Приемы конфигурирования Web-сервера Apache 2

Web-сервер Apache – это мощный и многофункциональный программный продукт с разнообразными возможностями. Здесь будут рассмотрены приемы конфигурирования Apache, наиболее часто встречающиеся при разработке Web-сайтов.
Как известно, все настройки сервера Apache находятся в файле httpd.conf, доступ к которому имеется не всегда. Например, если используется виртуальный сервер на хостинге, когда один сервер Apache обслуживает сотни сайтов, то, естественно, нельзя позволить владельцу одного сайта менять конфигурацию сервера, которая отразится на всех остальных сайтах. Тем не менее Web-сервер Apache допускает конфигурирование на уровне отдельных каталогов при помощи файлов .htaccess. Именно на работу с этими файлами, как единственными конфигурационными файлами, которые доступны большинству Web-разработчиков, и будет здесь сделан основной упор.

Особенности конфигурирования в Windows

Web-сервер Apache в настоящее время является, действительно, многоплат-формной системой, работающей на различных клонах UNIX, в Windows и других операционных системах. Но т. к. корни Apache идут из UNIX-систем, при конфигурировании Apache под Windows следует учитывать некоторые особенности, не очевидные для пользователей этой системы.
Во-первых, в системах UNIX и Windows используются различные разделители каталогов и файлов. В UNIX для разделения каталогов применяется прямой слеш (/), например, /pub/server/bin, в то время как в Windows традиционно используется обратный слеш (\), например, c:\apache\bin\, хотя данная операционная система поддерживает и прямой слеш. При конфигурировании Apache следует придерживаться UNIX-нотации, т. е. в качестве разделителей каталогов указывать прямой слеш, например, c:/apache/bin/. Еще одной особенностью файловой системы Windows является наличие пробелов в именах файлов и каталогов. Для использования таких путей в конфигурационных файлах их следует обрамлять двойными кавычками (").

Файл .htaccess

Файл .htaccess (с точкой в начале имени) – это конфигурационный файл, который дает возможность настраивать работу сервера на уровне отдельных каталогов: устанавливать права доступа к файлам в каталогах, менять названия индексных файлов, самостоятельно обрабатывать коды ответов протокола HTTP, модифицировать адреса запрошенных страниц.

> **Замечание**
> В UNIX точка в начале файла является признаком скрытого файла. Поэтому большинство конфигурационных файлов предваряются точкой. Именно отсюда берет начало обозначение текущего каталога точкой ("."), а родительского каталога двумя точками ("..").

Файл .htaccess может быть размещен в любом каталоге. Директивы этого файла действуют на все файлы в текущем каталоге и во всех его подкаталогах (если эти директивы не переопределены директивами файлов .htaccess во вложенных каталогах).

Изменения, вносимые в файлы .htaccess, вступают в силу немедленно и не требуют перезагрузки сервера в отличие от изменений, вносимых в главный конфигурационный файл httpd.conf.
Для того чтобы файлы .htaccess можно было использовать, необходимы соответствующие настройки главного конфигурационного файла httpd.conf, где должны быть прописаны директивы, которые разрешат файлу .htaccess переопределять конфигурацию Web-сервера в каталоге. Список этих директив задается директивой AllowOverride.
Директива AllowOverride может включать в себя одну из следующих директив или их комбинацию:

- AuthConfig – разрешает использование директив аутентификации и управления доступом (таких как AuthDBMGroupFile, AuthDBMUserFile, AuthType, AuthName, AuthUserFile, AuthGroupFile, Require);
- FileInfo – разрешает использование директив, управляющих типами документов (AddEncoding, AddLanguage, AddType, DefaultType, ErrorDocument, LanguagePriority);
- Indexes – разрешает использование директив, управляющими индексами каталогов (таких как AddDescription, AddIcon, AddIconByEncoding, AddIconByType, DefaultIcon, DirectoryIndex, FancyIndexing, HeaderName, IndexIgnore, IndexOptions, ReadmeName);
- Limit – разрешает применение директив, управляющих доступом к хостам (Allow, Deny, Order);
- Options – разрешает использование директив, управляющих специфическими свойствами каталогов (Options, XBitHack).

Синтаксис директивы следующий:

AllowOverride *опция_1, опция_2, опция_3* . . .

Для того чтобы дать директивам файлов .htaccess максимальные права на изменения директив, значение директивы AllowOverride в файле httpd.conf должно быть равно All. Оно является значением по умолчанию.

AllowOverride All

Запретить переопределение любых директив в конфигурационных файлах .htaccess можно при помощи значения None:

AllowOverride None

При установке значения директивы AllowOverride в None сервер не читает файлы .htaccess и соответственно управление сервером с помощью этих файлов невозможно. Установка значения None может ускорить ответ сервера.

> **Замечание**
> Название конфигурационного файла можно изменить, и например, назвать его не .htaccess, а access.conf. За название этого файла отвечает директива AccessFileName в файле httpd.conf. Изменение названия конфигурационного файла .htaccess не рекомендуется, т. к. это может усложнить дальнейшую поддержку сервера.

Синтаксис файла .htaccess

Перед тем, как будут рассмотрены примеры, остановимся на синтаксисе директив в файлах .htaccess. Пути к файлам и каталогам должны указываться от корня сервера, например, /pub/home/server1/html/. Если абсолютный путь от корня сервера не известен, то его можно узнать, спросив у администратора сервера, либо посмотрев самостоятельно, запустив на сайте функцию PHP phpinfo(). Данная функция выведет на экран конфигурацию PHP – значение переменной doc_root будет содержать путь от корня сервера до корневого каталога виртуального хоста. Иногда эта переменная не инициализирована, поэтому следует проверить значения переменных: open_basedir, DOCUMENT_ROOT и SCRIPT_FILENAME.

При указании абсолютных URL обязательно должны быть заданы протоколы, например:

Redirect / http://www.newsite.ru

В файлах .htaccess недопустимы пробелы в указаниях путей к файлам и в названиях самих файлов, т. к. это приводит к генерации кода ответа 500 – ошибка конфигурации сервера: "Internal Server Error". В листинге 2.1 можно видеть пример такого ошибочного файла.

Листинг 1. *Ошибка в файле .htaccess*

```
AuthType Basic
AuthName admin
AuthUserFile c:/web sites/.htpasswd
<Limit GET POST>
   require user admin
</Limit>
```

Ошибки с указанием пути с пробелами характерны для Windows. Если в имени файла или каталога есть пробелы, то заключите соответствующий параметр в кавычки, как это сделано в листинге 2.2.

Листинг 2. *Правильный файл .htaccess*

```
AuthType Basic
AuthName admin
AuthUserFile "c:/web sites/.htpasswd"
<Limit GET POST>
   require user admin
</Limit>
```

Далее будут рассмотрены часто встречающиеся задачи конфигурирования сайтов и их решения с помощью файлов .htaccess.

Индексные страницы

Обычно названия индексных страниц по умолчанию определены в директиве DirectoryIndex конфигурационного файла httpd.conf, например:

DirectoryIndex index.html index.shtml index.htm

Могут возникнуть ситуации, когда необходимо изменить состав индексных файлов, например, если нужна индексная страница index.php, а в основном конфигурационном файле httpd.conf она не прописана. Эту задачу можно решить при помощи файла .htaccess, в котором необходимо создать директиву DirectoryIndex, где будут перечислены имена индексных страниц (листинг 3).

Листинг 3. *Определение индексных страниц*
```
DirectoryIndex index.php index.html index.shtml index.htm
```

При запросе каталога без указания имени файла сначала будет осуществлен поиск страницы с именем index.php. Если страницы с таким именем нет в каталоге, то аналогичные операции будут произведены с файлом index.html и т. д. до конца списка, пока не будет найдена и открыта соответствующая страница.

Запрет на отображение содержимого каталога при отсутствии индексного файла

Часто требуется запретить отображение списка файлов в каталоге, если не указан или отсутствует индексный файл (листинг 4). Например, запретить отображение содержимого каталога с изображениями. Если такой запрет не поставить, то пользователь, обратившийся напрямую к такому каталогу, получит список всех изображений.

Листинг 4. *Запрет на отображение содержимого каталога*
```
Options -Indexes
```

Обработка кодов ответов Web-сервера Apache

Ни один сайт не застрахован от возникновения ошибок. Самой частой ошибкой является переход по ссылке на несуществующую страницу. В этом случае Apache генерирует код ответа 404 и отображает автоматически сгенерированную страницу с сообщением об ошибке. Наличие несуществующих страниц производит плохое впечатление на посетителей сайта. Это впечатление можно сгладить, если вместо стандартных страниц, приевшихся посетителю, подставлять собственные страницы с сообщением об ошибке, на которых будут принесены извинения и предоставлено меню для того, чтобы посетитель мог продолжить работу с сайтом. За назначение страниц-обработчиков кодов ответа протокола HTTP несет ответственность директива ErrorDocument (листинг 5).

Листинг 5. *Обработка ошибок Apache*

```
ErrorDocument 401 /401.php
ErrorDocument 403 /403.php
ErrorDocument 404 /index.php
ErrorDocument 500 /500.php
```

После директивы ErrorDocument следует указать код ответа и страницу, на которую необходимо перенаправить посетителя при возникновении данного кода ответа. Страницы-обработчики можно использовать не только для отображения или маскировки ошибок, но также для их подсчета и сохранения в log-файлах, анализ которых может предотвратить взлом системы администрирования сайта или указать на ошибку в проектировании системы навигации по сайту. В этом случае в код страниц нужно вставить соответствующие скрипты.

В табл. 2.1 приведена расшифровка возможных кодов ответов.

Таблица 2.1. *Возможные коды ответов*

Ответ	**Описание**
"400 Bad Request"	В запросе клиента сервер нашел синтаксическую ошибку
"401 Unautorized"	Запрос требует аутентификации пользователя
"403 Forbidden"	Доступ к запрашиваемому ресурсу запрещен. Клиент не должен повторять запрос
"404 Not Found"	Запрашиваемый документ на сервере отсутствует
"405 Method Not Allowes"	Метод запроса, используемый клиентом, неприемлем
"406 Not Acceptable"	Запрашиваемый ресурс недоступен в том формате, который может принимать клиент
"407 Proxy Authentification Required"	Несанкционированный запрос доступа к proxy-серверу. Сервер отправляет заголовок Proxy-Authentificate со схемой аутентификации и областью запрашиваемого ресурса
"408 Request Time-Out"	Клиент не завершил свой запрос за время ожидания запроса, заданное серверу
"409 Conflict"	Возник конфликт запроса клиента с другим запросом
"410 Gone"	Запрашиваемый ресурс удален с сервера
"411 Length Required"	В своем запросе клиент должен предоставить заголовок Content-Length, в котором указан размер запроса
"413 Request Entity Too	Сервер отказывается обрабатывать запрос:

Large"	слишком большое тело сообщения. Сервер может прервать соединение, чтобы клиент не продолжал отправлять этот запрос
"414 Request-URI Too Long"	Сервер отказывает обрабатывать запрос: слишком большой размер заданного URI
"415 Unsupported Media Type"	Сервер отказывается обрабатывать запрос: отсутствует поддержка формата тела сообщения
"500 Internal Server Error"	Ошибка конфигурации сервера или внешней программы
"501 Not Implemented"	Сервер не поддерживает требуемые функции для выполнения запроса
"502 Bad Gateway"	Неверный ответ вышестоящего сервера или proxy-сервера
"503 Service Unavailabel"	Служба временно недоступна
"505 HTTP Version Not Supported"	Версия HTTP, используемая клиентом, не поддерживается

Как выполнять код PHP в файлах HTML?

Обычно PHP-код выполняется в файлах с расширением php. Иногда возникают ситуации, когда необходимо выполнять PHP-код в файлах с другим расширением. Наиболее часто такие задачи встречаются при обновлении сайта, когда в статические страницы нужно внести динамические вставки на PHP, а переименовывать все страницы сайта (изменять расширение) не хочется, т. к. это влечет серьезную работу по изменению всех ссылок в HTML-страницах. В этом случае можно дать указание Web-серверу выполнять PHP-код не только в файлах с расширением php, но и в файлах с расширением html (листинг 6).

Листинг 6. *Выполнять код PHP в файлах HTML*

```
RemoveHandler .html .htm
AddType application/x-httpd-php .php .htm .html .phtml
```

Первая строка в листинге 2.6 удаляет обработчик файлов с расширениями html и htm, а вторая строка сообщает серверу о необходимости использовать для файлов с расширениями htm и html обработчик PHP.

Замечание

Следует отметить, что введение дополнительных расширений увеличивает нагрузку на Web-сервер.

Задание кодировки файлов на сервере

Для того чтобы указать серверу, с применением какой кодировки созданы файлы в каталоге, следует использовать директиву AddDefaultCharset (листинг 7). Указанная кодировка отправляется браузеру в заголовке Content-Type и позволит браузеру клиента автоматически переключиться на требуемую кодировку.

Листинг 7. *Задание кодировки файлов*
```
AddDefaultCharset Windows-1251
```

По умолчанию в Apache значение этой директивы установлено в ISO-8859-1, что исключает поддержку кириллицы. Далее приведены имена нескольких часто применяемых кодировок:

- ISO-8859-1 – западноевропейская;
- ISO-8859-15 – западноевропейская с поддержкой символа Евро;
- Windows-1251 – кириллица Windows;
- KOI8-R – кириллица UNIX;
- utf-8 – 8-битовая кодировка Unicode;
- utf-7 – 7-битовая кодировка Unicode.

Задание кодировки загружаемых файлов

При загрузке файлов на сервер можно указать, в какой кодировке сервер должен ожидать файл. Для этого служит директива CharsetSourceEnc (листинг 8).

Листинг 8. *Задание кодировки загружаемых файлов*
```
CharsetSourceEnc windows-1251
```

Иногда возникают ситуации, когда Apache некорректно перекодирует загружаемые на сервер двоичные файлы. В результате файлы оказываются *"битыми"*. Это проблема актуальна при использовании *"русского Apache"*. Для того чтобы исправить такое поведение Apache, следует отменить перекодировку (листинги 2.9 и 2.10).

Листинг 9. *Отмена перекодировки*
```
CharsetDisable On
```

Листинг 10. *Отмена перекодировки конкретного файла*

```
<FILES filename.php>
   CharsetDisable On
</FILES>
```

Отключение директивы *MultiViews*

Включенная на хостинге опция MultiViews может вызвать неожиданные проблемы, например, отображение несуществующих страниц сайта. Скажем, на сайте существует страница с адресом **http://www.server.ru/downloads.php,** и если посетители обратятся к несуществующему каталогу **http:// www.server.ru/downloads/,** то включенная опция MultiViews вместо этого каталога подставит файл downloads.php. Однако подстановка будет выполнена не полностью– пути к изображениям, таблицам стилей и т. п. будут подставлены неверно. То есть страница будет отображена с искажениями. Это может испортить репутацию сайта, особенно если URL такого вида попадут в каталоги поисковых систем, т. к. посетители не осведомлены, что это проделки Web-сервера, а не халатность Web-разработчиков. Для подавления такого поведения Apache опцию MultiViews следует отключить (листинг 11).

Листинг 11. *Отключение директивы MultiViews*
```
Options -MultiViews
```

Запрет доступа к файлам

Для того чтобы посетители не могли получить доступ к служебным файлам из окна браузера, следует запретить доступ к таким файлам. В листингах 2.12–2.17 приведены примеры использования директив запрета Deny и разрешения доступа Allow.

> **Примечание**
> Использование директив Deny и Allow управляет только доступом к файлам из браузера, либо из другой программы-клиента. Подобные запреты не распространяются на скрипты сервера.

Листинг 12. *Запрет доступа к файлам из браузера*
```
Deny from all
```

При использовании такой директивы будет запрещен доступ из браузера ко всем файлам и каталогам текущего каталога.

Листинг 13. *Запрет доступа к определенному файлу*
```
<Files config.php>
   Deny from all
</Files>
```

Здесь запрещен доступ только к файлам с именем config.php.

Листинг 14. *Запрет доступа к файлам расширения inc*
```
<Files "*.inc">
Deny from all
</Files>
```

Здесь запрещен доступ к файлам с расширением inc. Директива <Files>, при указании имени файлов, позволяет использовать подстановочные символы:

? – любой одиночный символ;
. – любая последовательность символов, исключая символ слеша (/).

Директива <Files>, по умолчанию, не работает с регулярными выражениями, но их можно включить, поставив символ тильды (~) в опциях директивы. Синтаксис директивы в этом случае следующий:

[~] [пробел] *[регулярное_выражение]*

Листинг 15. *Запретить доступ к файлам с несколькими типами расширений*

```
<Files ~ "\.(inc|conf|cfg)$">
   Deny from all
</Files>
```

Запрещен доступ к файлам с расширением inc, conf и cfg.

Для выбора файлов также может применяться директива <FilesMatch>. Она выполняет действия, аналогичные действиям директивы <Files>, но вместо имени файла в качестве параметра принимает регулярное выражение по поиску файлов. Для того чтобы решить задачу из листинга 2.15 с помощью директивы <FilesMatch>, следует модифицировать код следующим образом:

```
<FilesMatch "\.(inc|conf|cfg)$">
   Deny from all
</Files>
```

Листинг 16. *Запретить доступ с определенного IP-адреса*

```
Deny from 195.135.232.70
```

Листинг 17. ***Разрешить доступ только с определенного IP-адреса***

```
Order deny,allow
Deny from all
Allow from 195.135.232.70
```

Директива order позволяет задать порядок, в котором будут выполняться директивы. Сначала выполняется директива запрета доступа (директива Deny), а затем разрешается доступ только для IP-адреса 195.135.232.70 (директива Allow). Если в первой строке поменять порядок следования директив на order allow,deny, то доступ для IP-адреса 195.135.232.70 не будет открыт, т. к. директива Deny, выполняемая последней, перекроет действие директивы Allow.

Примечание

Следует отметить, что разрешение доступа только с определенного IP-адреса иногда может не срабатывать. Например, в том случае, если на хостинге установлен обратный кэширующий

proxy-сервер. Если директивы разрешения доступа не работают, то вам нужно проконсультироваться по этому вопросу со службой поддержки хостинга.

Перенаправление на другой адрес

Часто встречаются задачи, когда все запросы к определенному каталогу или странице нужно перенаправить (redirect) на другой адрес. Например, при реорганизации сайта, когда скрипты переносятся из одного каталога в другой или сайт меняет доменное имя. Для того чтобы посетители, пришедшие по ссылкам из поисковых систем, по ссылкам с других ресурсов или из закладок своих браузеров, не испытывали неудобств, следует организовать переадресацию всех несуществующих URL на новые адреса.

Это можно сделать с помощью директив Redirect и RedirectMatch. Они сообщают, что ресурс по запрошенному URL отсутствует, и указывают адрес, по которому следует перейти. Директивы Redirect посылают браузеру соответствующий заголовок, и уже браузер осуществляет перенаправление (листинги 2.18–2.20).

Листинг 18. *Глобальное перенаправление на новый адрес*

```
Redirect / http://www.raysite.ru/
```

Если в условиях поиска указать /, то перенаправление на новый адрес http://www.mysite/ будет осуществляться при обращении к корню сайта: http://www.server.ru/.

Листинг 19. *Перенаправление при обращении к определенному файлу*

```
Redirect /about/index.php http://www.mysite.ru/company/
```

Листинг 20. *Перенаправление при обращении к каталогу*

```
Redirect /about http://www.mysite.ru/company/
```

Рассмотрим, как будет работать перенаправление, если запросить страницу http://www.server.ru/about/page1.php. При обращении в каталог about будет сделана попытка перенаправить запрос по адресу **http://www.mysite.ru/company/page1.php.** То есть имя запрашиваемой страницы присоединится к новому URL, хотя оно не было указано в условиях перенаправления. Если по новому адресу такой страницы нет, то будет выдана стандартная страница с 404-й ошибкой о недоступности страницы. Директива Redirect является регистрозависимой. То есть если в запросе написать имя каталога About с большой буквы (http://www.server.ru/About/), то перенаправление, указанное в листинге 2.20, не сработает, т. к. в условиях поиска каталог about написан с маленькой буквы.

Директива RedirectMatch расширяет функциональность директивы Redirect. С ее помощью можно применять регулярные выражения при указании URL, используемых при перенаправлении. Например, нужно сделать перенаправление при запросе любых страниц из каталога about и в том числе при обращении к каталогу без указания страницы (листинги 2.21 и 2.22).

Листинг 21. *Перенаправление при обращении к любым страницам каталога*
```
RedirectMatch /about/.* http://www.mysite.ru/company/
```

Листинг 22. *Перенаправление при обращении к любым страницам сайта*
```
RedirectMatch /.* http://www.mysite.ru/
```

У директив Redirect и RedirectMatch имеется дополнительный параметр, с помощью которого можно управлять состоянием перенаправления. Данный параметр может принимать следующие значения:

- permanent – ресурс перемещен навсегда (листинг 23); код состояния 301;
- temp – ресурс перемещен временно; код состояния 302;
- seeother – ресурс был заменен другим ресурсом; код состояния 303;
- gone – ресурс удален навсегда; код состояния 410. Параметр URL директивы Redirect должен быть опущен.

В зависимости от передаваемого кода состояния, браузеры могут предпринять различные действия, например, при коде состояния 303 интеллектуальные браузеры могут заменить адреса ссылок в закладках.

Листинг 23. *Ресурс перемещен навсегда*
```
RedirectMatch permanent /.* http://www.mysite.ru/
```

Преобразование адресов

В предыдущем разделе были рассмотрены директивы Redirect, с помощью которых можно сделать подмену одних запросов другими. Более широкие возможности по преобразованию URL обеспечивают директивы модуля mod_rewrite, который должен быть доступен на сервере.
Чтобы проверить наличие этого модуля, выполните функцию PHP phpinfo() и проверьте в таблице apache значение строки с именем Loaded Modules, в которой описаны загруженные модули Apache.

> **Примечание**
> Использование директив модуля mod_rewrite увеличивает нагрузку на сервер, поэтому без крайней необходимости не следует их применять.

Для того чтобы правила преобразования работали в опциях вашего сайта (виртуальный хост сервера), в файле httpd.conf должна быть включена директива:

```
Options FollowSymlinks
```

Чтобы включить механизм преобразования адресов, установите директиву RewriteEngine в значение On:

```
RewriteEngine On
```

Установка этой директивы в значение Off отключит механизм преобразования. Таким образом, можно быстро включать и отключать правила преобразования, описанные в файле .htaccess.
Директива RewriteRule определяет правила преобразования. Она имеет следующий синтаксис:

RewriteRule *шаблон_поиска строка_для_замены [флаги]*

В шаблоне поиска записывается регулярное выражение, применяемое к запрашиваемому URL. В строке для замены могут быть использованы: текст, ссылки на подстроки из регулярного выражения шаблона поиска, значения серверных переменных.
Перечислим некоторые серверные переменные, которые могут быть использованы для преобразования адресов:

- server_name – имя Web-сервера, например: www.domain.ru;
- server_port – номер порта Web-сервера;
- document_root– каталог документов верхнего уровня для Web-сайта, например: /usr/host/mysite/html;
- http_forwarded – переадресованная ссылка;
- http_host – имя компьютера Web-сервера;
- http_referer– адрес страницы, с которой был осуществлен переход на текущую страницу;
- http_user_agent – информация о Web-клиенте, который запросил текущую страницу;
- remote_addr – IP-адрес посетителя;
- remote_host – имя компьютера посетителя;
- request_method– метод запроса, который был использован при обращении к текущей странице;
- script_filename– физический путь к запрошенной странице, например: /usr/host/mysite/html/page.php;
- path_info – путь к запрошенной странице от document_root;
- query_string – параметры запроса к странице, например: id=3&parent=4;
- auth_type – тип используемой аутентификации;
- request_uri – запрошенный URL. Содержит строку запроса после имени сервера, например: /company/test.php?id=3&parent=4;
- request_filename – то же самое, что и request_uri;
- time_year – текущий год;
- time_mon – текущий месяц;
- time_day – текущий день;
- time_hour – текущий час;
- time_min – текущая минута;
- time_sec – текущая секунда;
- time_wday – текущий день недели;
- time – текущее время.

Для того чтобы использовать серверные переменные в директивах модуля mod_rewrite, их следует писать в формате: *%{имя_переменной)*. Например:

RewriteCond %{HTTP_USER_AGENT} Opera

Рассмотрим пример (листинг 24).

Листинг 24. *Скрытая подмена страницы oldpage.html на страницу* newpage.html

```
RewriteEngine on
RewriteBase /
RewriteRule ^oldpage\.html$ newpage.html
```

Здесь, при запросе страницы oldpage.html, URL преобразовывается в запрос к странице newpage.html. При этом подмена URL происходит незаметно для посетителя. В адресной строке продолжает отображаться имя страницы oldpage.html.
Если необходимо использовать внешнюю переадресацию, чтобы посетителю был отображен реальный адрес страницы, то в директиве RewriteRule нужно указать флаг [R] (листинг 25).

Листинг 25. *Переадресация запроса страницы oldpage.html на страницу newpage.html*

```
RewriteEngine On
RewriteBase /
RewriteRule ^oldpage\.html$ newpage.html [R]
```

Рассмотрим задачу подмены обращений к пользовательским каталогам формата **http://www.server.ru/~username/** на обращение по адресам **http:// www.server.ru/users/username/** (листинг 26).

Листинг 26. *Подмена URL каталогов пользователей*

```
RewriteEngine on
RewriteBase /
RewriteRule /~([^/]+)?(/*)/ /users/$1/$2 [R]
```

Директива RewriteCond позволяет задавать условия при преобразовании адресов. Синтаксис директивы RewriteCond:

RewriteCond *контрольная_строка шаблон_поиска [флаги]*

Если шаблон поиска найден в контрольной строке, т. е. условие выполнено, то выполняются директивы, указанные сразу за директивой RewriteCond, в противном случае управление переходит к следующему блоку директив.
Рассмотрим задачу изменения адресов в зависимости от типа браузера (листинг 27).

Листинг 27. *Изменение адреса в зависимости от типа браузера*

```
RewriteEngine On
RewriteBase /
RewriteCond %{HTTP_USER_AGENT} Opera
RewriteCond %{REQUEST_FILENAME} server\.ru/$|server\.ru/index\.php
RewriteRule ^(.*) index_opera.php?%{QUERY_STRING} [L]
RewriteCond %{HTTP_USER_AGENT} Netscape
```

```
RewriteCond %{REQUEST_FILENAME} server\.ru/$|server\.ru/index\.php
RewriteRule ^(.*) index_netscape.php?%{QUERY_STRING} [L]
```

Первое условие проверяет наличие в заголовке user-Agent подстроки брега. Если подстрока opera найдена, то проверяется следующее условие: направлен запрос к странице index.php, либо к корню сайта без указания имени файла. В обоих случаях следует подставить индексный файл, адаптированный под браузер Opera – index_opera.php. Флаг [L] (Last) говорит о том, что дальнейшее выполнение директив следует прекратить.
Если же первое условие:

```
RewriteCond % {HTTP_USER_AGENT} Opera
```

не выполнено, то управление переходит на следующий блок, начинающийся с условия:

```
RewriteCond %{HTTP_USER_AGENT} Netscape
```

В нем проверяется принадлежность браузера к семейству Netscape, и если условие истинно, то выполняется переадресация на страницу index_netscape.php, адаптированную под браузер Netscape. Если же условие проверки принадлежности браузера к Netscape также не выполнено, то преобразование адресов не происходит.
С помощью директив модуля mod_rewrite легко подвергать URL модификации в зависимости от времени суток, месяца, года и т. п. (листинг 28).

Листинг 28. *Изменение адреса в зависимости от времени суток*

```
RewriteEngine On
RewriteBase /
RewriteCond %{TIME_HOUR}%{TIME_MIN} >0700
RewriteCond %{TIME_HOUR}%{TIME_MIN} <1900
RewriteRule ^page\.php$ page_day.php?%{QUERY_STRING} [L]
RewriteRule ^page\.php$ page_night.php?%{QUERY_STRING} [L]
```

С помощью серверных переменных time_hour и time_min проверяется текущее время. Если оно лежит в диапазоне от 7:00 до 19:00, т. е. выполняются оба условия RewriteCond, то запрошенный адрес преобразуется в запрос к странице page_day.php. Ключ [L] прекращает дальнейшую обработку директив. Если хотя бы одно условие не выполнено, то осуществляется запрос к странице page_night.php.
Рассмотрим задачу запрета посещений сайта нежелательными роботами1. Управление посещениями сайта роботами возложено на файл robots.txt, но далеко не все роботы следуют указаниям этого файла. В таком случае для блокирования посещений сайта некорректно работающими роботами можно применить директивы модуля mod_rewrite (листинг 29).

Примечание

Файл robots.txt – это текстовый файл, который используется для управления посещения сайта роботами поисковых систем. С его помощью можно запретить роботам посещать определенные разделы и страницы сайта. Директивы файла robots.txt имеют рекомендательный

характер и могут игнорироваться некорректно работающими роботами, поэтому для достижения гарантированного запрета следует использовать директивы Apache.

В терминах протокола HTTP робот поисковой системы относится к интеллектуальному агенту. Это автономная программа, осуществляющая обход Web-ресурса для сбора текстовой информации, расположенной на сайте. Полученная роботом информация используется для построения каталога поисковой системы.

Листинг 29. *Блокирование посещений сайта нежелательными роботами*

```
RewriteEngine On
RewriteBase /
RewriteCond %{HTTP_USER_AGENT} ^robot
RewriteCond %{REMOTE_ADDR} ^196\.56\.78\.18
RewriteRule ^(.*) for_bad_robots.php
```

Здесь в двух условиях RewriteCond проверяется имя робота, переданное в заголовке user-Agent, а затем IP-адрес, с которого робот пришел на сайт. Если оба условия выполняются, то осуществляется запрос к странице for_bad_robots.php.

Примечание

Более подробно о директивах модуля mod_rewrite вы можете почитать по адресу **http://httpd.apache.org/docs-2.0/mod/mod_rewrite.html**.