-магазин по факту оплаты. Например, для системы Webmoney это будет атрибут `LMI_PAYEE_PURSE`, направляемый методом `POST`. Проверка, в данном случае, будет выглядеть следующим образом:

```

if (isset($_POST['LMI_PAYEE_PURSE']))
{
    $this->ProcessResult();
    return true;
}

```

Если подтверждения оплаты не было, значит пользователь только планирует произвести оплату, для вывода формы оплаты используется метод `ShowPurseRequest()`, например:

```

$this->ShowPurseRequest();

```

Формирование страницы оплаты

В методе `ShowPurseRequest` осуществляется расчет стоимости при оплате в разных валютах и производится вывод формы оплаты. Рассмотрим более подробно:

В самом начале необходимо получить идентификатор заказа и определить текущего авторизованного пользователя, далее определим идентификатор платежной системы и заполним множество других атрибутов заказа. В случае, если пользователи сайтов отсутствуют (например в редакциях «Халява» и «Малый бизнес»), данные о заказчике внесем в поле описание заказа.

В случае, если удалось получить данные о заказе по его идентификатору, проведем расчеты стоимости заказа в валюте (или валютах) платежной системы. Для удобства настройки всевозможные коэффициенты пересчета, секретные ключи и номера кошельков выносятся в свойства класса.

Для определения пути, по которому платежная система уведомит магазин о совершении платежа используется следующая конструкция:

```

/* Информация об алиасе сайта */
$site = new site();
$site_alias = $site->GetCurrentAlias($shop_row['site_id']);

/* Получаем путь к магазину */
$Structure = new Structure();
$shop_path = "/" . $Structure->GetStructurePath($shop_row['structure_id'], 0);
$handler_url = 'http://' . $site_alias . $shop_path . 'cart/';

```

Далее выводится форма для перехода на сайт платежной системы, поля формы заполняются в соответствии с требованиями конкретной платежной системы.

Не забываем отправить письмо администратору и заказчику с информацией, что его заказ принят:

```
/* Отправляем письмо заказчику */
$GLOBALS['shops']->SendMailAboutOrder($shop_id, $order_id, $site_users_id,
$GLOBALS['LA']['xsl_letter_to_admin'], $GLOBALS['LA']['xsl_letter_to_user'], $user_email,
array('admin-content-type' => 'html', 'user-content-type' => 'html'));
```

Обработка уведомления платежной системы об оплате

Обработка уведомления является важной подсистемой магазина. Большинство платежных системы после оплаты вызывают определенным методом (в зависимости от платежной системы) страницу, на которую передают информацию о совершенном платеже. Адрес этой страницы в большинстве случаев настраивается в форме заказа или в атрибутах магазина (кошелька) в платежной системе.

Обработка уведомления осуществляется методом `ProcessResult()`, вызываемом из метода `Execute()`.

В самом начале метода необходимо получить номер заказа и удостовериться, что заказ существует. Для проверки корректности принятых от платежной системы данных, многие платежные системы поддерживают проверку контроля целостности передаваемых данных, для этого используются контрольные сумму, рассчитываемые по разным алгоритмам – CRC32, MD5 и т.д., например:

```
/* Сравниваем хэши */
if (strtoupper($str) == strtoupper(to_str($_POST['LMI_HASH'])))
{
    /* Проверка прошла успешно!
    Добавляем комментарий */
    $param['system_information'] = "Товар оплачен через WebMoney.\n".
    "Атрибуты:\n".
    "Кошелек продавца: {$_POST['LMI_PAYEE_PURSE']}\n".
    "Сумма: {$_POST['LMI_PAYMENT_AMOUNT']} {$currency_name}\n".
    "Номер покупки: {$_POST['LMI_PAYMENT_NO']}\n".
    "Режим (0 - реальный, 1 - тестовый): {$_POST['LMI_MODE']}\n".
    "Номер счета для покупателя: {$_POST['LMI_SYS_INVS_NO']}\n".
    "Номер платежа: {$_POST['LMI_SYS_TRANS_NO']}\n".
    "Кошелек покупателя: {$_POST['LMI_PAYER_PURSE']}\n".
    "WM-идентификатор покупателя: {$_POST['LMI_PAYER_WM']}";

    /* Устанавливаем признак оплаты */
    $param['date_of_pay'] = date("Y-m-d H:i:s");
    $param['status_of_pay'] = true;
}
else
{
    $param['system_information'] = 'WM хэш не совпал!';
}

// Обновляем информацию о заказе
$GLOBALS['shops']->InsertOrder($param);
```

Проверив корректность данных их необходимо записать в атрибуты заказа, при этом если данные корректны — установить статус и дату оплаты:

```
/* Устанавливаем признак оплаты */
```

```
$param['date_of_pay'] = date("Y-m-d H:i:s");
$param['status_of_pay'] = true;
```

Не забываем уведомить пользователя и администратора о получении оплаты:

```
// Отправляем письмо администратору о подтверждении платежа
$GLOBALS['shops']->SendMailAboutOrder($order_row['shop_shops_id'], $order_id,
$order_row['site_users_id'],
$GLOBALS['LA']['xsl_letter_to_admin'], $GLOBALS['LA']['xsl_letter_to_user'],
$order_row['shop_order_users_email'],
array(
'admin-content-type' => 'html',
'user-content-type' => 'html',
'admin-subject' => 'Подтверждение оплаты, заказ '. $order_id,
'user-subject' => 'Подтверждение оплаты, заказ '. $order_id));
```

Формирование страницы оплаты

В методе ShowPurseRequest осуществляется расчет стоимости при оплате в разных валютах и производится вывод формы оплаты. Рассмотрим более подробно:

В самом начале необходимо получить идентификатор заказа и определить текущего авторизованного пользователя, далее определим идентификатор платежной системы и заполним множество других атрибутов заказа. В случае, если пользователи сайтов отсутствуют (например, в редакциях «Халыва» и «Малый бизнес»), данные о заказчике внесем в поле описание заказа.

В случае, если удалось получить данные о заказе по его идентификатору, проведем расчеты стоимости заказа в валюте (или валютах) платежной системы. Для удобства настройки всевозможные коэффициенты пересчета, секретные ключи и номера кошельков выносятся в свойства класса.

Для определения пути, по которому платежная система уведомит магазин о совершении платежа используется следующая конструкция:

```
/* Информация об алиасе сайта */
$site = new site();
$site_alias = $site->GetCurrentAlias($shop_row['site_id']);

/* Получаем путь к магазину */
$Structure = new Structure();
$shop_path = "/" . $Structure->GetStructurePath($shop_row['structure_id'], 0);
$handler_url = 'http://' . $site_alias . $shop_path . 'cart/';
```

Далее выводится форма для перехода на сайт платежной системы, поля формы заполняются в соответствии с требованиями конкретной платежной системы.

Не забываем отправить письмо администратору и заказчику с информацией, что его заказ принят:

```
/* Отправляем письмо заказчику */
$GLOBALS['shops']->SendMailAboutOrder($shop_id, $order_id, $site_users_id,
$GLOBALS['LA']['xsl_letter_to_admin'], $GLOBALS['LA']['xsl_letter_to_user'], $user_email,
array('admin-content-type' => 'html', 'user-content-type' => 'html'));
```

Обработка уведомления платежной системы об оплате

Обработка уведомления является важной подсистемой магазина. Большинство платежных системы после оплаты вызывают определенным методом (в зависимости от платежной системы) страницу, на которую передают информацию о совершенном платеже. Адрес этой страницы в большинстве случаев настраивается в форме заказа или в атрибутах магазина (кошелька) в платежной системе.

Обработка уведомления осуществляется методом `ProcessResult()`, вызываемом из метода `Execute()`.

В самом начале метода необходимо получить номер заказа и удостовериться, что заказ существует. Для проверки корректности принятых от платежной системы данных, многие платежные системы поддерживают проверку контроля целостности передаваемых данных, для этого используются контрольные суммы, рассчитываемые по разным алгоритмам – CRC32, MD5 и т.д., например:

```

/* Сравниваем хэши */
if (strtoupper($str) == strtoupper(to_str($_POST['LMI_HASH'])))
{
    /* Проверка прошла успешно!
    Добавляем комментарий */
    $param['system_information'] = "Товар оплачен через WebMoney.\n".
    "Атрибуты:\n".
    "Кошелек продавца: {$_POST['LMI_PAYEE_PURSE']}\n".
    "Сумма: {$_POST['LMI_PAYMENT_AMOUNT']} {$currency_name}\n".
    "Номер покупки: {$_POST['LMI_PAYMENT_NO']}\n".
    "Режим (0 - реальный, 1 - тестовый): {$_POST['LMI_MODE']}\n".
    "Номер счета для покупателя: {$_POST['LMI_SYS_INVS_NO']}\n".
    "Номер платежа: {$_POST['LMI_SYS_TRANS_NO']}\n".
    "Кошелек покупателя: {$_POST['LMI_PAYER_PURSE']}\n".
    "WM-идентификатор покупателя: {$_POST['LMI_PAYER_WM']}";

    /* Устанавливаем признак оплаты */
    $param['date_of_pay'] = date("Y-m-d H:i:s");
    $param['status_of_pay'] = true;
}
else
{
    $param['system_information'] = 'WM хэш не совпал!';
}

// Обновляем информацию о заказе
$GLOBALS['shops']->InsertOrder($param);

```

Проверив корректность данных их необходимо записать в атрибуты заказа, при этом если данные корректны — установить статус и дату оплаты:

```

/* Устанавливаем признак оплаты */
$param['date_of_pay'] = date("Y-m-d H:i:s");
$param['status_of_pay'] = true;

```

Не забываем уведомить пользователя и администратора о получении оплаты:

```

// Отправляем письмо администратору о подтверждении платежа
$GLOBALS['shops']->SendMailAboutOrder($order_row['shop_shops_id'], $order_id,
$order_row['site_users_id'],
$GLOBALS['LA']['xsl_letter_to_admin'], $GLOBALS['LA']['xsl_letter_to_user'],
$order_row['shop_order_users_email'],
array(

```

```
'admin-content-type' => 'html',  
'user-content-type' => 'html',  
'admin-subject' => 'Подтверждение оплаты, заказ '. $order_id,  
'user-subject' => 'Подтверждение оплаты, заказ '. $order_id));
```

Публикация интернет-магазина на сайте

Созданный в системе управления магазин необходимо опубликовать на сайте. Перейдите в раздел «Структура сайта» и создайте новый узел. Укажите для узла путь, например *shop*, заполните основные атрибуты узла, тип раздела укажите «Типовая динамическая страница», выберите раздел типовой динамической страницы «Интернет-магазин», страницу «Интернет-магазин». Укажите параметры динамической страницы согласно предоставленному ниже примеру.

Родительский раздел	...
Раздел меню	Верхнее Меню
Отображать в меню сайта	<input checked="" type="checkbox"/>
Шаблон страницы	Шаблон для Интернет-магазина
Активность страницы	<input checked="" type="checkbox"/>
Индексировать	<input checked="" type="checkbox"/>
Название раздела (только латинские буквы и цифры)	shop
Сортировка для текущего уровня	20
Группа доступа	Все
Тип раздела	<input type="radio"/> Страница <input checked="" type="radio"/> Типовая динамическая страница <input type="radio"/> Динамическая страница
Макет	Основной макет сайта
Раздел	Интернет-магазин
Страница	Интернет-магазин
Идентификатор магазина	Демонстрационный магазин [1]
XSL каталога	Интернет-магазин МагазинКаталогТоваров
XSL товара	Интернет-магазин МагазинТовар
XSL письма администратору	Интернет-магазин ПисьмоАдминистратору
XSL письма пользователю	Интернет-магазин ПисьмоПользователю

Создайте раздел «Корзина», родительским узлом для которого будет являться узел «Интернет магазин», укажите для него название раздела *cart*, заполните основные атрибуты страницы, выберите раздел типовой динамической страницы «Интернет-магазин», страницу «Интернет-магазин корзина». Укажите параметры динамической страницы согласно предоставленному ниже примеру.

Родительский раздел	
Интернет-магазин	
Раздел меню	
Верхнее Меню	
Отображать в меню сайта <input checked="" type="checkbox"/>	
Шаблон страницы	
Основной	
Активность страницы <input checked="" type="checkbox"/>	
Индексировать <input checked="" type="checkbox"/>	
Название раздела (только латинские буквы и цифры)	
cart	
Сортировка для текущего уровня	
0	
Группа доступа	
Как у родителя	
Тип раздела	
<input type="radio"/> Страница <input checked="" type="radio"/> Типовая динамическая страница <input type="radio"/> Динамическая страница	
Макет	
Основной макет сайта	
Раздел	
Интернет-магазин	
Страница	
Интернет-магазин корзина	
Идентификатор магазина	Демонстрационный магазин [1]
XSL адреса доставки	Интернет-магазин
	МагазинАдресДоставки
XSL платежной системы	Интернет-магазин
	МагазинПлатежнаяСистема
XSL оформления заказа	Интернет-магазин
	ОформлениеЗаказа
XSL письма администратору	Интернет-магазин
	ПисьмоАдминистратору
XSL письма пользователю	Интернет-магазин
	ПисьмоПользователю
XSL корзины	Интернет-магазин
	МагазинКорзина
XSL письма подтверждения регистрации	Пользователи сайта
	ПисьмоПодтверждениеРегистрации
XSL быстрой регистрации	Интернет-магазин
	МагазинБыстраяРегистрация
XSL магазина доставки	Интернет-магазин
	МагазинДоставки

Для организации возможности пользователям магазина распечатывать квитанции, создайте под узлом корзины узел «Версия для печати», укажите для него название раздела *print*, заполните основные атрибуты страницы, выберите раздел типовой динамической страницы «Интернет-магазин», страницу «Версия для печати».

Родительский раздел	Корзина
Раздел меню	Верхнее Меню
Отображать в меню сайта	<input checked="" type="checkbox"/>
Шаблон страницы	Основной
Активность страницы	<input checked="" type="checkbox"/>
Индексировать	<input checked="" type="checkbox"/>
Название раздела (только латинские буквы и цифры)	print
Сортировка для текущего уровня	0
Группа доступа	Как у родителя
Тип раздела	<input type="radio"/> Страница <input checked="" type="radio"/> Типовая динамическая страница <input type="radio"/> Динамическая страница
Макет	Основной макет сайта
Раздел	Интернет-магазин
Страница	Версия для печати
<input checked="" type="checkbox"/> Типовая динамическая страница не содержит параметров.	

Автоматическое построение прайс-листа для магазина

Система управления имеет возможность построения прайс-листа по всему магазину для публикации в сокращенном варианте. Для публикации Прайс листа создайте узел с именем «Прайс-лист», например, под узлом «Интернет магазин». Укажите для него название раздела *price*, заполните основные атрибуты страницы, выберите раздел типовой динамической страницы «Интернет-магазин», страницу «Прайс», укажите идентификатор магазина, для которого отображается Прайс-лист и XSL-шаблон для отображения прайс-листа.

Раздел	Интернет-магазин
Страница	Прайс
XSL шаблон	Интернет-магазин
	МагазинПрайс
Идентификатор магазина	Демонстрационный магазин [1]

Вывод информации о продавцах

Вывод информации о продавцах используется при наличии нескольких продавцов в магазине, например в магазине-посреднике.

Создайте узел с именем «Продавцы», в качестве родительского узла укажите «Интернет магазин». Укажите для созданного узла название раздела *sellers*, заполните основные атрибуты страницы, выберите раздел типовой динамической страницы «Интернет-магазин», страницу «Продавцы», укажите XSL-шаблон для отображения информации о продавце.

Раздел	Интернет-магазин
Страница	Продавцы
XSL продавцов	Интернет-магазин
	МагазинПродавец

Сравнение товаров

Создайте узел с именем «Сравнение товаров», в качестве родительского узла укажите «Интернет магазин». Укажите для созданного узла название раздела *compare_items*, заполните основные атрибуты страницы, выберите раздел типовой динамической страницы «Интернет-магазин», страницу «Сравнение товаров».

Раздел	Интернет-магазин
Страница	Сравнение товаров
XSL сравнения товаров	Интернет-магазин
	СравнениеТоваров
Идентификатор магазина	Демонстрационный магазин [1]

Экспорт в Яндекс.Маркет

Создайте узел в структуре сайта под узлом интернет-магазина, укажите для созданного узла название раздела, например *yandex_market*, заполните основные атрибуты страницы, выберите раздел типовой динамической страницы «Интернет-магазин», страницу «Экспорт в Яндекс.Маркет». Выберите из выпадающего списка магазин, для которого необходимо создать экспорт.

Раздел	Интернет-магазин
Страница	Экспорт в Яндекс.Маркет
Идентификатор магазина	Демонстрационный магазин [1]

В Яндекс.Маркет добавляете адрес страницы, например, http://www.site.ru/shop/yandex_market/

Кэширование

Управление блоками кэширования

Для хранения однотипной информации создаются кэши с некоторыми именами.

Атрибуты элемента кэша задаются в настройках модуля виде массива. Ниже дан пример создания кэша с именем **TMP**:

```
<?php
$GLOBALS['CACHE_CONFIG']['TMP']['name'] = 'IP-адреса';
$GLOBALS['CACHE_CONFIG']['TMP']['block'] = 1000; // Максимальное число элементов в кэше
$GLOBALS['CACHE_CONFIG']['TMP']['block_size'] = 512; // В байтах
$GLOBALS['CACHE_CONFIG']['TMP']['expires'] = 900; // В секундах
$GLOBALS['CACHE_CONFIG']['TMP']['active'] = true; // Активность
$GLOBALS['CACHE_CONFIG']['TMP']['type'] = 'eaccelerator';
?>
```

`$GLOBALS['CACHE_CONFIG']['ИМЯ']` — имя кэша, в котором будут сохраняться элементы. Используется при работе с кэшем через API системы.

`$GLOBALS['CACHE_CONFIG']['ИМЯ']['name']` — текстовое имя кэша, используется для вывода в центре администрирования.

`$GLOBALS['CACHE_CONFIG']['ИМЯ']['block']` — предполагаемое максимальное число элементов в кэше (индикативное значение, не влияющее на реальный размер кэша).

`$GLOBALS['CACHE_CONFIG']['ИМЯ']['block_size']` — максимальный размер элемента кэша, указывается в байтах.

`$GLOBALS['CACHE_CONFIG']['ИМЯ']['active']` — указывает на возможность работы с кэшем, true — разрешено, false — запрещено.

`$GLOBALS['CACHE_CONFIG']['ИМЯ']['type']` — тип хранилища, указывает место размещения кэша.

Кэшируемые данные могут размещаться в различных хранилищах, доступных на площадке:

- **file** — данные сохраняются в файлах (хранилище по умолчанию);
- **eaccelerator** — хранение данных передается eAccelerator, при этом данные хранятся в памяти или в файлах на диске;
- **xcache** — хранение данных передается XCache, при этом данные хранятся в памяти.

Сохранение значения в кэше

Сохраняемые значения в кэше имеют некоторое имя, по которому могут быть извлечены. Наиболее часто в качестве имени элемента используется идентификатор или строка идентификаторов.

```
<?php
$Cache = & singleton('Cache');

// Имя элемента, наиболее часто используется идентификатор или строка идентификаторов
$item_name = 'my_item_1';

$value = "Некое значение, может быть разных типов";
```

```
// Для кэширования различных элементов используются различные кэши. Список кэшей задается в
// настройках модуля "Кэширование"
$cache_name = "TMP";

$result = $Cache->Insert($item_name, $value, $cache_name);

if ($result)
{
    echo "Вставка выполнена успешно";
}
else
{
    echo "Ошибка вставки";
}
?>
```

Извлечение информации из кэша

Извлечение информации из кэша осуществляется по имени кэша и имени элемента, сохраненного в кэше.

```
<?php
$Cache = & singleton('Cache');

// Имя элемента, наиболее часто используется идентификатор или строка идентификаторов
$item_name = 'my_item_1';

// Для кэширования различных элементов используются различные кэши. Список кэшей задается в
// настройках модуля "Кэширование"
$cache_name = "TMP";

if ($in_cache = $Cache->GetCacheContent($item_name, $cache_name))
{
    $value = $in_cache['value'];

    // В $value содержится закэшированный элемент
    echo $value;
}
else
{
    echo "Элемент в кэше не найден!";
}
?>
```

Организация страниц для печати

В макете сайта в самом начале необходимо прописать код, который будет отображать содержание страницы в формате, удобном для печати.

```
<?php
// Проверяем, если нажали ссылку "Печать"
if(isset($_GET['action']) && $_GET['action'] == "print" )
{
// Выводим только содержание страницы (версия для печати)
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title><?$_kernel->show_title()?></title>
<meta name="description" content="<?$_kernel->show_description()?>">
<meta name="keywords" content="<?$_kernel->show_keywords()?>">
<meta http-equiv="Content-Language" content="ru">
<meta content="text/html; charset=windows-1251" http-equiv=Content-Type>
<?$_kernel->show_CSS()?>
</head>
<body topmargin="0" leftmargin="0" marginwidth="0" marginheight="0">
  <?
  /* Устанавливаем шаблон для текущей страницы
  Значение 10 замените на ID шаблона страницы (не макета), в нем сверху
  и снизу можете добавить шапку и подвал с копирайтом */
  $_kernel->set_current_page_data_template(10);

  // Вызов шаблона для текущей страницы
  $_kernel->show_current_template();
?>
</body>
</html>
<?php
// Вывод версии для печати закончен
return;
}
?>
```

В указанном коде необходимо заменить идентификатор шаблона страницы на нужный.

Ссылка на версию для печати будет выглядеть следующим образом:

```
<a href="./?action=print">Версия для печати</a>
```

Использование кода подтверждения для защиты от автоматического заполнения форм (Captcha)

Последнее время защита от автоматического заполнения форм стала намного актуальнее.

Отображением защитного кода занимается объект класса `Captcha`.

Внедрение в HTML страничку производится с помощью тега ``:

```

```

где `x1` – числовое значение, сгенерированное с помощью метода `GetCaptchaId()` класса `Captcha`.

Пример внедрения кода:

```
<?php
$Captcha = new Captcha();
$x1 = $Captcha->GetCaptchaId();
?>
<form name="myform" action="/mypath/" method="POST">

...

<!-- Код подтверждения -->

<input name="captcha_key" value="<?=$x1?>" type="hidden">

Введите код подтверждения: <input name="captcha_value" type="text">
<!-- //Код подтверждения -->
...

<input type="submit" name="button1" />

</form>
```

Пример проверки правильности введенного кода:

```
<?php
$Captcha = new Captcha();

/* Внимание! Если Вы передаете данные методом GET, замените $_POST на $_GET */

/* $captcha_result будет содержать true или false */
$captcha_result = $Captcha->ValidCaptcha($_POST['captcha_key'], $_POST['captcha_value']);

if ($captcha_result)
{
    ?>
    Код верный.
    <?php
}
else
{
    ?>
    Введите код подтверждения еще раз.
```



```
<?php  
}  
?>
```

Импорт RSS-каналов

Импорт RSS-каналов наиболее часто происходит с некоторой периодичностью. Периодичность вызова организуется с помощью Cron/crontab¹.

Необходимо создать файл, который будет вызываться с некоторой периодичностью и осуществлять импорт данных из RSS-канала.

Файл будет начинаться с подключения основных классов и инициализации модулей.

```
/* Подключаем основные классы */
require_once('main_classes.php');

/* Загружаем модули */
$GLOBALS['kernel']->LoadModules();
```

Чтение RSS-канала осуществляется с использованием метода *ReadRSS* класса *RssRead()*.

Пример импорта RSS-канала:

```
<?php

ini_set('display_errors', 1);

/* Подключаем основные классы */
require_once('main_classes.php');

/* Загружаем модули */
$GLOBALS['kernel']->LoadModules();

/* Адрес RSS-канала */
$url = 'http://www.vsesmi.ru/rss/all/';

/* Идентификатор информационной системы, в которую помещаются элементы */
$info_id = 21;

/* Группа, в которую помещается новый элемент */
$info_group_id = 0;

$rss = new RssRead();

$result = $rss->ReadRSS($url);

$infoSystem = new InformationSystem();

$dateClass = new DateClass();

/* Цикл по полученным элементам */
for($i = 0; $i < count($result['items']) - 1; $i++)
{
```

¹ Cron/crontab — система для автоматического запуска программ и скриптов на сервере в определенное время. Более подробную информацию смотрите на сайте <http://www.hostcms.ru/documentation/crontab/>

```
/* Если не найдено элементов с таким же именем */
if (mysql_num_rows(
    $InformationSystem->GetExternalInformationSystemItem
(array('information_items_name'=>$result['items'][$i]['title'],
      'information_systems_id'=>$infsys_id))) == 0)
{
    /* Формируем полный путь к источнику материала */
    $link = $result['items'][$i]['link'];

    /* Заголовок */
    $title = $result['items'][$i]['title'];

    /* Тест элемента */
    $desc = $result['items'][$i]['desc'];

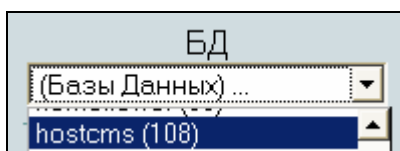
    /* Описание элемента */
    $text = $result['items'][$i]['desc'];

    /* Дата */
    $date = $DateClass->DateUnixToSQL (strtotime ($result['items'][$i]['pubdate']));

    if (!empty($title))
    {
        /* Вставка элемента */
        $InformationSystem->InsertInformationItems(0, 0, $infsys_id, $infgroup_id,
        $date, $title, $desc, 1, $text, '', 0, $_SERVER['REMOTE_ADDR']);
    }
}
?>
```

Восстановление пароля администратора в случае его утери

Для восстановления пароля администратора в случае его утери необходимо иметь доступ к PhpMyAdmin или другому веб-интерфейсу для администрирования СУБД MySQL. Подключитесь к СУБД, выберите нужную базу данных в левом поле.



В открывшемся списке таблиц выберите обзор таблицы "users_table".

<input type="checkbox"/>	users_access_denied_table						
<input type="checkbox"/>	users_access_table						
<input type="checkbox"/>	users_table						
<input type="checkbox"/>	users_type_table						

В столбце "User_name" найдите пользователя с Вашим логином и нажмите для него "редактировать".

	users_id	users_type_id	users_name	users_password	users_superuser
	19	3	admin	21232f297a57a5a743894a0e4a801fc3	1

В открывшемся окне для поля "users_password" введите значение 21232f297a57a5a743894a0e4a801fc3, которое соответствует паролю "admin".

Поле	Тип	Функция	Ноль	Значение
users_id	int(10) unsigned	<input type="text"/>		19
users_type_id	int(11)	<input type="text"/>		3
users_name	varchar(255)	<input type="text"/>	<input type="checkbox"/>	admin
users_password	varchar(255)	<input type="text"/>	<input type="checkbox"/>	21232f297a57a5a743894a0e4a801fc3
users_superuser	tinyint(4)	<input type="text"/>		1

Авторизуйтесь в системе с использованием Вашего логина и пароля admin, после чего обязательно измените пароля для этого логина.

Очищение списка неудачных попыток авторизации

Система управления имеет механизм защиты от подбора данных для авторизации. При использовании большого числа неудачных попыток авторизации, система будет предлагать подождать все большее время. Если Вы совершили большое число неудачных попыток авторизации и не можете зайти в систему управления, очистите через PhpMyAdmin таблицу "users_access_denied_table".

Предопределенные константы

CURRENT_SITE – содержит ID текущего сайта

IS_ERROR_404 – объявлена и содержит истину, если генерируется страница для 404 ошибки (страница не найдена)

CURRENT_STRUCTURE_ID – содержит ID текущего узла структуры для отображения пользователю

COUNTER_INSTALL – флаг наличия активного модуля «Статистика посещаемости сайта»

CACHE_INSTALL – флаг включенного модуля кэширования

SITE_LOCAL - Локаль

SITE_CODING - Кодировка

MAX_SIZE_LOAD_IMAGE - Максимальный размер в одном из измерений при преобразовании загруженных изображений (малое изображение)

MAX_SIZE_LOAD_IMAGE_BIG - Максимальный размер в одном из измерений при преобразовании загруженных изображений (большое изображение)

EMAIL_TO - Адрес эл. почты администратора

CHMOD - Права доступа к директориям

CHMOD_FILE - Права доступа к файлам

DATE_FORMAT - Формат вывода даты

DATE_TIME_FORMAT - Формат вывода даты и времени

Полезные константы для управления работой системы

DENY_ADD_STRICT_INTO_LOG — Запрещает добавление ошибок с уровнем E_STRICT в log-файл системы управления

ADD_COMMENT_DELAY — Время (в секундах) в течение которого пользователь не может добавлять комментарии

ALLOW_SHOW_XML — Разрешает отображение сгенерированного XML кода при поступлении методом GET запроса 'show_xml' по умолчанию — выключено

ALLOW_WYSIWYG_COMMENT_IS_ITEM — Константа, определяющая отображение WYSIWYG редактора для текста комментария элемента информационной системы. true — использовать WYSIWYG, false — выключен

ALLOW_WYSIWYG_DESCRIPTION_IS — Константа, определяющая отображение WYSIWYG редактора в описании информационной системы. true — использовать WYSIWYG, false — выключен

ALLOW_WYSIWYG_DESCRIPTION_IS_GROUP — Константа, определяющая отображение WYSIWYG редактора в описании информационной группы. true — использовать WYSIWYG, false — выключен

ALLOW_WYSIWYG_DESCRIPTION_IS_ITEM — Константа, определяющая отображение WYSIWYG редактора в описании элемента информационной системы. true — использовать WYSIWYG, false — выключен

ALLOW_WYSIWYG_TEXT_IS_ITEM — Константа, определяющая отображение WYSIWYG редактора для задания текста элемента информационной системы. true — использовать WYSIWYG, false — выключен

DEFAULT_LNG — Константа, определяющая язык раздела администрирования по умолчанию

DENY_ADD_STRICT_INTO_LOG — Запрещает добавление ошибок с уровнем E_STRICT в log-файл системы управления

DIAGRAMM_LIMIT — Ограничение вывода значений для диаграммы

DISABLE_COMPRESSION — Флаг выключает компрессию передаваемых страниц

EXTENSION_NOT_IN_BACKUP — Расширения файлов, не включаемые в архив резервной копии, указываются в одну строчку через пробел

GISTOGRAMM_LIMIT — Ограничение вывода значений для гистограммы

JPG_QUALITY — Качество (уровень компрессии с потерей качества) JPG изображений при преобразовании

MAIL_EVENTS_STATUS — Константа, определяет, какие события будут отправляться администратору по почте. Например, при значении 2 будут отправляться сообщения о событиях со статусом 2 и выше Статусы событий: 0 — Нейтральные события; 1 — Успешные события; 2 — События низкого уровня критичности; 3 — События среднего уровня критичности; 4 — События наивысшего уровня критичности.

ON_PAGE — Количество сообщений, выводимых на страницу

POLLS_WIDTH — Эталонная ширина колонки для голосований, в px

STAT_ON_PAGE — Число записей, выводимых на страницу (для системы статистики посещаемости)

STAT_PERIOD_STORAGE — Время хранения подробной статистики посещаемости сайта (в днях)

SUPERUSER_EMAIL — Адрес эл. почты главного администратора системы управления

UPLOADDIR — Директория для размещения загружаемых файлов

USER_NONE - Имя пользователя в log-ах, если пользователь не определен (например, ошибка в клиентской части)

USE_WYSIWYG — Разрешает использование визуального редактора

ALLOW_PANEL — Разрешает использование панели, при значении false панель в клиентском разделе не отображается.

DENY_INI_SET — запрещает выполнение функции ini_set() методами HostCMS. Введено в версии 3.2.4.

SET_LAST_MODIFIED_DOCUMENT — разрешает установку времени для Last-Modified в соответствии с датой текущей версии документа, отображаемого структурой. По умолчанию разрешено.

Примечание: Константа используется до версии 4.0

SET_LAST_MODIFIED_INFORMATION_SYSTEM — разрешает установку времени для Last-Modified в соответствии с датой элемента информационной системы. По умолчанию разрешено.

Примечание: Константа используется до версии 4.0

EXPIRES_TIME — время истечения страницы в секундах. Если не определена, используется значение 300.

LAST_MODIFIED_TIME — время последней модификации страницы, относительно текущего серверного времени. Если не определена, используется значение 0.

TMP_DIR — содержит относительный путь к директории для размещения временных файлов. Значение по умолчанию — «tmp/».

ALLOW_SET_LOCALE — разрешает установку локали, по умолчанию true

ALLOW_CACHE_LOG_LARGE_SIZE — разрешает модулю кэширования протоколировать в журнал событий ситуации превышения размера записываемого элемента.

TAG_TRANSLIT — автоматически транслитерировать путь для вновь создаваемых тегов.

OPTIONAL_LAST_SLASH — позволяет не указывать последний слэш в пути к разделу. По умолчанию имеет значение false.

При значении **true** эквивалентным является указание <http://www.site.ru/news> и <http://www.site.ru/news/>

При значении **false** верным является указание <http://www.site.ru/news/>

INDEX_PAGE_IS_DEFAULT — используется при размещении информационной системы на главной странице. По умолчанию имеет значение false, для корректной обработки URL информационной системой, размещенной на главной странице (с путем /) необходимо установить в true.

USE_ONLY_HTTPS_AUTHORIZATION — Устанавливает возможность авторизации в центре администрирования только по защищенному протоколу HTTPS.

DENY_LOCATION_302_LAST_SLASH — Запрещает 302-й редирект к последнему слэшу. При отсутствии константы редирект происходит. Для запрета редиректа установите значение true.

NOT_EXISTS_FILE_404_ERROR — Разрешает выдачу заголовка 404 при попытке запросить отсутствующий файл, например: */myfile.htm*. Используется для ускорения обработки подобных запросов. Для запрета выдачи заголовка 404 установите значение false.

URL_SPACE_SEPARATOR — символ для замены пробела при преобразовании путей, если не указан, используется дефис.

SHOP_GROUP_PATH_PREFIX — префикс при формировании пути для группы, если путь не задан. Если константа не задана, используется префикс «group_».

SHOP_ITEM_PATH_PREFIX — префикс при формировании пути для товаров, если путь не задан. Если константа не задана, используется префикс «item_».

Настройка файла `main_classes.php`

Константа, определяющая возможность внесения изменений на сайт из раздела администрирования.

```
define ('READ_ONLY', false);
```

Список блоков меню центра администрирования:

```
$GLOBALS['gAdminSubMenu'][0]['name'] = 'Структура сайта';
$GLOBALS['gAdminSubMenu'][0]['image'] = '/admin/images/structure.gif';

$GLOBALS['gAdminSubMenu'][1]['name'] = 'Сервисы';
$GLOBALS['gAdminSubMenu'][1]['image'] = '/admin/images/service.gif';

$GLOBALS['gAdminSubMenu'][2]['name'] = 'Пользователи';
$GLOBALS['gAdminSubMenu'][2]['image'] = '/admin/images/users.gif';

$GLOBALS['gAdminSubMenu'][3]['name'] = 'Системные функции';
$GLOBALS['gAdminSubMenu'][3]['image'] = '/admin/images/system.gif';
```