Настройка WWW-сервера и других публичных служб

Wiki

Как сделать школьный веб-сайт

Когда вы заходите по протоколу HTTP на сервер, он в ответ на ваш запрос посылает вам некий текст, обычно в формате HTML. HTML — это язык разметки для организации гипертекстовых данных. Для просмотра этого документа пользователем используется программа, называемая веб-браузером («навигатором»), которая умеет показывать HTML-страницы описанным в них образом и обрабатывать разного рода деятельность пользователя: например, вы нажимаете мышкой на ссылку, и она содержимое страницы по этой ссылке показывает. Возникает вопрос: откуда берутся эти массивы данных в виде HTML-страниц? Для ответа на этот вопрос создадим собственный веб-сайт, который будет являться источником HTML-документов.

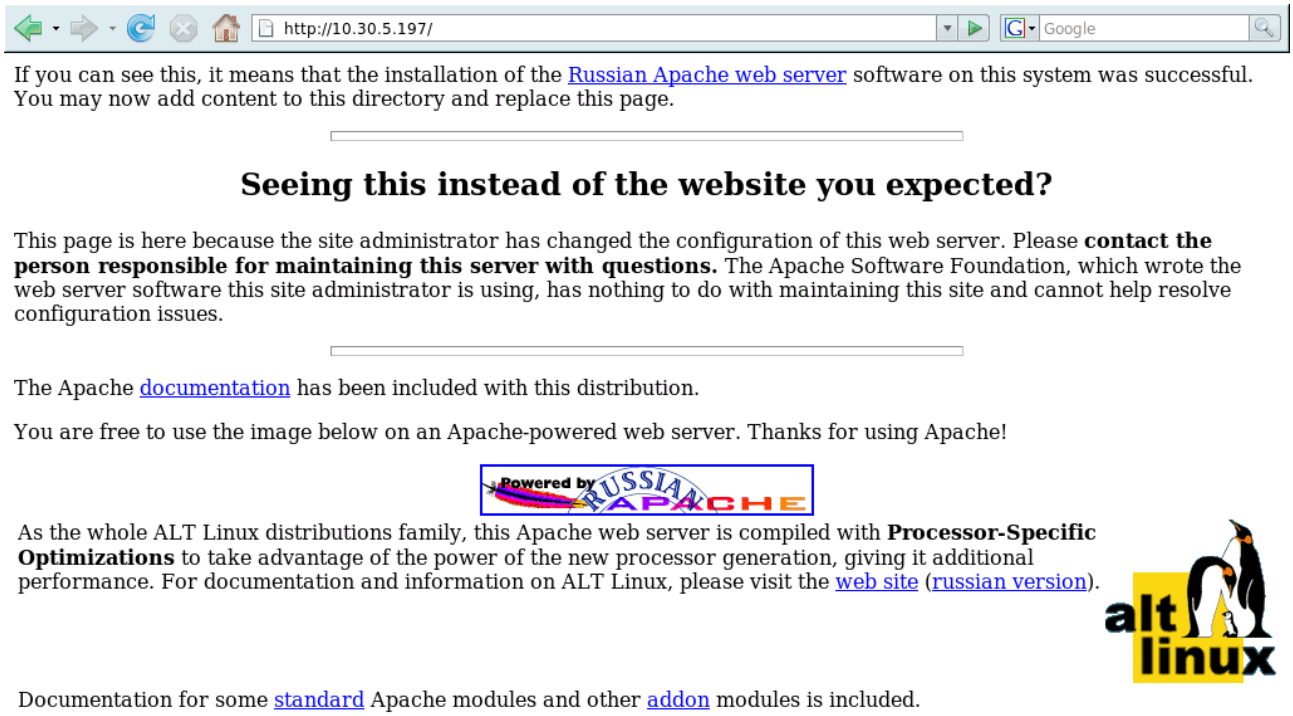
Веб-сервер Apache

Для создания веб-сайта в начале необходимо установить веб-сервер — программу, общающуюся с клиентами по протоколу HTTP. В связи с тем, что на момент начала создания ПСПО в веб-сервере Apache версии 2.0 были определенные проблемы с безопасностью при включении модуля mod_proxy, а версия 2.2 ещё не была доступна, в ПСПО используется старая, но испытанная версия Apache 1.3, хотя в большинстве дистрибутивов сейчас используется Apache 2.2.

Для развертывания веб-сервера достаточно установить пакет apache и запустить службу httpd. В случае ПСПО веб-сервер будет работоспособен сразу после установки. Установим и запустим сервер Apache следующей командой:

```
# apt-get install apache && service httpd start
```

Теперь при попытке зайти на доменное имя или IP-адрес используемой машины по протоколу HTTP (например, при помощи браузера) будет показано следующее:



По умолчанию сервер Apache использует в качестве корневого каталога сайта (DocumentRoot) каталог со своей документацией. То, что мы видим приведённую выше страницу, означает, что веб-сервер работает, однако у нас отсутствует собственно веб-сайт, поскольку нет никакого содержания.

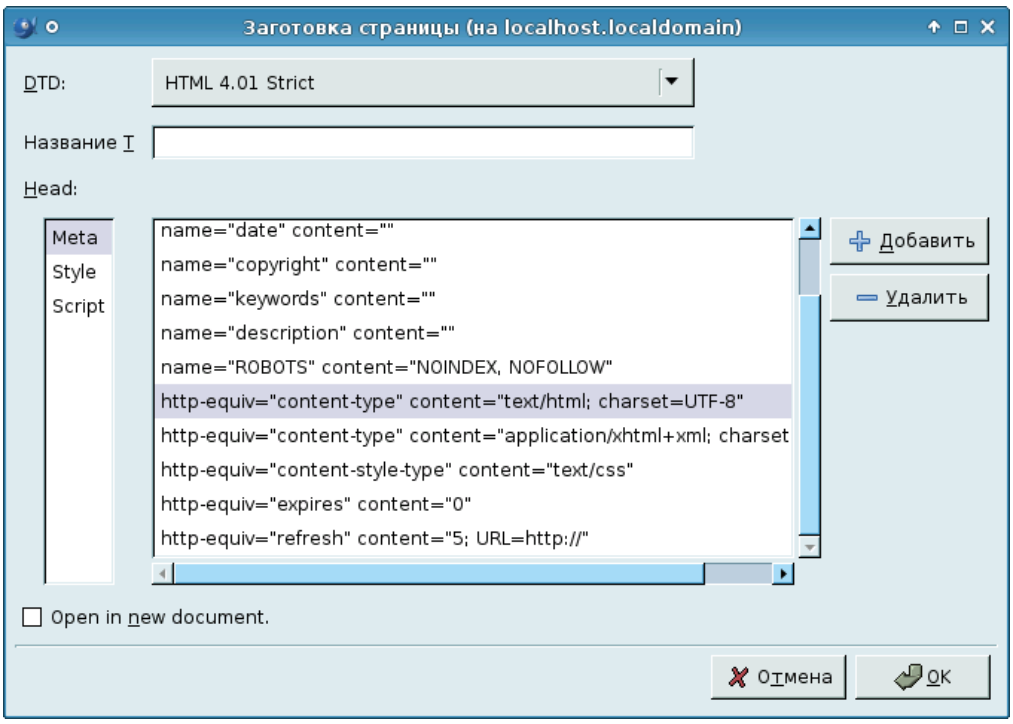
Создания HTML-страниц с помощью Bluefish

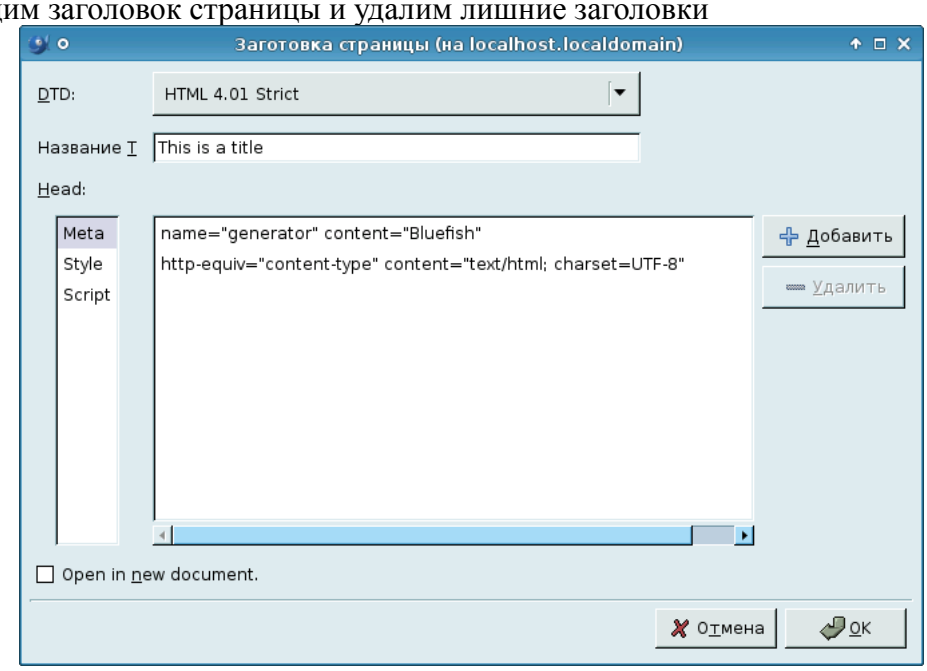
Для создания сайта можно использовать специально предназначенную среду разработки — Bluefish. Перед созданием каких-либо страниц создадим подкаталог в месте, предназначенном для размещения файлов контента веб-сервера, `/var/www`, и для простоты разрешим изменять его содержимое всем локальным пользователям:

```
# mkdir /var/www/site
# chmod o+w /var/www/site
```

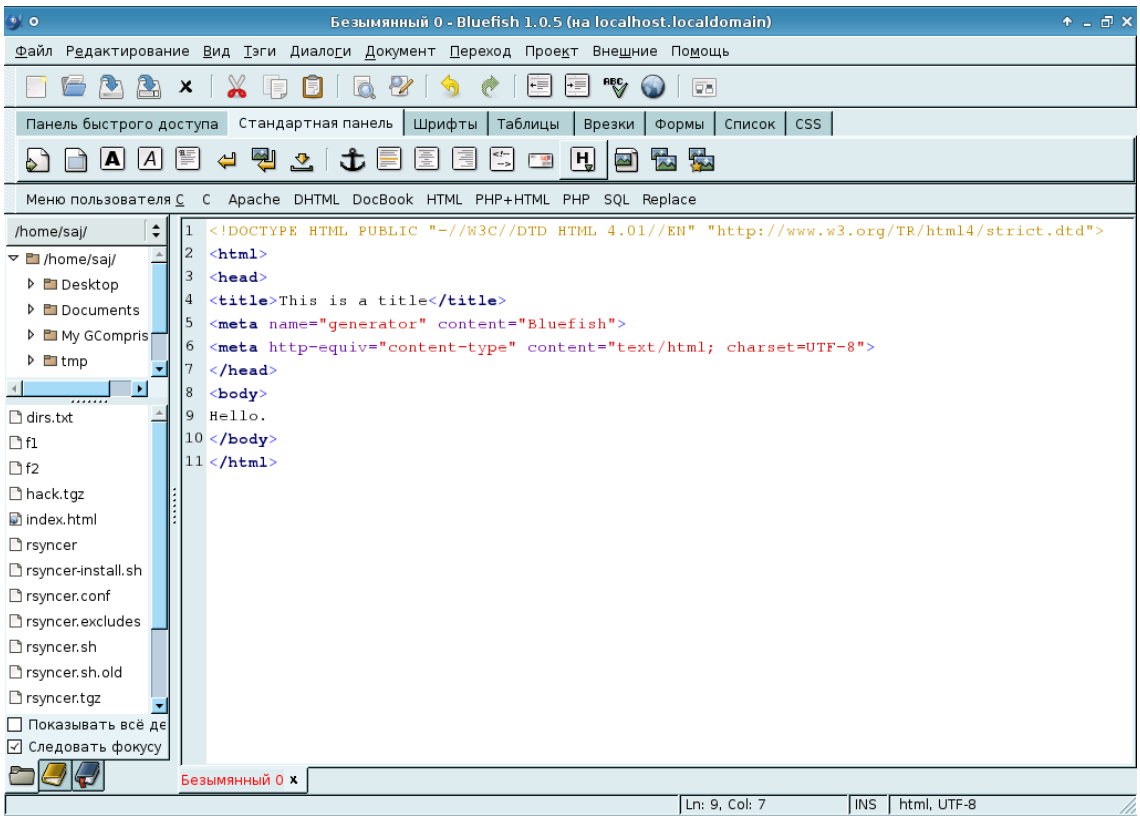
Создадим с помощью редактора Bluefish HTML-страницу:

1. Откроем bluefish
2. Откроем диалог создания новой HTML-страницы

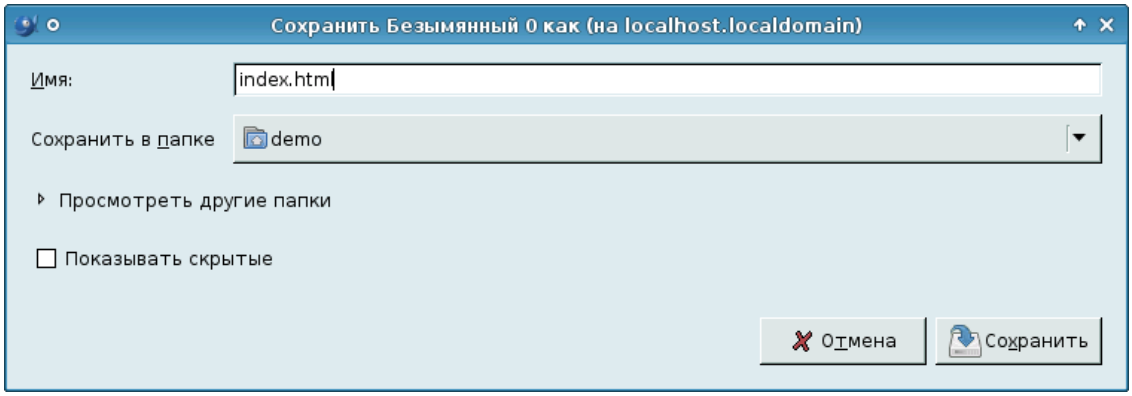
- 

3. Зададим заголовок страницы и удалим лишние заголовки
   - 

4. Сгенерируем страницу и добавим содержимое страницы
5. Добавим содержимое страницы.
   -



6. Сохраним полученный результат



Сохраним её под именем index.html в свою домашнюю директорию и скопируем её в `/var/www/site/`.

Затем отредактируем файл конфигурации `/etc/httpd/conf/httpd.conf`. Поменяем в нём поле DocumentRoot следующим образом:

```
# DocumentRoot: The directory out of which you will serve your /Root
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
#
DocumentRoot "/var/www/site"
```

Теперь, после перезапуска apache командой `service httpd restart`, в браузере при обращении к данной машине можно будет увидеть только что созданную страницу.

Подходы к автоматизированному созданию веб-сайтов

Продемонстрированный подход к созданию HTML-страниц вручную, хоть и полезен для

знакомства с языками разметки, но является устаревшим. Практика создания HTML-страниц вручную на данный момент по большей части ушла в прошлое. Дело в том, что при таком подходе требуется очень много усилий для поддержания сайта в актуальном состоянии. Ранее существовала даже специальная профессия — «контенщик», человек, которому платили за регулярные обновления сайта (она существует и сейчас, но имеет несколько иную специфику). Задача создания осмысленной структуры сайта так же не слишком проста. В дополнение ко всему, вручную очень легко случайно или по незнанию написать HTML-код, который будет отображаться некоторыми браузерами некорректно.

В какой-то момент произошло осознание того, что дизайн сайта и его наполнение информацией, и представление его виде HTML — это различные задачи, которые стоит решать по отдельности. Появилось два подхода к решению задачи наполнения сайта информацией:

- Сайт может представлять собой ответственную структуру. Информация, в таком случае, попадает на сайт после прохождения непростого жизненного цикла: какой-нибудь отдел пишет запрос, запрос обрабатывается службой контента, которая обращается к авторам; авторы пишут текст, текст вычитывается редактором, шеф утверждает текст, и лишь затем текст попадает в нужное место на сайте. Иногда этапов намного меньше, их может быть всего два — один человек готовит информацию, другой размещает её. В любом случае, до появления на сайте информация должна пройти сколько-нибудь этапов обработки.
- Может существовать сообщество людей (к примеру — учителя, ALT Linux team, любители пива). Члены сообщества могут не быть ни веб-дизайнерами, ни веб-программистами, однако им может быть надо оперативно пополнять информацию на сайте наиболее простым способом. В этом случае, когда речь идет о совместном использовании и редактировании контента, важнее всего простота работы.

Для представления информации в виде веб-страниц используются веб-движки. То есть, помимо веб-сервера, на сервере имеется еще одна программа, которая при запросе от клиента агрегирует хранящуюся на сайте информацию, представленную в специальном виде, и формирует из неё HTML-страницы. Таких движков достаточно много, и среди них тоже можно выделить две большие группы:

- CMS (Content Management System, системы управления контентом). Это инструменты людей, занимающихся сайтостроением. В них есть уже готовые большое количество готовых модулей, например, системы поддержки блогов, документооборота, внедрения оригинального дизайна.
    - Одним из наиболее развитых инструментов такого плана является движок (точнее, application server) Zope, написанный на Python. Под него существуют специальные программы, например, реализации системы управления содержимым сайта.
    - Сайт ALTLinux использует другой аналогичный движок — Joomla.
    - Сайт fosscenter.ru сделан на движке Drupal. Drupal интересен легкостью установки и удобством управления, однако написан на PHP и в некоторой степени функционально и архитектурно перегружен.
    - Существует и движок, ориентированный на сайты поддержки учебного процесса — Moodle. Сайты, сделанные на нем, выглядят как система управления учебными курсами.
    - В любом случае, портальный движок обязательно должен сопровождаться ответственным системным администратором, который в этом движке разбирается. Кроме того, существует множество коммерческих портальных движков.

- Вики (Wiki). Это технология, упрощающая групповую работу с информационным наполнением сайта. На гавайском языке «wiki» означает «быстро». Основная концепция этой технологии — создание веб-страницы должно занимать времени не больше, чем просто набор текста. Кроме того, должна быть максимально упрощена процедура отмены изменений. В отличие от портальных решений, значительная часть которых не бесплатна, существуют десятки свободных wiki-движков.

Использование Wiki

В качестве движка Wiki рассмотрим движок MoinMoin. Он написан на Python и является разумным компромиссом между готовой работающей программой и гибко настраиваемой системой, которую так же и удобно перепрограммировать и расширять. На самих дисках с ПСПО moin отсутствует, однако установить его достаточно просто, если подключить школьный репозиторий и выполнить следующую команду.

```
# apt-get update && apt-get install moin
```

Следующим шагом будет создание экземпляра wiki-сайта, для чего существует специальный сценарий `moin-instance-setup`. Его следует запустить с именем вики-узла (последнее должно включать только английские буквы, цифры и дефис):

```
# moin-instance-setup school
Checking configuration sanity for httpd: DONE
Stopping libhttpd.ep service: DONE
Starting libhttpd.ep service: DONE
Moin-Moin school installation is finished:
        Wiki pages: /var/www/wiki/school
        Wiki url: http://localhost.localdomain/school
           Additional Apache config file: /etc/httpd/conf/addon-modules.d/moin-
school.conf
Edit /var/www/wiki/school/cgi-bin/wikiconfig.py to set your site up.
```

При запуске показываются сведения о том, где находятся файлы настроек и содержимого и какой используется веб-адрес.

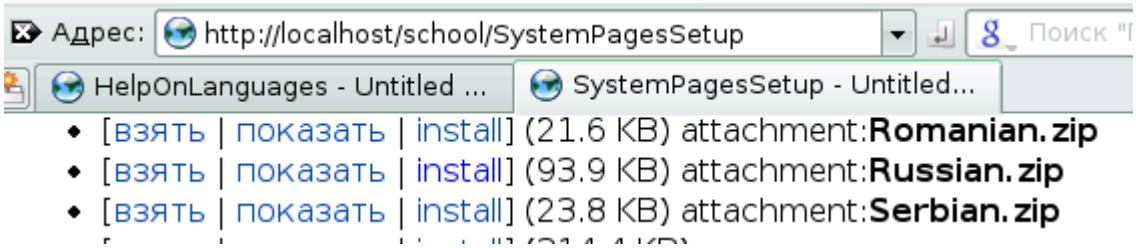
После этого следует зайти с помощью браузера на адрес `http://localhost/school` и завести там учётную запись, например с именем Admin (для этого так же можно перейти по адресу `http://localhost/school/UserPreferences`).

Для задания суперпользователя Wiki нужно изменить файл `/var/www/wiki/school/cgi-bin/wikiconfig.py`, где необходимо задать переменной `superuser` в качестве значения список имен супер-пользователей. Например, следующая команда дает права супер-пользователя пользователю с именем `Admin`:

```
# sed 's/#superuser = \[u"YourName", \]/superuser = [u"Admin", ]/' -i
/var/www/wiki/school/cgi-bin/wikiconfig.py
```

(данная команда заменяет закомментированную по умолчанию строку `#superuser = \ [u"YourName", \]` на строку `superuser = [u"Admin", ]`)

После того, как в MoinMoin появилась учетная запись суперпользователя, можно установить необходимые пакеты локализации и сменить интерфейс системы на русский. Это можно сделать на странице `http://localhost/school/SystemPagesSetup`:



Для запрещения создания новых пользователей в MoinMoin, входящем в состав ПСПО, проще всего создать необходимых пользователей, а затем запретить изменения в каталоге, где MoinMoin хранит информацию о пользователях:

```
# chmod u-w /var/www/wiki/school/data/user
```

(Подробнее про это можно прочитать на сайте MoinMoin)

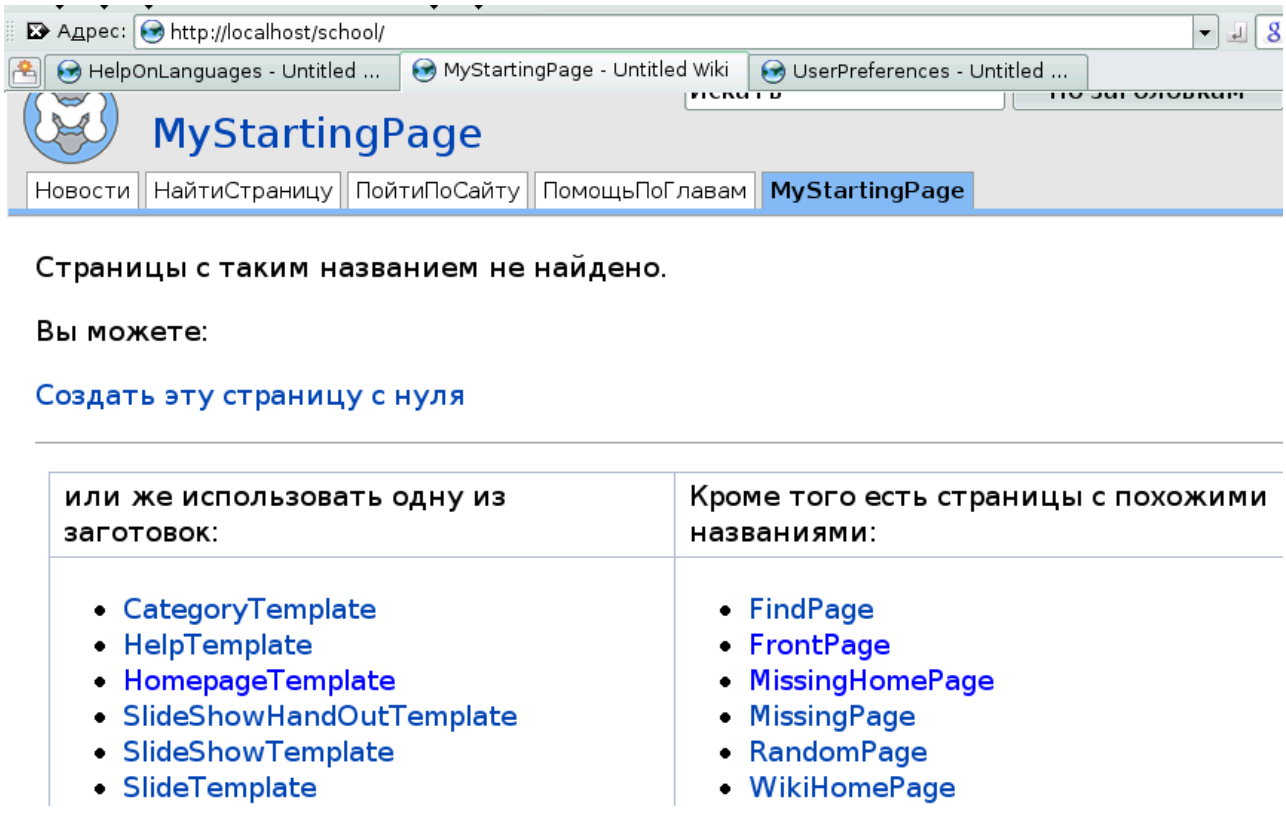
Если этого не сделать, каждый желающий может зарегистрировать себе пользователя, что может привезти к нежелательным последствиям. Перед добавлением нового пользователя придётся вернуть разрешение на запись:

```
# chmod u+w /var/www/wiki/school/data/user
```

После заведения пользователей следует установить стартовую страницу wiki-сайта. Для одноязычной вики следует раскомментировать (и, при необходимости, отредактировать) следующую строчку в файле `/var/www/wiki/school/cgi-bin/wikiconfig.py`:

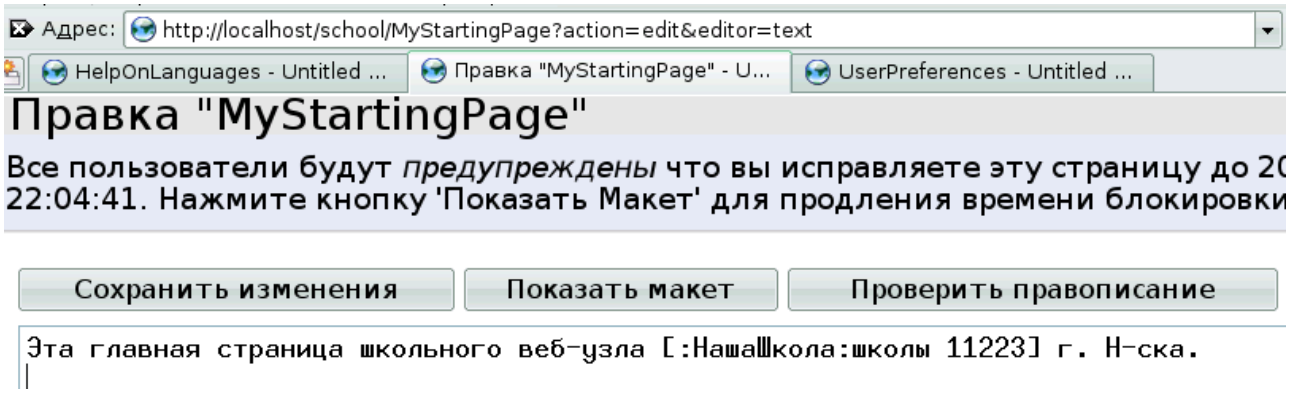
```
page_front_page = u'MyStartingPage';
```

Теперь можно приступать к редактированию содержимого wiki-сайта. При переходе на адрес `http://localhost/school` появляется сообщение об отсутствующей странице. Это ситуация достаточно типична для содержимого wiki-сайтов: ссылка на страницу есть, а страницы ещё нет.

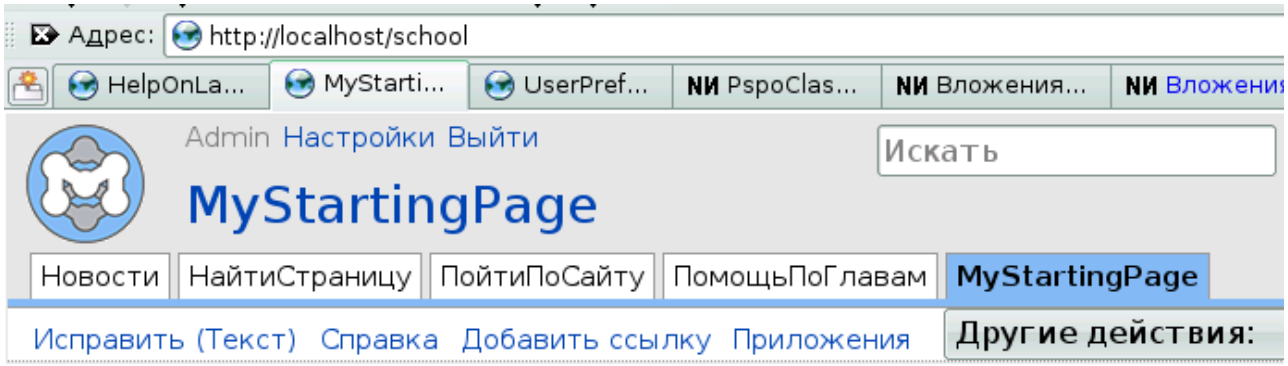


Для создания страницы выберем вариант «Создать страницу с нуля» и введем некоторый

текст, содержащий ссылку на другую страницу. Для идентификации страниц wiki используется CamelCase.



Теперь у wiki-сайта есть главная страница, содержащая ссылку на (ещё не созданную) страницу НашаШкола. Нажав на эту ссылку, можно создать страниу с информацией о школе — и так далее.



MoinMoin позволяет создавать страницы с ссылками, таблицами, рисунками, поддерживает персональные страницы пользователей и страницы категорий, а также поддерживает (как и все wiki) историю изменения страницы. Внешний вид MoinMoin при необходимости полностью настраивается.