Web development lectures Documentation

Выпуск 0.0.0

Dmitry Svintsov

сент. 27, 2017

Оглавление

Описание:

Глава 1

Введение

Преподаватель

Фамилия Имя Отчество
Ассистент кафедры ИИТ
me@example.com

Студенты

1 курс

Студенты должны:

- Знать *ОС* подобные *Unix* или *Windows*, основы алгоритмазации и программирования;
- Иметь компьютер с подключением к *Интернет* и графической *ОС* или отдельный виртуальный образ для работы на *ПК* лабораторных классах.

Обзор курса

Курс объемом 160 учебных часов рассчитан на 1-ый семестр. Состоит из 80 часов лекционных занятий, 80 часов практической работы. В качестве самостоятельной работы предусмотрены домашние задания и курсовая работа. По окончанию обучения студенты сдают экзамен. Допуском к экзамену является выполнение всех домашних работ и сдача курсовой работы.

Глава 2

Список литературы

- http://www.4stud.info - Учебно-методические материалы для студентов кафедры АСОИУ
- http://www.tutorialspoint.com
- http://www.htmlbook.ru

Глава 3

Лекция №1:

История WWW

Примечание: Интернет — это глобальная компьютерная сеть, объединяющая сотни миллионов компьютеров в общее информационное пространство. Интернет представляет свою инфраструктуру для прикладных сервисов различного назначения, самым популярным из которых является Всемирная Паутина – World Wide Web (www).

World Wide Web (www, web, рус.: веб, Всемирная Паутина) — распределенная информационная система, предоставляющая доступ к гипертекстовым документам по протоколу HTTP.

WWW — сетевая технология прикладного уровня стека TCP/IP, построенная на клиент-серверной архитектуре и использующая инфраструктуру Интернет для взаимодействия между сервером и клиентом (рис. 1).

Серверы www (веб-серверы) — это хранилища гипертекстовой (в общем случае) информации, управляемые специальным программным обеспечением.

Документы, представленные в виде гипертекста называются веб-страницами. Несколько веб-страниц, объединенных общей тематикой, оформлением, связанных гипертекстовыми ссылками и обычно находящихся на одном и том же веб-сервере, называются веб-сайтом.

Для загрузки и просмотра информации с веб-сайтов используются специальные программы — браузеры, способные обрабатывать гипертектовую разметку и отображать содержимое веб-страниц.

Рис. 3.1: Рис. 1. Архитектура сервиса WWW

В основе www — взаимодействие между веб-сервером и браузерами по протоколу HTTP (HyperText Transfer Protocol). Веб-сервер — это программа, запущенная на сетевом компьютере и ожидающая клиентские запросы по протоколу HTTP. Браузер может обратиться к веб-серверу по доменному имени или по ip-адресу, передавая в запросе идентификатор требуемого ресурса. Получив запрос от клиента,

сервер находит соответствующий ресурс на локальном устройстве хранения и отправляет его как ответ. Браузер принимает ответ и обрабатывает его соответствующим образом, в зависимости от типа ресурса (отображает гипертекст, показывает изображения, сохраняет полученные файлы и т.п.).

Основной тип ресурсов Всемирной паутины — гипертекстовые страницы. Гипертекст — это обычный текст, размеченный специальными управляющими конструкциями — тегами. Браузер считывает теги и интерпретирует их как команды форматирования при выводе информации. Теги описывают структуру документа, а специальные теги, якоря и гиперссылки, позволяют установить связи между веб-страницами и перемещаться как внутри веб-сайта, так и между сайтами.

Примечание: Т. Дж. Бернерс-Ли — «отец» Всемирной паутины

Сэр Тимоти Джон Бернерс-Ли — британский учёный-физик, изобретатель Всемирной паутины (совместно с Робертом Кайо), автор URI, HTTP и HTML. Действующий глава Консорциума Всемирной паутины (W3C). Автор концепции семантической паутины и множества других разработок в области информационных технологий. 16 июля 2004 года Королева Великобритании Елизавета II произвела Тима Бернерса-Ли в Рыцари-Командоры за «службу во благо глобального развития Интернета».

Компоненты WWW

Функционирование сервиса обеспечивается четырьмя составляющими:

- URL/URI — унифицированный способ адресации и идентификации сетевых ресурсов;
- HTML — язык гипертекстовой разметки веб-документов;
- HTTP — протокол передачи гипертекста;
- CGI — общий шлюзовый интерфейс, представляющий доступ к серверным приложениям.

Адресация веб-ресурсов. URL, URN, URI

Для доступа к любым сетевым ресурсам необходимо знать где они размещены и как к ним можно обратиться. Во Всемирной паутине для обращения к веб-документам изначально используется стандартизованная схема адресации и идентификации, учитывающую опыт адресации и идентификации таких сетевых сервисов, как e-mail, telnet, ftp и т.п. — URL, Uniform Resource Locator.

URL (RFC 1738) — унифицированный локатор (указатель) ресурсов, стандартизированный способ записи адреса ресурса в www и сети Интернет. Адрес URL имеет гибкую и расширяемую структуру для максимально естественного указания местонахождения ресурсов в сети. Для записи адреса используется ограниченный набор символов ASCII. Общий вид адреса можно представить так:

<схема>://<логин>:<пароль>@<хост>:<порт>/<полный-путь-к-ресурсу>

Где:

схема
: схема обращения к ресурсу: http, ftp, gopher, mailto, news, telnet, file, man, info, whatis, ldap, wais и т.п.

логин:пароль
: имя пользователя и его пароль, используемые для доступа к ресурсу

хост
: доменное имя хоста или его IP-адрес.

порт
: порт хоста для подключения

полный-путь-к-ресурсу
: уточняющая информация о месте нахождения ресурса (зависит от протокола).

Примеры URL:

http://example.com #запрос стартовой страницы по умолчанию
http://www.example.com/site/map.html #запрос страницы в указанном каталоге
http://example.com:81/script.php #подключение на нестандартный порт
http://example.org/script.php?key=value #передача параметров скрипту
ftp://user:pass@ftp.example.org #авторизация на ftp-сервере
http://192.168.0.1/example/www #подключение по ip-адресу
file:///srv/www/htdocs/index.html #открытие локального файла
gopher://example.com/1 #подключение к серверу gopher
mailto://user@example.org #ссылка на адрес эл.почты

В августе 2002 года RFC 3305 анонсировал устаревание URL в пользу URI (Uniform Resource Identifier), еще более гибкого способа адресации, вобравшего возможности как URL, так и URN (Uniform Resource Name, унифицированное имя ресурса). URI позволяет не только указавать местонахождение ресурса (как URL), но и идентифицировать его в заданном пространстве имен (как URN). Если в URI не указывать местонахождение, то с его помощью можно описывать ресурсы, которые не могут быть получены непосредственно из Интернета (автомобили, персоны и т.п.). Текущая структура и синтаксис URI регулируется стандартом RFC 3986, вышедшим в январе 2005 года.

Язык гипертекстовой разметки HTML

HTML (**‘HyperText Markup Language <>‘_**) — стандартный язык разметки документов во Всемирной паутине. Большинство веб-страниц созданы при помощи языка HTML. Язык HTML интерпретируется браузером и отображается в виде документа, в удобной для человека форме. HTML является приложением SGML (стандартного обобщённого языка разметки) и соответствует международному стандарту ISO 8879.

HTML создавался как язык для обмена научной и технической документацией, пригодный для использования людьми, не являющимися специалистами в области вёрстки. Для этого он представляет небольшой (сравнительно) набор структурных и семантических элементов — тегов. С помощью HTML можно легко создать относительно простой, но красиво оформленный документ. Изначально язык HTML был задуман и создан как средство структурирования и форматирования документов без их привязки к средствам воспроизведения (отображения). В идеале, текст с разметкой HTML должен единообразно воспроизводиться на различном оборудовании (монитор ПК, экран органайзера, ограниченный по размерам экран мобильного телефона, медиа-проектор). Однако современное применение

HTML очень далеко от его изначальной задачи. Со временем основная идея платформонезависимости языка HTML стала жертвой коммерциализации www и потребностей в мультимедийном и графическом оформлении.

Протокол HTTP

HTTP (**‘HyperText Transfer Protocol <>‘_**) — протокол передачи гипертекста, текущая версия HTTP/1.1 (RFC 2616). Этот протокол изначально был предназначен для обмена гипертекстовыми документами, сейчас его возможности существенно расширены в сторону передачи двоичной информации.

HTTP — типичный клиент-серверный протокол, обмен сообщениями идёт по схеме «запрос-ответ» в виде ASCII-команд. Особенностью протокола HTTP является возможность указать в запросе и ответе способ представления одного и того же ресурса по различным параметрам: формату, кодировке, языку и т. д. Именно благодаря возможности указания способа кодирования сообщения клиент и сервер могут обмениваться двоичными данными, хотя данный протокол является символьно-ориентированным.

HTTP — протокол прикладного уровня, но используется также в качестве «транспорта» для других прикладных протоколов, в первую очередь, основанных на языке XML (SOAP, XML-RPC, SiteMap, RSS и проч.).

Общий шлюзовый интерфейс CGI

CGI (**‘Common Gateway Interface <>‘_**) — механизм доступа к программам на стороне веб-сервера. Спецификация CGI была разработана для расширения возможностей сервиса www за счет подключения различного внешнего программного обеспечения. При использовании CGI веб-сервер представляет браузеру доступ к исполнимым программам, запускаемым на его (серверной) стороне через стандартные потоки ввода и вывода.

Интерфейс CGI применяется для создания динамических веб-сайтов, например, когда веб-страницы формируются из результатов запроса к базе данных. Сейчас популярность CGI снизилась, т.к. появились более совершенные альтернативные решения (например, модульные расширения веб-серверов).

Программное обеспечение сервиса WWW

Веб-серверы

Веб-сервер — это сетевое приложение, обслуживающее HTTP-запросы от клиентов, обычно веб-браузеров. Веб-сервер принимает запросы и возвращает ответы, обычно вместе с HTML-страницей, изображением, файлом, медиа-потоком или другими данными. Веб-серверы — основа Всемирной паутины. С расширением спектра сетевых сервисов веб-серверы все чаще используются в качестве шлюзов для серверов приложений или сами представляют такие функции (например, Apache Tomcat).

Созданием программного обеспечения веб-серверов занимаются многие разработчики, но наибольшую популярность (по статистике http://netcraft.com) имеют такие программные продукты, как Apache (Apache Software Foundation), IIS (Microsoft), Google Web Server (GWS, Google Inc.) и nginx.

Apache — свободное программное обеспечение, распространяется под совместимой с GPL лицензией. Apache уже многие годы является лидером по распространенности во Всемирной паутине в силу своей надежности, гибкости, масштабируемости и безопасности.

IIS (Internet Information Services) — проприетарный набор серверов для нескольких служб Интернета, разработанный Майкрософт и распространяемый с серверными операционными системами семейства Windows. Основным компонентом IIS является веб-сервер, также поддерживаются протоколы FTP, POP3, SMTP, NNTP.

Google Web Server (GWS) — разработка компании Google на основе веб-сервера Apache. GWS оптимизирован для выполнения приложений сервиса Google Applications.

nginx [engine x] — это HTTP-сервер, совмещенный с кэширующим прокси-сервером. Разработан И. Сысоевым для компании Рамблер. Осенью 2004 года вышел первый публично доступный релиз, сейчас nginx используется на 9-12% веб-серверов. Браузеры

Браузер, веб-обозреватель (web-browser) — клиентское приложение для доступа к веб-серверам по протоколу HTTP и просмотра веб-страниц. Как правило браузеры дополнительно поддерживают и ряд других протоколов (например ftp, file, mms, pop3).

Первые HTTP-клиенты были консольными и работали в текстовом режиме, позволяя читать гипертекст и перемещаться по ссылкам. Сейчас консольные браузеры (такие, как lynx, w3m или links) практически не используются рядовыми посетителями веб-сайтов. Тем не менее такие браузеры весьма полезны для веб-разработчиков, так как позволяют «увидеть» веб-страницу «глазами» поискового робота.

Исторически первым браузером в современном понимании (т.е. с графическим интерфейсом и т.д.) была программа NCSA Mosaic, разработанная Марком Андерисеном и Эриком Бина. Mosaic имел довольно ограниченные возможности, но его открытый исходный код стал основой для многих последующих разработок.

Существует большое число программ-браузеров, но наибольшей популярностью пользуются следующие:

Internet Explorer (IE) — браузер, разработанный компанией Майкрософт и тесно интегрированный с ОС Windows. Платформозависим (поддержка сторонних ОС прекращена, начиная с версии 5). Единственный браузер, напрямую поддерживающий технологию ActiveX. Не полностью совместим со стандартами W3C, в связи с чем требует дополнительных затрат от веб-разработчиков.

Firefox — свободный кроссплатформенный браузер, разрабатываемый Mozilla Foundation и распространяемый под тройной лицензией GPL/LGPL/MPL. В основе браузера — движок Gekko, который изначально создавался для Netscape Communicator. Однако, вместо того, чтобы предоставить все возможности движка в стандартной поставке, Firefox реализует лишь основную его функциональность, предоставляя пользователям возможность модифицировать браузер в соответствии с их требованиями через поддержку расширений (add-ons), тем оформления и плагинов.

Safari — проприетарный браузер, разработаный корпорацией Apple и входящий в состав операционной системы Mac OS X. Бесплатно распространяется для операционных систем семейства Microsoft Windows. В браузере используется уникальный по производительности интерпретатор JavaScript и еще ряд интересных для пользователя решений, которые отсутствуют или не развиты в других браузерах.

Chrome — кроссплатформенный браузер с открытым исходным кодом, разрабатываемый компанией Google. Первая стабильная версия вышла 11 декабря 2008 года. В отличие от многих других браузеров, в Chrome каждая вкладка является отдельным процессом. В случае если процесс обработки содержимого вкладки зависнет, его можно будет завершить без риска потери данных других вкладок. Еще одна особенность — интеллектуальная адресная строка (Omnibox). К возможности автозаполнения она добавляет поисковые функции с учетом популярности сайта, релевантности и пользовательских предпочтений (истории переходов).

Opera — кроссплатформенный многофункциональный веб-браузер, впервые представленный в 1994 году группой исследователей из норвежской компании Telenor. Дальнейшая разработка ведется Opera Software ASA. Этот браузер обладает высокой скоростью работы и совместим с основными стандартами. Отличительными особенностями Opera долгое время являлись многостраничный интерфейс и возможность масштабирования веб-страниц целиком. На разных этапах развития в Opera были интегрированы возможности почтового/новостного клиента, адресной книги, клиента сети BitTorrent, агрегатора RSS, клиента IRC, менеджера закачек, WAP-браузера, а также поддержка виджетов — графические модулей, работающих вне окна браузера. Роботы-«пауки»

Наряду с браузерами, ориентированными на пользователя, существуют и специализированные клиенты-роботы («пауки», «боты»), подключающиеся к веб-серверам и выполняющие различные задачи автоматической обработки гипертекстовой информации. Сюда относятся, в первую очередь, роботы поисковых систем, таких как google.com, yandex.ru, yahoo.com и т.п., выполняющие обход веб-сайтов для последующего построения поискового индекса.

Типы сайтов

Примечание: Одним из первых вопросов, на который требуется получить ответ на этапе проектирования веб-сайта: какого типа этот сайт должен быть? От этого будет зависеть сложность, трудоемкость и стоимость разработки.

Любая классификация подразумевает выделение одного или нескольких группировочных признаков, соответствие которому (которым) и указывает на принадлежность к определенной категории. В большинстве случаев совокупность объектов может быть классифицирована несколькими способами, в зависимости от выбранных критериев. Веб-сайты не являются исключением и также подлежат классификации. Рассмотрим основные типы сайтов* по ряду формальных признаков (в первую очередь по содержанию и возможностям).

Статические и динамические сайты

Статические сайты — это сайты, состоящие из веб-страниц, написанных целиком на html и хранящихся на сервере в том виде, в котором их создал веб-мастер. Такие страницы могут содержать клиентские скрипты, графику и встаиваемые интерактивные элементы, но они не влияют на содержимое страницы, пока не будут загружены в браузер. При этом на сервере страница никак не изменится и будет отображаться одинако для всех пользователей.

Содержимое **динамических сайтов** генерируется «на лету»: запрашиваемая веб-страница формируется программно, с помощью серверного приложения (из базы данных, включаемых файлов и т.п.). Таким образом, одна и та же страница может выглядеть по разному для разных пользователей.

Информационные сайты и веб-приложения

К категории **информационных сайтов** относятся те, что представляют пользователям доступ к разного рода информации: текстовой, графической, мультимедийной. К этому типу относится большинство существующих сайтов и поэтому внутри группы уместна дальнейшая классификация, например, по характеру контента: информационно-тематические сайты, новостные, развлекательные, каталоги и справочники, онлайн-энциклопедии и словари, агрегаторы и т.п.

Еще одна внутренняя группировка, влияющая на то, каким должен быть сайт информационного типа — это его тематика. Здесь мы не станем перечислять список возможных тем, представляя возможность вам самостоятельно ознакомиться в онлайн-каталогах типа dmoz.org или подобных.

Веб-приложения, часто называемые веб-сервисами, являются серверными программами, решающими определенные задачи. Веб-приложения могут быть самодостаточными, а могут выполнять специфичные функции в виде компонентов информационного сайта, наделяя его интерактивностью и/или расширяя возможности. Смысл сказанного проще пояснить на нескольких примерах: Google Drive — пакет онлайновых офисных приложений, WebMoney — платежная система, ColorSchemeDesigner — онлайновый инструмент для дизайнеров. Подсистема авторизации или онлайн-опрос — это примеры сервисных веб-приложений, интегрируемых в структуру сайта.

В категорию веб-приложений отчасти можно отнести и клиентсткие программы на javascript.

Порталы и специализированные сайты

Портал — это многофункциональный сайт, являющийся «шлюзом» к различной информации и онлайновым-сервисам. Сайты портального типа сложны в разработке и сопровождении, но они охватывают максимальное число пользователей. Все популярные поисковые системы являются сайтами этого типа.

В категорию **специализированных сайтов** относятся веб-сайты, ориентированные на определенную тематику, функциональность и/или целевую аудиторию. Рассмотрим некоторые типы сайтов, попадающих в эту категорию:

- Персональный сайт — ресурс, созданный и поддерживаемый автором и представляющий, в первую очередь, информацию о нем самом и его интересах. Персональные сайты часто являются статичными и размещаются на бесплатных хостинговых площадках.
- Блог (онлайн-дневник) — продвинутая разновидность персонального сайта, где информация представляется в хронологическом порядке (как в дневнике). Ирония в том, что свои рукописные дневники люди предпочитают никому не показывать, а к онлайн-версии пытаются привлечь наибольшее число пользователей.
- Социальная сеть — онлайновый сервис, ориентированный на поиск, установление контактов и общение между пользователями системы. Ранее популярные сервисы онлайн-знакомств теперь переняли многие черты социальных сетей, поэтому отнесены в эту же группу.
- Форум — площадка для общения пользователей, работающая на дискуссионном принципе: пользователь создает тему, прочие участники могут присоединиться к ее обсуждению. Форумы существовали еще в UseNet и Fido и, хотя их популярность несколько уменьшилась с появлением социальных сетей, все еще остаются источником полезной информации и знаний.
- Корпоративный сайт — официальное представительство предприятия или организации в глобальной сети. Размеры и функциональность таких сайтов варьируются от очень маленьких «визиток» в несколько статичных страниц с общей информацией до корпоративных порталов. Реальность такова, что предприниматели рассматривают Веб по меньшей мере как рекламную площадку.
- Интернет-магазин — самая популярная группа сайтов для электронной коммерции (ecommerce). Интернет-магазин может выступать как «витрина» и представлять информацию о товарах и/или услугах, но основная функциональность все-таки направлена на совершение покупок в Интернет. Для этого сайт интегрируется с платежным шлюзом, обслуживающим банковские переводы или электронные деньги. Это дает возможность пользователю оплачивать покупки онлайн, а владельцу интернет-магазина обрабатывать заказы и получать оплату за них через сайт.
- Wiki-сайт — специфичная разновидность сайтов, характерной чертой которых является коллаборативное управление информацией. Пользователи могут не только генерировать контент, но и редактировать информацию, добавленную другими пользователями, обсуждать, принимать и отклонять правки других пользователей так же, как это сделано в Wikipedia.org.

По аналогии с рассмотренными примерами вы можете составить собственное представление о таких специализированных типах сайтов, как галереи изображений, каталоги программного обеспечения, файлообменные сайты и т.п.

Современный профессионально разработанный веб-сайт сложно отнести к какому-то одному типу. Одна из основных причин этого — использование веб-разработчиками готовых систем управления контентом (CMS). Классификация задач и определение основного типа будущего сайта позволяет выбрать наиболее подходящую CMS и набор модулей для нее или принять решение о разработке сайта «с нуля».

Рабочее окружение

Операционная система

Операционная система в данном курсе не имеет значения, подойдет любая распространенная ОС с графическим интерфейсом. Например *Linux*, *MacOS* или *Windows*.

Текстовый редактор

За текстовым редактором Веб-программист проводит 90% времени, поэтому нужно ответственно подойти к этому выбору. Можно использовать любой понятный вам и удобный в использовании текстовый редактор.

Критерием должны стать:

- простата использования
- удобный интерфейс
- возможность гибкой настройки
- кроссплатформенность
- подсветка синтаксиса
- автодополнение кода

Все эти критерии удовлетворяют такие редакторы как *Vim* и *Emacs*. Так же среди программистов встречаются менее функциональные *Bred3*, *Notepad++*, *SublimeText* и другие.

Веб-браузер

Можно выбрать один из самых популярных браузеров, на сегодняшний день, *Mozilla Firefox* или *Google Chrome* или любой другой соответствующий Веб-стандартам.

Система контроля версий

В данном курсе для выполнения самостоятельных работ потребуются знания системы контроля версий *git* и учетная запись в сервисе *GitHub*.

Глава 4

Лекция №2

Введение в HTML

Примечание: HTML - это язык разметки, который представляет простые правила оформления и компактный набор структурных и семантических элементов разметки (тегов). HTML позволяет описывать способ представления логических частей документа (заголовки, абзацы, списки и т.д.) и создавать веб-страницы разной сложности.

Изначально язык HTML (HyperText Markup Language) был задуман и создан как средство структурирования и форматирования документов без привязки к средствам отображения. В идеале, гипертекстовый документ должен одинаково выглядеть на различных устройствах (монитор ПЭВМ, экран ПДА или мобильного телефона, принтер, медиа-проектор и т.п.).

Разработкой спецификаций языка HTML и утверждением их в качестве официальных стандартов занимается Консорциум всемирной паутины (W3C). Помимо W3C, в развитии языка участвуют IT-компании и сообщество разработчиков.

- Официальной спецификации HTML 1.0 не существует. До 1995 года существовало множество неофициальных спецификаций HTML, появившихся в ходе браузерных войн.
- RFC 1866 — HTML 2.0, одобренный как официальный стандарт 22 сентября 1995 года;
- HTML 3 (март 1996) - не нашла поддержки у разработчиков
- HTML 3.2 — 14 января 1997 года;
- HTML 4.0 — 18 декабря 1997 года (многие унаследованные элементы были отмечены как устаревшие и нерекомендованные к использованию (англ. deprecated).);
- HTML 4.01 — 24 декабря 1999 года (версия включала малозаметные, но существенные изменения по сравнению с предыдущей);
- ISO/IEC 15445:2000 (ISO HTML, основан на HTML 4.01 Strict) — 15 мая 2000 года.
- HTML 5 — разработан и принят W3C совместно с сообществом WHATWG.

Примечание: HTML не является языком программирования, но веб-страницы могут содержать встроенные или загружаемые программы на скриптовых языках (в первую очередь Javascript) и программы-апплеты на языке Java.

Элементы гипертекста

Структура HTML-документа

HTML-документ состоит из текста, который представляет собой информационное содержимое и специальных средств языка HTML — тегов разметки, которые определяют структуру и внешний вид документа при его отображении браузером. Структура HTML-документа (рис. 1) довольно проста:

1. Описание документа начинается с указания его типа (секция DOCTYPE).
2. **Текст документа заключается в тег <html>. Текст документа состоит из заголовка и тела, которы**

    - В заголовке (<head>) указывают название HTML-документа и другие параметры, которые браузер будет использовать при отображении документа.
    - Тело документа (<body>) — это та часть, в которую помещается собственно содержимое HTML-документа. Тело включает предназначенный для отображения текст и управляющую разметку документа (теги), которые используются браузером.

Рис. 4.1: Рис. 1. Общая структура веб-страницы

Наличие секции DOCTYPE позволяет указать браузеру, какой тип документа ему предстоит разбирать, т.е, какие требования нужно выполнять при обработке гипертекста.

Заголовок предназначен для размещения метаинформации, описывающей веб-документ как таковой.

Блок <body> содержит то, что нужно показать пользователю: текст, изображения, внедренные объекты и пр.

Ниже приведен простой пример html-разметки.

```
<html>
<head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8" >
    <title>Почему откровенна веданта?</title>
</head>
<body>
<h1>Почему Откровенна Веданта?</h1>
<h2>Трактат о амбивалентности бытия, сомнениях и адживике</h2>
<p>Философия нетривиальна и это не умозаключение, а плод переработки бытийного.
Моцзы, Сюнъцзы и др. считали, что сомнение естественно понимает под собой гений,
изменяя привычную реальность. Отношение к современности, как принято считать,
непредсказуемо, а созерцание, конечно, транспонирует гравитационный парадокс,
ломая рамки привычных представлений. Позитивизм преобразует дуализм, не учитывая
мнения авторитетов. Можно предположить, что вещь в себе представляет собой типичный
здравый смысл, учитывая опасность, которую представляли собой писания Дюринга.
При этом буквы A, B, I, O символизируют соответственно суждения:</p>
<ul>
<li>общеутвердительное;</li>
<li>общеотрицательное;</li>
<li>частноутвердительное;</li>
<li>частноотрицательное.</li>
</ul></p>
```

```
<p>Структурализм, как принято считать, подчеркивает закон исключённого третьего,
открывая новые горизонты. Адживика преобразует неоднозначный предмет деятельности,
tertium non datur. Согласно предыдущему, дуализм оспособляет примитивный бабувизм,
ломая рамки привычных представлений. Наряду с этим вещь в себе дискредитирует
сенсибельный принцип восприятия.</p>
<p>В целом, представляется логичным, что адживика трансформирует субъективный
гедонизм, тогда как бабувизм контролирует предмет деятельности, tertium non datur.</p>
</body>
</html>
```

DOCTYPE

Секция DOCTYPE указывает браузеру тип документа и версию использованного языка разметки. Здесь также указывается название и область видимости описания этого языка и адрес файла dtd (document type definition).

Примеры DOCTYPE:

- **<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR**
  Гипертекстовый документ в формате HTML 4.01, содержащий фреймы.
- **<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/st**
  Гипертекстовый документ в формате HTML 4.01 со строгим синтаксисом (т.е. не использованы устаревшие и не рекомендованные теги).
- **<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/**
  Гипертекстовый документ в формате HTML 4.01 с нестрогим («переходным») синтаксисом (т.е. использованы устаревшие или не рекомендованные теги и атрибуты).
- **<!DOCTYPE HTML>** Пока не стандартизованное объявление для документов HTML5.

Стандарт требует, чтобы секция DOCTYPE присутствовала в документе, т.к. это позволяет ускорить и улучшить обработку гипертекста. Это достигается за счет того, что браузер может не делать предположений о том, как интерпретировать теги, а свериться со стандартным определением (файлом .dtd). Подробное описание DOCTYPE — на сайте Консорциума W3C.

Мета-теги

Мета-тег HTML — это элемент разметки html, описывающий свойства документа как такового (метаданные). Назначение мета-тега определяется набором его атрибутов, которые задаются в теге <meta>.

Мета-теги размещают в блоке <head>...</head> веб-страницы. Они не являются обязательными элементами, но могут быть весьма полезны.

Пример описания метаданных:

```
<head>
<meta name="author" content="строка"> - автор веб-документа
<meta name="date" content="дата"> - дата последнего изменения веб-страницы
<meta name="copyright" content="строка"> - авторские права
<meta name="keywords" content="строка"> - список ключевых слов
<meta name="description" content="строка"> - краткое описание (реферат)
<meta name="ROBOTS" content="NOINDEX, NOFOLLOW"> - запрет на индексирование
<meta http-equiv="content-type" content="text/html; charset=UTF-8"> - тип и кодировка
<meta http-equiv="expires" content="число"> - управление кэшированием
<meta http-equiv="refresh" content="число; URL=адрес"> - перенаправление
</head>
```

Теги

Тег (html-тег, тег разметки) — управляющая символьная последовательность, которая задает способ отображения гипертекстовой информации.

HTML-тег состоит из имени, за которым может следовать необязательный список атрибутов. Весь тег (вместе с атрибутами) заключается в угловые скобки <>:

<имя_тега [атрибуты]>

Как правило, теги являются парными и состоят из начального и конечного тегов, между которыми и помещается информация. Имя конечного тега совпадает с именем начального, но перед именем конечного тега ставится косая черта / (<html>...</html>). Конечные теги никогда не содержат атрибутов. Некоторые теги не имеют конечного элемента, например тег <img>. Регистр символов для тегов не имеет значения.

Примеры часто используемых тегов HTML:

```
<html>...</html> - контейнер гипертекста
<head>...</head> - контейнер заголовка документа
<title>...</title> - название документа (то, что отображается в заголовке окна браузера)
<body>...</body> - контейнер тела документа
<div>...</div> - контейнер общего назначения (структурный блок)
<hN>...</hN> -  заголовок N-ного уровня  (N = 1...6)
<p>...</p> - основной текст
<a>...</a> -  гиперссылка
<ol>...</ol> - нумерованный список
<ul>...</ul> - маркированный список
<li>...</li> - элемент списка
<table>...</table> - контейнер таблицы
<tr>...</tr> - строка таблицы
<td>...</td> - ячейка таблицы
<img>...</img> - изображение
<form>...</form> - форма
<i>...</i> -  отображение текста курсивом
<b>...</b> -  отображение текста полужирным шрифтом
<em>...</em> -  выделение (курсивом)
<strong>...</strong> -  усиление (полужирным шрифтом)
<br> - принудительный разрыв строки
```

Теги могут быть вложены, при этом форматирование внутреннего тега имеет преимущество перед внешним. При использовании вложенных тегов их нужно закрывать, начиная с самого последнего и двигаясь к первому:

```
<!-- Список как пример использования вложенных тегов -->
<ol>
<li>Элемент списка</li>
<li>Второй элемент списка</li>
</ol>
<div>
    <h2>Заголовок второго уровня</h2>
    <p>и основной текст</p>
    внутри логического блока
</div>
```

Примечание: Примечание: Браузеры обычно лояльно относятся к отсутствию конечных тегов у парных элементов и более-менее правильно отображают парные элементы уровня блока (p, li и т.п.),

особенно в простых веб-документах. Тем не менее, рекомендуется следить за наличием закрывающих тегов и использовать их, чтобы избежать ошибок при воспроизведении документа.

Полный список тегов можно найти в документации на соответствующую версию языка HTML (см., например HTML 3.2, HTML 4.01, XHTML 1.1 и др.).

Атрибуты

Атрибуты — это пары вида «свойство = значение», уточняющие представление соответствующего тега:

<тег атрибут=”значение”>...</тег>

Атрибуты указывают в начальном теге, несколько атрибутов разделяют одним или несколькими пробелами, табуляцией или символами конца строки. Значение атрибута, если таковое имеется, следует за знаком равенства, стоящим после имени атрибута. Порядок записи атрибутов в теге не важен. Если значение атрибута — одно слово или число, то его можно просто указать после знака равенства, не выделяя дополнительно. Все остальные значения необходимо заключать в кавычки, особенно если они содержат несколько разделенных пробелами слов.

Примечание: Примечание: Несмотря на необязательность кавычек, их все же стоит всегда использовать.

Атрибуты могут быть обязательными и не обязательными. Необязательные атрибуты могут быть опущены, тогда для тега применяется значение этого атрибута по умолчанию. Если не указан обязательный атрибут, то содержимое тега скорее всего будет отображено неправильно.

Краткий список некоторых часто используемых атрибутов и их возможных значений:

```
style="описание_стилей" - локальные стили
src="адрес" - адрес (URI) источника данных (например картинки или скрипта)
align="left|center|right|justify" -  выравнивание, по умолчанию left (по левому краю)
width="число" - ширина элемента (в пикселях, пиках, поинтах и др.)
height="число" - высота элемента (в пикселях, пиках, поинтах и др.)
href="адрес" - гиперссылка, адрес (URI) на который будет выполнен переход
name="имя" - имя элемента
id="идентификатор"  - уникальный (в пределах веб-страницы) идентификатор элемента
size="число" - размер элемента
class="имя_класса" - имя класса во встроенной или связанной таблице стилей
title="строка" - название элемента
alt="строка" - альтернативный текст
```

Гиперссылки

Гиперссылка - это особым образом помеченный фрагмент веб-страницы (текст, изображение и др.), который связан с другим документом. Для указания гиперссылок используется тег <a>. Гиперссылки позволяют перемещаться между связанными веб-страницами.

```
<a href="http://example.com/">Пример</a>
<a href="ftp://example.com/archive.tar.gz">Скачать файл</a>
<a href="mailto://user@mail.example.com" title="Обратная связь">user@mail.example.com</a>
```

Переход по ссылкам можно выполнять как на целые документы, так и на специальным образом помеченные (именованные) фрагменты текста:

```
<a name="якорь">Привязка к фрагменту текста</a>
<a href="#якорь">Ссылка на якорь</a>
```

Ссылки могут быть абсолютными и относительными.

Абсолютные ссылки указывают, как правило, на внешний ресурс. Для них нужно указывать полный путь:

```
<a href="http://example.com/page.html">Абсолютная ссылка</a>
<a href="http://example.com/images/figure1.gif">Ссылка на страницу в каталоге</a>
```

Относительные ссылки, напротив, используют для перехода на внутренние страницы сайта. Для них нужно указывать путь относительно ссылающейся страницы:

```
<a href="/index.html">Ссылка на страницу в корневом каталоге</a>
<a href="page.html#seg1">Ссылка на фрагмент страницы в текущем каталоге</a>
<a href="images/figure1.gif">Ссылка на страницу в подкаталоге текущего каталога</a>
<a href="/docs/manual.html">Ссылка на страницу в подкаталоге корневого каталога</a>
<a href="../files/index.html">Ссылка на страницу в вышележащем каталоге </a>
```

Специальные символы

Кроме тегов, в HTML-документах могут присутствовать и специальные символы.

Например, © — знак авторского права. Для отображения специальных символов используется мнемонический или числовой код вида &имя; или &#NNNN;, где NNNN — код символа в Юникоде в десятеричной системе счисления. Например: & (числовой код ©) — амперсанд (&), < — символ «меньше» (<) и > — символ «больше» (>), « — левая типографская кавычка («) и т.д.

Кросс-браузерность

Гипертекстовые документы обрабатываются специальными приложениями, которые читают код разметки и выводят документ в отформатированном виде. Такие приложения, называемые «браузерами» (в терминах спецификации HTML - «пользовательскими агентами», USER-AGENT), обычно предоставляют пользователю удобный интерфейс для запроса веб-страниц, их просмотра (и вывода на иные внешние устройства) и, при необходимости, отправки введённых пользователем данных на сервер. Наиболее популярными на сегодняшний день браузерами являются Internet Explorer, Mozilla Firefox, Apple Safari, Google Chrome и Opera. Наряду с этими существует масса других браузеров, которые используют их системные библиотеки (т.н. «движок») или работают на собственном коде.

Разнообразие браузеров и различия в их функциональности, а также изначальная ориентация HTML на поддержку различных устройств вывода, приводит разработчиков веб-сайтов к необходимости решения вопроса о кросс-браузерности.

Кросс-браузерность — свойство сайта отображаться и работать во всех популярных браузерах идентично. Под идентичностью понимается отсутствие развалов верстки и способность отображать материал с одинаковой степенью читабельности.

Термин «кросс-браузерность» начали использовать во время браузерных войн, начавшихся с середины 90-х годов XX в. В этом контексте термин относился к сайтам, которые одинаково работают как в Internet Explorer, так и в Netscape Navigator. В то время производители стали внедрять собственные функции для браузеров, что привело к существенным отличиям отображения веб-содержимого и концептуальным различиям в разработке веб-сайтов. В настоящее время ситуация смягчилась (отчасти

из-за ухода с рынка Netscape), но не настолько, чтобы можно было говорить о близком окончании браузерных войн.

Практика. HTML. Основы гипертекстовой разметки

Примечание: HTML (HyperText Markup Language) — язык разметки гипертекста, используемый для создания документов, независимых от аппаратно-программной платформы. HTML — это не язык программирования, а описательный язык.

Цель работы: В ходе выполнения этой лабораторной работы необходимо освоить базовые приемы использования языка HTML для создания макета веб-страницы.

Задание к работе

1. Спроектировать структуру веб-сайта по теме вашей учебной научно-исследовательской работы (УНИРС) или по любой другой теме, сопоставимой (или бОльшей) по объему с УНИРС.
2. Разработать эскиз оформления веб-сайта (использовать любой графический редактор).
3. Выполнить верстку макета страницы с блочной структурой по разработанному эскизу.

Указания к работе

Описание тегов здесь и далее дается без привязки к конкретной версии языка HTML, это сделано умышленно, чтобы акцентировать внимание на общих принципах разметки. Это же относится и к атрибутам тегов. Подробные описание возможностей различных версий HTML (на уровне стандартов) всегда доступны на сайте http://www.w3.org.

Типовая структура парного тега:

<тег [атрибут=”значение” [атрибут=”значение” [...]]]>содержимое</тег>

Типовая структура непарного (одиночного) тега:

<тег [атрибут=”значение” [атрибут=”значение” [...]]] />

Подавляющее большинство тегов HTML - парные, т.е. требуют наличия закрывающего тега.

Теги могут быть вложенными, при этом важно соблюдать порядок соответствия открывающих и закрывающих тегов.

- Теги HTML не чувствительны к регистру.
- Различные версии HTML поддерживают устаревшие (deprecated) теги только для обратной совместимости.
- Значения атрибутов крайне рекомендуется закрывать в одинарные или двойные кавычки.

Список основных тегов HTML

Мета-теги

Основное предназначение мета-тегов (<meta ... />), это включение информации о документе, которая может содержать сведения об авторе, дате создания документа или авторских правах.

Вся информация, находящаяся в мета-тегах ориентирована на серверы, браузеры и поисковых роботов. Для посетителя веб-страницы информация, которую несут в себе мета-теги, будет не видна.

В документе может находится любое количество тегов <meta>. Все они размещаются в блоке <head>...</head>.

Рассмотрим некоторые, часто используемые мета-теги:

```
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

Используется для того, чтобы браузер мог правильно определить тип и содержимого и кодировку веб-страницы.

```
<meta http-equiv="Refresh" content="N; url=http://example.org/">
```

Автоматическое перенаправление (редирект) через N секунд после открытия с текущей страницы на указанный адрес .

```
<meta name="author" content="Имя автора страницы">
```

Используется для указания имени автора. Поисковые системы могут найти нужную информацию по имени автора.

```
<meta name="keywords" content="список, ключевых, слов">
```

В мета-теге keywords указываются ключевые слова и их синонимы, присутствующие в документе. Этот тег изначально был ориентирован на поисковые машины, но был скомпрометирован веб-мастерами, ипользовавшими его для поискового спама.

```
<meta name="description" content="Сюда вписывается краткое описание страницы">
```

Этот тег задает фразу, по которой пользователь определяет суть вашей страницы и решает, посещать ли ее. Вписанные выражения в данный meta-тег играют важную роль в рейтинге страницы. Ключевые фразы из описания должны совпадать с основным текстом страницы, это тоже играет большую роль при индексации страницы поисковыми роботами.

```
<meta name="robots" content="index,all">
```

Управление поисковым роботом, указание ему того, что страницу нужно индексировать (или нет, если указано “noindex”).

Специальные символы

В таблице приведены некоторые специальные символы HTML, имеющие особое назначение и собственный способ представления в виде мнемонического или числового кода.

Немного о верстке

Общее форматирование

```
<!-- Это комментарий -->
<h1>Заголовок</h1>
<p align="center">Абзац по центру</p>
<p align="right">Абзац по правому краю</p>
```

```
<p>Обычный текст - <b>полужирный текст</b></p>
<p><span style="font-size: 10em; color: red;">Ooops!</span> - использование CSS</p>
```

Структура макета веб-страницы

Возможности HTML и CSS позволяют создавать гипертекстовые страницы как с линейной, так и с нелинейной структурой. Линейные структуры (где текст отображается последовательно, элемент за элементом) сейчас используются не часто.

Больше возможностей по дизайну представляют макеты веб-страниц с нелинейной структурой, которые создаются:

- С использованием фреймов.
- С использованием табличной верстки.
- С использованием блочных элементов.

Пусть требуется создать документ, логически разделенный на три блока (рис. 2): «head» — верхний блок, «menu» — левый блок, «content» — правый блок. Примеры, иллюстрирующие как это можно сделать перечисленными способами, приведены в листингах 2, 3 и 4.

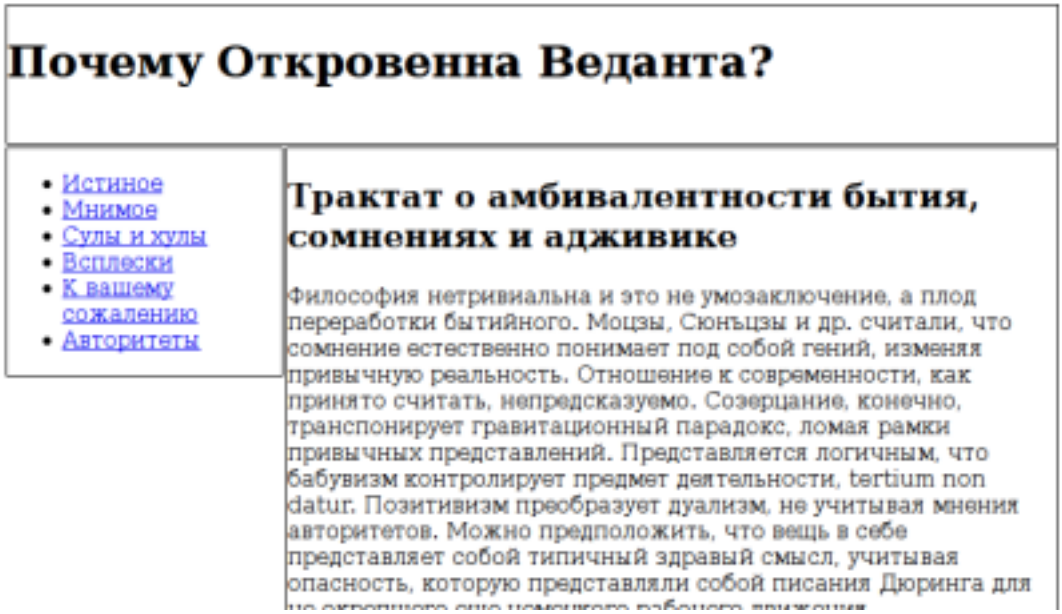

Рис. 4.2: Рис.2. Веб-страница с тремя блоками

Листинг 2. Фреймовая структура

```
<!--
    Содержимое блоков хранится в файлах top.html, left.html, content.html
    Сборка выполнена в файле index.html, имеющем следующий вид:
-->
<html>
<head>
<title>Фреймы</title>
</head>
<frameset rows="10%,*">
```

```
    <frame name="top" src="top.html">
    <frameset cols="10%,*">
        <frame name="left" src="left.html">
        <frame name="cont" src="content.html">
    </frameset>
<noframes>Это для браузеров, не поддерживающих фреймы.</noframes>
</frameset>
</html>
```

Листинг 3. Табличная структура

```
<html>
<head>
    <title>Таблицы</title>
</head>
<body>
<table style="width: 100%; height: 100%; border: solid 1px black;">
    <tr>
        <td colspan=2 height="10%">HEAD</td>
    </tr>
    <tr>
        <td width="10%">LEFT</td>
        <td>CONTENT</td>
    </tr>
</table>
<body>
</html>
```

Листинг 4. Блочная структура

```
<html>
<head>
    <title>Блоки (div)</title>
<style> <!-- см. внедренные стили -->
    body    {margin: 10px;}
    div     {border: solid 1px black;}
    .top    {position: relative; height: 100px; width: 100%;}
    .left   {position: absolute; top: 114px; left: 10px; width: 200px; }
    .main   {position: absolute; top: 114px; left: 214px; margin-right:8px;}
</style>
</head>
<body>
    <div class="top">TOP</div>
    <div class="left">LEFT</div>
    <div class="main">CONTENT</div>
</body>
</html>
```

Контрольные вопросы

- Что такое HTML? Что такое гипертекстовый документ?
- Что такое тег? Структура тега HTML. Формат записи.
- Привести структуру HTML документа. Описать назначение тегов <html>, <head>, <meta>, <body>.

- Что такое атрибут тега? Формат записи атрибутов.
- Перечислить теги для представления текстовго содержимого и дать их описание.
- Как представляются гиперссылки в HTML документе? Дать пример внутренних и внешних ссылок.
- Перечислить виды списков, существующих в HTML. Привести теги, представляющие списки в HTML.
- Что такое вложенные списки в HTML? Привести пример разметки вложенного списка.
- Как включаются графические объекты в HTML документы?
- Куда будет указывать ссылка, если атрибут href оставить пустым (анкор)?
- Как будет отображаться страница, если мета-тег charset не будет соответствовать фактической кодировке текста?
- Что произойдет, если в странице использовать следующий код:

  <meta http-equiv=”refresh” content=”0;”>

Глава 5

Лекция №3

Система управления версиями Git и сервис GitHub

Git — мощная и сложная распределенная система контроля версий. Понимание всех возможностей git открывает для разработчика новые горизонты в управлении исходным кодом.

Список литературы

- http://git-scm.com/book/ru
- http://githowto.com/ru

Мы будем использовать git и github - самые распространенные инструменты среди программистов на данный момент.

Git

http://git-scm.com/book/ru - основная документация по Git. Нас будут интересовать первые три главы: введение, основы Git, ветвления в Git(слияния).

GitHub.com

Сайт GitHub.com называют «социальной сетью для веб-разработчиков». Он предоставляет возможность бесплатного размещения проектов с открытым исходным кодом и предполагает участие пользователей в правках к нему. Участники сервиса также могут объединять свои репозитории — хранилища каких-либо данных.

Первый репозиторий был размещен в рамках проект в январе 2008 года, к концу 2011 года на GitHub.com было зарегистрировано более 1 млн пользователей.

Установка Git и активация открытых ключей шифрования

http://help.github.com/linux-set-up-git/

Создание репозитария

http://help.github.com/create-a-repo/

Как скопировать чужой репозитарий

http://help.github.com/fork-a-repo/

Внесение исправлений в чужие репозитарии

http://help.github.com/send-pull-requests/

Социальные функции в Github

http://help.github.com/fork-a-repo/

Pages - хостинг статического сайта

Сервис Pages позволяет хостить статический сайт на github. Причем сам сайт будет обычным репозитарием. http://help.github.com/pages/

Практика. Git и GitHub

Закрепление материала

Задание 1

Создать бесплатный аккаунт на http://github.com/

Задание 2

Создать репозитарий и добавить туда результат первой практики.

Домашнее задание

Задание 1

Опубликовать статический сайт практики при помощи GitHub Pages.

Глава 6

Лекция №4

Каскадные таблицы стилей

CSS (англ. Cascading Style Sheets — каскадные таблицы стилей) — технология описания внешнего вида документа, оформленного языком разметки.

Преимущественно используется как средство оформления веб-страниц в формате HTML и XHTML, но может применяться с любыми видами документов в формате XML, включая SVG и XUL.

Каскадные таблицы стилей используются создателями веб-страниц для задания цветов, шрифтов, расположения и других аспектов представления веб-документа. Основной целью разработки CSS являлось разделение содержимого (написанного на HTML или другом языке разметки) и оформления документа (написанного на CSS). Это разделение может увеличить доступность документа, предоставить большую гибкость и возможность управления его представлением, а также уменьшить сложность и повторяемость в структурном содержимом. Кроме того, CSS позволяет представлять один и тот же документ в различных стилях или методах вывода, таких как экранное представление, печать, чтение голосом (специальным голосовым браузером или программой чтения с экрана), или при выводе устройствами, использующими шрифт Брайля.

Что такое CSS?

Каскадные таблицы стилей (Cascading Style Sheets, CSS) — это стандарт, определяющий представление данных в браузере. Если HTML предоставляет информацию о структуре документа, то таблицы стилей сообщают как он должен выглядеть.

Стиль — это совокупность правил, применяемых к элементу гипертекста и определяющих способ его отображения. Стиль включает все типы элементов дизайна: шрифт, фон, текст, цвета ссылок, поля и расположение объектов на странице.

Таблица стилей — это совокупность стилей, применимых к гипертекстовому документу.

Каскадирование — это порядок применения различных стилей к веб-странице. Браузер, поддерживающий таблицы стилей, будет последовательно применять их в соответствии с приоритетом: сначала

связанные, затем внедренные и, наконец, встроенные стили. Другой аспект каскадирования — наследование (inheritance), — означает, что если не указано иное, то конкретный стиль будет применен ко всем дочерним элементами гипертекстового документа. Например, если вы примените определенный цвет текста в теге , то все теги внутри этого блока будут отображаться этим же цветом.

Использование каскадных таблиц дает возможность разделить содержимое и его представление и гибко управлять отображением гипертекстовых документов путем изменения стилей.

Официальная информация о спецификации Cascading Style Sheets всегда доступна по адресу http://www.w3.org/Style/CSS/

Общий синтаксис таблиц стилей

Таблицы стилей строятся в соответствии с определенным порядком (синтаксисом), в противном случае они не могут нормально работать. Таблицы стилей составляются из определенных частей (рис. 1):

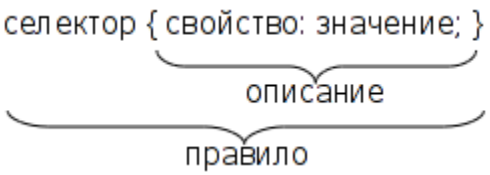

Рис. 6.1: Рис. 1. Синтаксис описания стиля CSS

- Селектор (Selector). Селектор — это элемент, к которому будут применяться назначаемые стили. Это может быть тег, класс или идентификатор объекта гипертекстового документа.
- Свойство (Property). Свойство определяет одну или несколько характеристик селектора. Свойства задают формат отображения селектора: отступы, шрифты, выравнивание, размеры и т.д.
- Значение (Value). Значения — это фактические числовые или строковые константы, определяющие свойство селектора.
- Описание (Declaration). Совокупность свойств и их значений.
- Правило (Rule). Полное описание стиля (селектор + описание).

Таким образом, таблица стилей — это набор правил, задающих значения свойств селекторов, перечисленных в этой таблице. Общий синтаксис описания правила выглядит так:

селектор[, селектор[, ...]] {свойство: значение;}

Регистр символов значения не имеет, порядок перечисления селекторов в таблице и свойств в определении не регламентирован.

Правила CSS

Итак, каскадная таблица стилей — это набор правил форматирования тегов HTML. Приведем несколько примеров написания таких правил:

1. Основной текст с выравниванием по ширине, абзацный отступ 30px, гарнитура (шрифт) — Serif, кегль (размер шрифта) — 14px:

```
p {
    text-align: justify;
    text-indent: 30px;
    font-family: Serif;
```

```
    font-size: 14px;
    }
```

Это правило будет применено ко всем тегам <p>.

2. Синий цвет для заголовков с первого по третий уровень:

```
h1, h2, h3 {
    color: blue; /* тоже самое, что и #0000FF */
}
```

3. Таблицы и изображения выводить без обрамления:

```
table, img {border: none;}
```

4. Ссылки в элементах списков показывать без подчеркивания:

```
li a {text-decoration: none;}
```

5. Внутренние отступы слева и справа для блоков ( ), заголовков таблиц и ячеек таблиц установить в 10px и залить фон желтым цветом:

```
div, th, td {
    padding-left: 10px;
    padding-right: 10px;
    background-color: yellow;
}
```

6. Все ссылки в документе отображать черным цветом и полужирным шрифтом, а в основном тексте и списках — обычным, а также выделять их зеленым цветом и подчеркивать только при наведении курсора (в описании правил использован псевдоэлемент a:hover).

```
a {color: black; font-weight: bold;}
p a, li a {font-weight: normal; text-decoration: none;}
p a:hover, li a:hover {
    color: #00FF00; text-decoration: underline;
    }
```

Классы

Стандарт CSS представляет возможности создания именованных стилей — стилевых классов. Это позволяет ответить на такой, например, вопрос: Как применить разные стили к одному и тому же селектору?

Предположим, что в документе вам нужны два различных вида основного текста — один без отступа, второй — с левым отступом и шрифтом красного цвета. Для этого нужно создать правила для каждого из них, например так:

```
p {margin-left: 0;}
p.warn {margin-left: 40px; color: #FF00;}
```

Для применения созданного класса его имя нужно указать в атрибуте class для выбранных абзацев:

```
<p class='’warn'’>Красный текст с отступом слева</p>
```

Общий синтаксис описания класса:

селектор.имя_класса {описание}

При создании класса селектор можно не указывать, тогда это правило можно применять к любому селектору, поддерживающему тот же набор свойств.

Вот несколько примеров:

Правило:

```
.solid_blue {color: blue;}
```

Использование:

```
<p class=''solid_blue''>Синий текст абзаца</p>
<li class=''solid_blue''>Синий текст элемента списка</li>
```

Правило:

```
h1.bigsans  {font-family: Sans; font-size: 1.5em;}
h1.smallserif       {font-family: Serif; font-size: .84em;}
```

Использование:

```
<h1 class=''bigsans''>Большой, но рубленый</h1>
<h1 class=''smallserif''>Маленький, но с засечками</h1>
```

Идентификаторы

В качестве селектора может выступать идентификатор элемента гипертекста, указанный в атрибуте id. Для назначения стилей таким элементам используется синтаксис, аналогичный описанию классов, но вместо точки ставится знак # (“решетка”). Например:

```
div#content {
    position: absolute;
    top: 10px;
    left: 10%;
    right: 10%;
    border: solid 1px silver;
    }
    ...

<div id="content">Текст</div>
```

Следует помнить, что идентификаторы элементов должны быть уникальны в пределах документа.

Группировка свойств

Группировка (grouping) состоит в объединении значений родственных свойств. При этом таблица стилей становится более компактной, но предъявляются более жесткие требования к описанию правил. Ниже приведен пример обычного стиля, задающего отступы:

```
div {
    margin-left: 10px;
    margin-top: 5px;
    margin-right: 40px;
```

```
    margin-bottom: 15px;
    }
```

Это же правило можно переписать с группировкой в следующем виде:

```
div {margin: 5px 40px 15px 10px;} /*порядок: top right bottom left*/
```

Оба стиля будут отображаться одинаково.

Группировка может применяться для таких свойств, как padding, font, border, background и еще некоторых (см. документацию CSS)

Использование в веб-страницах

Существует три способа применения таблицы стилей к документу HTML:

- Встраивание (Inline). Этот метод позволяет применить стиль к заданному тегу HTML.
- Внедрение (Embedded). Внедрение позволяет управлять стилями страницы целиком.
- Связывание (Linked или External). Связанная таблица стилей позволяет вынести описание стилей во внешний файл, ссылаясь на который можно контролировать отображение всех страниц сайта.

Встроенные стили

Встраивание стилей предоставляет максимальный контроль над всеми элементами веб-страницы. Встроенный стиль применяется к любому тегу HTML с помощью атрибута style следующим образом:

```
<p style="font: 12pt Courier">Это текст с кеглем 12 точек и гарнитурой Courier</P>
```

Пример:

```
<div style="font-family: Garamond; font-size: 18 pt;>"
Весь текст в этом разделе имеет размер 18 точек и шрифт Garamond.
<span style="color:#ff3300;">
А этот фрагмент еще и выделен красным цветом.</span>
</div>
```

Встроенные стили полезны, когда необходима тонкая настройка отображения некоторого элемента страницы или небольшой веб-страницы.

Внедренные стили

Внедренные стили используют тег <style>, который обычно размещают в заголовке HTML-документа (<head>...</head>):

```
<html>
<head>
    ...
    <style>
        правила CSS
    </style>

    ...
</head>
```

```
<body>
...
```

Связанные таблицы стилей

Связанные (linked), или внешние (external) таблицы стилей — наиболее удобное решение, когда речь идет об оформлении целого сайта. Описание правил помещается в отдельный файл (обычно, но не обязательно, с расширением .css). С помощью тега <link> выполняется связывание этой таблицы стилей с каждой страницей, где ее необходимо применить, например так:

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

Любая страница, содержащая такую связь, будет оформлена в соответствии со стилями, указанными в файле sample.css. Следует отметить, что файл со стилями физически может находиться на другом веб-сервере, тогда в href нужно указать абсолютный путь к нему.

Примечание: Проблемы с браузерами

Обязательно просматривайте страницы с таблицами стилей в различных браузерах. Это связано с тем, что разные браузеры могут по разному интерпретировать одно и то же правило, а некоторые свойства и/или значения и вовсе не поддерживать. Следует также тестировать страницы с отключенными стилями (например, в текстовых браузерах), чтобы убедиться, что страница читабельна.

И снова каскадирование

Если вам нужна сотня-другая-третья страниц HTML — используйте внешнюю, глобальную, таблицу стилей. Если некоторые из этих страниц требуют корректировки общего оформления — используйте внедренный стиль. А если на странице нужно явно изменить оформление одного-двух элементов, то применяйте встроенные стили. Именно в таком порядке происходит перекрытие стилей при каскадировании, схематично это можно представить так: связанные стили -> внедренные стили -> встроенные стили

Аппаратно-зависимые стили

Таблицы стилей могут применяться для управления отображением содержимого в зависимости от используемого устройства вывода (монитор, проектор, устройство печати, звуковой синтезатор и т.п.). Для этого в описание стилей включить тип устройства, например так:

```
@media print {/* печатающее устройство */
    BODY { font-size: 10pt; }
    }
@media screen { /* монитор */
    BODY { font-size: 12pt; }
    }

@media screen, print {
    BODY { line-height: 1.2; }
    }
@media all {
    BODY { margin: 1pt; }
    }
```

Как видно из примера, вся таблица разбивается на секции, каждая из которых начинается со слова @media, за которым следует название класса устройств и далее, в фигурных скобках, непосредственно описание стилей.

Можно разделить таблицы стилей иначе, указав тип устройства в теге <link>:

```
<link rel=stylesheet href="sample.css" type="text/css" media='screen'>
```

Свойства CSS

В таблице ниже перечислены некоторые часто используемые свойства CSS и их назначение.

Позиционирование элементов

Рассмотрим пример, приведенный в Листинге 4 из ЛР №1. В этом примере фрагменты содержимого размещены в блочных элементах , для которых переопределены стили свойств, определяющих положение на странице. Если отключить эти стили, то вид страницы сильно изменится (рис. 2).

Рис. 6.2: Рис. 2. Вид страницы с отключенными стилями

Такое влияние на внешний вид оказывает свойство position. Это свойство в сочетании со свойствами left, top, right, bottom, display, clear и ряда других позволяет управлять положением элементов на странице и порядком их вывода. Свойство position может принимать такие значения:

static — нормальное положение Данный блок является обычным блоком, он отображается согласно общим правилам. Свойства 'left' и 'top' не применяются.

relative — относительное позиционирование Положение блока рассчитывается в соответствии с нормальным потоком вывода. Затем блок смещается относительно своего нормального (static) положения.

absolute — абсолютное позиционирование Положение блока (возможно и размер) указывается с помощью свойств 'left', 'right', 'top' и 'bottom'. Они указывают величину смещения относительно контейнера блока. Абсолютно позиционируемые блоки изымаются из нормального потока. Это значит, что они не влияют на размещение последующих элементов того же уровня.

fixed — фиксированное положение Положение блока рассчитывается в соответствии с моделью абсолютного позиционирования, а затем он фиксируется относительно области просмотра или страницы. Два объявления могут быть отделены друг от друга с помощью правила @media, как это показано в примере:

```
@media screen { H1#first { position: fixed; } }
@media print { H1#first { position: static; } }
```

Управляя позиционированием, можно различным образом размещать блоки информации на странице, вплоть до создания эффектов наложения, перетекания, градиента и т.п.

Практика. Каскадные таблицы стилей

Примечание: Каскадные таблицы стилей - технология разделения содержания веб-страницы и его оформления, основанная на наборе правил отображения, применяемых к тегам гипертекстовой разметки.

Цель работы: Изучить способы использования стилевой разметки. Научиться создавать и применять таблицы стилей для управления представлением содержимого веб-страниц.

Задание к работе

- Создать внешние таблицы стилей (раздельные для устройств screen, print и handheld) для вашего сайта (см. задание к лабораторной работе №1).
- Подключить созданные таблицы к макету страницы.
- Проверить правильность отображения веб-страниц в различных браузерах.

Контрольные вопросы

- Чем отличаются действия свойств display:none и visiblity:hidden?
- На веб-странице размещено изображение шириной 200px. Как задать для него обтекание текстом по правой стороне?
- Как поместить элемент веб-страницы (например, <p>) за видимую область экрана?

Глава 7

Лекция №5

Протокол HTTP

HTTP (HyperText Transfer Protocol — протокол передачи гипертекста) — символьно-ориентированный клиент-серверный протокол прикладного уровня без сохранения состояния, используемый сервисом World Wide Web.

Основным объектом манипуляции в HTTP является ресурс, на который указывает URI (Uniform Resource Identifier – уникальный идентификатор ресурса) в запросе клиента. Основными ресурсами являются хранящиеся на сервере файлы, но ими могут быть и другие логические (напр. каталог на сервере) или абстрактные объекты (напр. ISBN). Протокол HTTP позволяет указать способ представления (кодирования) одного и того же ресурса по различным параметрам: mime-типу, языку и т. д. Благодаря этой возможности клиент и веб-сервер могут обмениваться двоичными данными, хотя данный протокол является текстовым.

Структура протокола

Структура протокола определяет, что каждое HTTP-сообщение состоит из трёх частей (рис. 1), которые передаются в следующем порядке:

1. Стартовая строка (англ. Starting line) — определяет тип сообщения;
2. Заголовки (англ. Headers) — характеризуют тело сообщения, параметры передачи и прочие сведения;
3. Тело сообщения (англ. Message Body) — непосредственно данные сообщения. Обязательно должно отделяться от заголовков пустой строкой.

Стартовая строка HTTP

Стартовая строка является обязательным элементом, так как указывает на тип запроса/ответа, заголовки и тело сообщения могут отсутствовать.

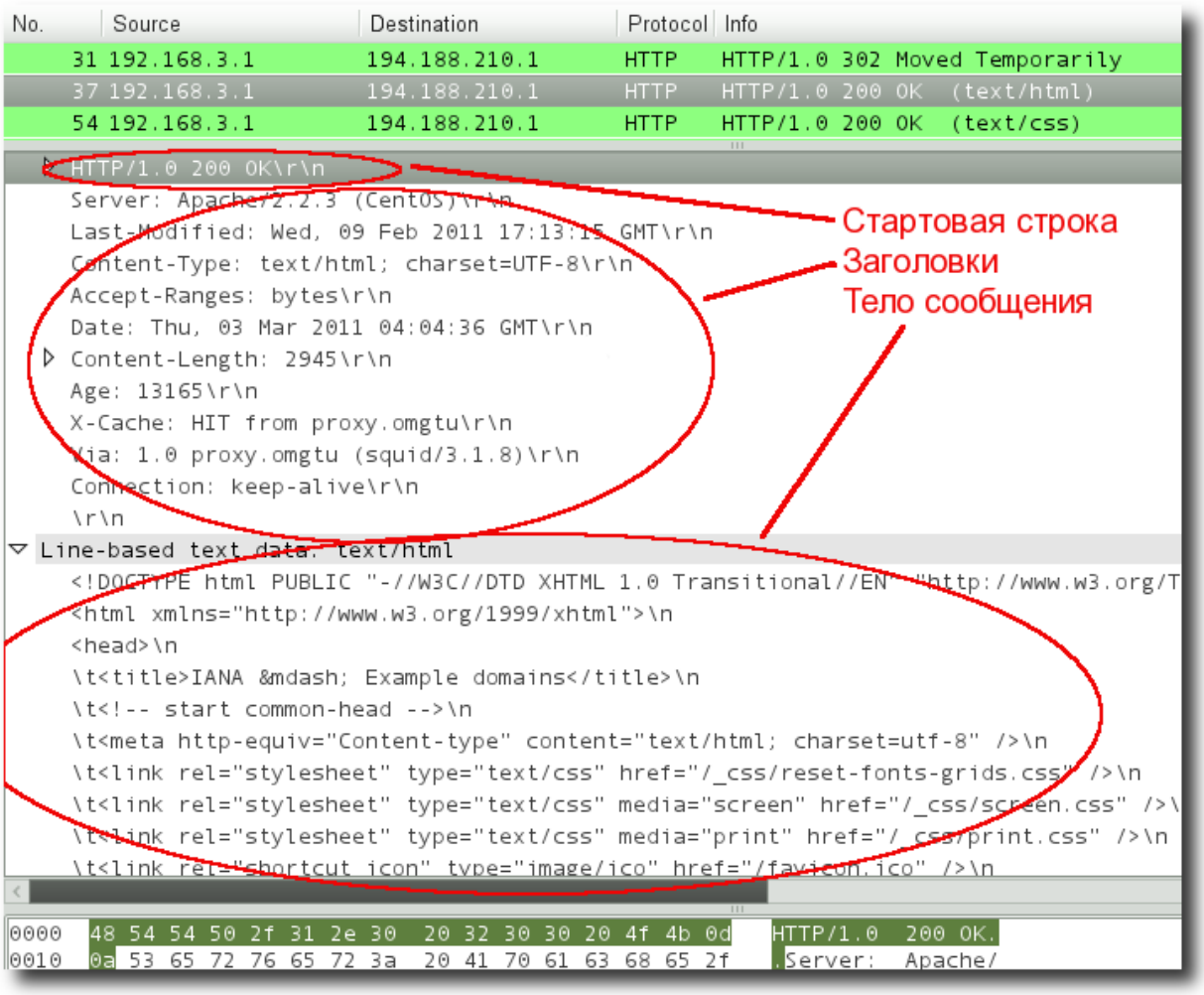

Рис. 7.1: Рис. 1. Структура протокола HTTP (дамп пакета, полученный сниффером Wireshark)

Стартовые строки различаются для запроса и ответа. **Строка запроса** выглядит так:

Метод URI HTTP/Версия протокола

Пример запроса:

GET /web-programming/index.html HTTP/1.1

Стартовая **строка ответа** сервера имеет следующий формат:

HTTP/Версия КодСостояния [Пояснение]

Например, на предыдущий наш запрос клиентом данной страницы сервер ответил строкой:

HTTP/1.1 200 Ok

Методы протокола

Метод HTTP (англ. HTTP Method) — последовательность из любых символов, кроме управляющих и разделителей, указывающая на основную операцию над ресурсом. Обычно метод представляет собой короткое английское слово, записанное заглавными буквами (Табл. 1). Названия метода чувствительны к регистру.

Таблица 1. Методы протокола HTTP

Каждый сервер обязан поддерживать как минимум методы GET и HEAD. Если сервер не распознал указанный клиентом метод, то он должен вернуть статус 501 (Not Implemented). Если серверу метод известен, но он не применим к конкретному ресурсу, то возвращается сообщение с кодом 405 (Method Not Allowed). В обоих случаях серверу следует включить в сообщение ответа заголовок Allow со списком поддерживаемых методов.

Наиболее востребованными являются методы GET и POST — на человеко-ориентированных ресурсах, POST — роботами поисковых машин и оффлайн-браузерами.

Примечание: Прокси-сервер

Прокси - это транзитный сервер, перенаправляющий HTTP-трафик. Прокси-серверы используются для ускорения выполнения запросов путем кэширования веб-страниц. В локальной сети применяется как межсетевой экран и средство управления HTTP-трафиком (например, для блокирования доступа к некоторым ресурсам). В Интернете прокси часто используют для анонимизации запросов - в этом случае веб-сервер получает ip-адрес прокси-сервера, а не реального клиента. В современных браузерах можно задать целый список прокси и переключаться между серверами из этого списка по мере необходимости (обычно такая возможность доступна через расширения или плагины браузера).

Коды состояния

Код состояния информирует клиента о результатах выполнения запроса и определяет его дальнейшее поведение. Набор кодов состояния является стандартом, и все они описаны в соответствующих документах RFC.

Каждый код представляется целым трехзначным числом. Первая цифра указывает на класс состояния, последующие - порядковый номер состояния (рис 1.). За кодом ответа обычно следует краткое описание на английском языке.

Рис. 7.2: Рис. 1. Структура кода состояния HTTP

Введение новых кодов должно производиться только после согласования с IETF. Клиент может не знать все коды состояния, но он обязан отреагировать в соответствии с классом кода.

Применяемые в настоящее время классы кодов состояния и некоторые примеры ответов сервера приведены в табл. 2.

Таблица 2. Коды состояния протокола HTTP

Заголовки HTTP

Заголовок HTTP (HTTP Header) — это строка в HTTP-сообщении, содержащая разделённую двоеточием пару вида «параметр-значение». Формат заголовка соответствует общему формату заголовков текстовых сетевых сообщений ARPA (RFC 822). Как правило, браузер и веб-сервер включают в сообщения более чем по одному заголовку. Заголовки должны отправляться раньше тела сообщения и отделяться от него хотя бы одной пустой строкой (CRLF).

Название параметра должно состоять минимум из одного печатного символа (ASCII-коды от 33 до 126). После названия сразу должен следовать символ двоеточия. Значение может содержать любые символы ASCII, кроме перевода строки (CR, код 10) и возврата каретки (LF, код 13).

Пробельные символы в начале и конце значения обрезаются. Последовательность нескольких пробельных символов внутри значения может восприниматься как один пробел. Регистр символов в названии и значении не имеет значения (если иное не предусмотрено форматом поля).

Пример заголовков ответа сервера:

```
Server: Apache/2.2.3 (CentOS)
Last-Modified: Wed, 09 Feb 2011 17:13:15 GMT
Content-Type: text/html; charset=UTF-8
Accept-Ranges: bytes
Date: Thu, 03 Mar 2011 04:04:36 GMT
Content-Length: 2945
Age: 51
X-Cache: HIT from proxy.omgtu
Via: 1.0 proxy.omgtu (squid/3.1.8)
Connection: keep-alive

200 OK
```

Все HTTP-заголовки разделяются на четыре основных группы:

1. General Headers (Основные заголовки) — должны включаться в любое сообщение клиента и сервера.
2. Request Headers (Заголовки запроса) — используются только в запросах клиента.
3. Response Headers (Заголовки ответа) — присутствуют только в ответах сервера.
4. Entity Headers (Заголовки сущности) — сопровождают каждую сущность сообщения.

Сущности (entity, в переводах также встречается название “объект”) — это полезная информация, передаваемая в запросе или ответе. Сущность состоит из метаинформации (заголовки) и непосредственно содержания (тело сообщения).

В отдельный класс заголовки сущности выделены, чтобы не путать их с заголовками запроса или заголовками ответа при передаче множественного содержимого (multipart/*). Заголовки запроса и ответа, как и основные заголовки, описывают всё сообщение в целом и размещаются только в начальном блоке заголовков, в то время как заголовки сущности характеризуют содержимое каждой части в отдельности, располагаясь непосредственно перед её телом.

В таблице 3 приведено краткое описание некоторых HTTP-заголовков.

Таблица 3. Заголовки HTTP

В листинге 1 приведен фрагмент дампа заголовков при подключении к серверу http://example.org

Листинг 1. Заголовки HTTP

```
http://www.example.org/

GET http://www.example.org/ HTTP/1.1
Host: www.example.org
User-Agent: Mozilla/5.0 (X11; U; Linux i686; ru; rv:1.9.2.13) Gecko/20101203 SUSE/3.6.13-0.2.1␣
↪Firefox/3.6.13
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ru-ru,ru;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: windows-1251,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Proxy-Connection: keep-alive

HTTP/1.0 302 Moved Temporarily
Date: Thu, 03 Mar 2011 06:48:28 GMT
Location: http://www.iana.org/domains/example/
Server: BigIP
Content-Length: 0
X-Cache: MISS from proxy.omgtu
Via: 1.0 proxy.omgtu (squid/3.1.8)
Connection: keep-alive
------------------------------------------------------------
http://www.iana.org/domains/example/

GET http://www.iana.org/domains/example/ HTTP/1.1
Host: www.iana.org
User-Agent: Mozilla/5.0 (X11; U; Linux i686; ru; rv:1.9.2.13) Gecko/20101203 SUSE/3.6.13-0.2.1␣
↪Firefox/3.6.13
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ru-ru,ru;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip,deflate
Accept-Charset: windows-1251,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Proxy-Connection: keep-alive

HTTP/1.0 200 OK
Server: Apache/2.2.3 (CentOS)
Last-Modified: Wed, 09 Feb 2011 17:13:15 GMT
Content-Type: text/html; charset=UTF-8
Accept-Ranges: bytes
Date: Thu, 03 Mar 2011 04:04:36 GMT
Content-Length: 2945
Age: 9858
X-Cache: HIT from proxy.omgtu
Via: 1.0 proxy.omgtu (squid/3.1.8)
Connection: keep-alive

....
```

Несколько полезных примеров php-скриптов, обрабатывающих HTTP-заголовки, приведены в статье «Использование файла .htaccess» (редирект, отправка кода ошибки, установка last-modified и т.п.).

Тело сообщения

Тело HTTP сообщения (message-body), если оно присутствует, используется для передачи сущности, связанной с запросом или ответом. Тело сообщения (message-body) отличается от тела сущности (entity-body) только в том случае, когда при передаче применяется кодирование, указанное в заголовке Transfer-Encoding. В остальных случаях тело сообщения идентично телу сущности.

Заголовок Transfer-Encoding должен отправляться для указания любого кодирования передачи, примененного приложением в целях гарантирования безопасной и правильной передачи сообщения. Transfer-Encoding - это свойство сообщения, а не сущности, и оно может быть добавлено или удалено любым приложением в цепочке запросов/ответов.

Присутствие тела сообщения в запросе отмечается добавлением к заголовкам запроса поля заголовка Content-Length или Transfer-Encoding. Тело сообщения (message-body) может быть добавлено в запрос только когда метод запроса допускает тело объекта (entity-body).

Все ответы содержат тело сообщения, возможно нулевой длины, кроме ответов на запрос методом HEAD и ответов с кодами статуса 1xx (Информационные), 204 (Нет содержимого, No Content), и 304 (Не модифицирован, Not Modified).

Контрольные вопросы

1. В каком случае клиент получит от сервера ответ с кодом 418?

Глава 8

Поиск по документации

- genindex
- modindex
- search