ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
СЕВЕРО-КАВКАЗСКИЙ ФИЛИАЛ
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО ОБРАЗОВАТЕЛЬНОГО БЮДЖЕТНОГО
УЧРЕЖДЕНИЯ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
МОСКОВСКОГО ТЕХНИЧЕСКОГО УНИВЕРСИТЕТА СВЯЗИ И ИНФОРМАТИКИ

Ткачук Е.О.

WEB-программирование

Методическое пособие по выполнению практических занятий

Ростов-на-Дону
2019

УДК 004.75:004:425.2

Ткачук Е.О.

WEB-программирование. Методическое пособие по выполнению практических занятий. / Моск. техн. ун-т связи и информатики, Сев.-Кавк. филиал. – Ростов н/Д, 2019, 68 с.

В пособии даются организационно-методические указания к лабораторным вопросам и порядок выполнения и оформления практических занятий.

Предназначено для студентов всех специальностей обеих форм обучения, изучающих дисциплину «WEB-программирование», а также может быть полезно всем остальным студентам, желающим самостоятельно современные методы WEB-программирование.

Рецензент канд. техн. наук, доц. А.Н. Чикалов (СКФ МТУСИ)

Обсуждено и утверждено на заседании кафедры СПОИ (протокол заседания кафедры №1 от 26.08.2019).

© Московский технический университет связи и информатики, Северо-Кавказский филиал, 2019

СОДЕРЖАНИЕ

Практическое занятие № 1. Организация совместного использования линий связи

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Адресация в компьютерных сетях бывает двух видов: физическая адресация (на основе MAC-адреса) и логическая (на основе IP-адреса).

MAC-адрес — это уникальный идентификатор, сопоставляемый с различными типами оборудования для компьютерных сетей. В широковещательных сетяхMAC-адрес позволяет уникально идентифицировать каждый узел сети и доставлять данные только этому узлу.

MAC адрес состоит из 6 частей (6 октетов = 48 байт) и содержит информацию, в т.ч., о производителе и типе оборудования. Обычно записывается в шестнадцатиричном виде и содержит знаки 0 - 9, A - F. Регистр символов роли не играет. Разделительные знаки (":" "-" и пр.) могут и отсутствовать, но их наличие делает число более читаемым. Эта информация вшита в оборудование, и позволяет идентифицировать сетевое устройство наподобие IP адреса. Иными словами, MAC – адрес аппаратный, IP – логический сетевой адрес.

Идентификация по MAC-адресам нужна для упрощения работы иудешевления сетевого оборудования. Однако, построить интернет на таком оборудовании невозможно, поскольку оно физически не способно хранить огромные таблицы уникальных MAC-адресов. Поэтому применяется адресация IP, в которой узлы сгруппированы в сети, и в таблицах межсетевых устройств (маршрутизаторов) хранятся именно адреса групп, что уменьшает размер таблиц.

IP-адрес - это 32-х битное двоичное число (4 октета). Обычно, для лучшей ясности, IP-адреса представляются в виде десятичных значений отдельных октетов, разделенных точками (dottedquadaddress). IP-адрес состоит из двух частей. Адрес сети (network ID) определяет, в какой логической сети находится адресованное сетевое соединение. Адрес устройства (host ID) определяет, о каком устройстве логической сети идет речь. Граница между адресом сети и адресом устройства не определена однозначно. Она зависит от класса IP-адреса и от возможного дополнительного подразделения сети на подсети (subnetworking). Четко граница между адресом сети и адресом устройства определяется маской подсети (subnetmask или networkmask). Маска подсети - это 32-битное число, имеющее непрерывную последовательность единиц на местах, относящихся к адресу сети, и последовательность нулей на местах, относящихся к адресу устройства.

Классовая и бесклассовая IP-адресация. Адресная и широковещательная рассылка в сети.

Классовая адресация IP сетей — архитектура сетевой адресации, которая использовалась в Интернете в период с 1981 по 1993 годы, до введения бесклассовой междоменной маршрутизации.

Этот метод адресации делит адресное пространство протокола Интернета версии 4 (IPv4) на пять классов адресов: A, B, C, D и E. Принадлежность адреса к конкретному классу задаётся первыми битами адреса. Каждый класс определяет либо соответствующий размер сети, то есть количество возможных адресов хостов внутри данной сети (классы A, B, C), либо сеть многоадресной передачи (класс D). Диапазон адресов пятого класса (E) был зарезервирован для будущих или экспериментальных целей.

Использование адресации на базе классов адресов в IP-сетях, в основном, прекращено: остатки классовых сетевых концепций на практике остаются лишь в ограниченном объеме в параметрах конфигурации по умолчанию некоторых сетевых программных и аппаратных компонентов (например, маска подсети по умолчанию). Применение этого метода адресации не позволяет экономно использовать ограниченный ресурс адресов IPv4, поскольку невозможно применение произвольных масок подсетей к различным подсетям.

Бесклассоваяадресация (англ. Classless Inter-Domain Routing, англ. CIDR) — метод IP-адресации, позволяющий гибко управлять пространством IP-адресов, не используя жёсткие рамки классовой адресации. Использование этого метода позволяет экономно использовать ограниченный ресурс IP-адресов, поскольку возможно применение различных масок подсетей к различным подсетям.

Бесклассовая адресация основывается на переменной длине маски подсети, в то время, как в классовой адресации длина маски строго фиксирована 0, 1, 2 или 3 установленными октетами.

Широковещательные рассылки предусмотрены во многих сетевых протоколах, например ARP и DHCP.В пакете широковещательной рассылки содержится IP-адрес назначения, в узловой части которого присутствуют только единицы. Это означает, что пакет получат и обработают все узлы в локальной сети (домене широковещательной рассылки).

В сети класса C 192.168.1.0 с маской подсети по умолчанию 255.255.255.0 используется адрес широковещательной рассылки 192.168.1.255. Узловая часть – 255 или двоичное 11111111 (все единицы).

В сети класса B 172.16.0.0 с маской подсети по умолчанию 255.255.0.0 используется адрес широковещательной рассылки 172.16.255.255.

В сети класса A 10.0.0.0 с маской подсети по умолчанию 255.0.0.0 используется адрес широковещательной рассылки 10.255.255.255.

Для сетевого IP-адреса широковещательной рассылки нужен соответствующий MAC-адрес в кадре Ethernet. В сетях, построенных на технологии Ethernet, используется

MAC-адрес широковещательной рассылки из 48 единиц, который в шестнадцатеричном формате выглядит как FF-FF-FF-FF-FF-FF.

Адрес одноадресной рассылки чаще всего встречается в сети IP.Пакет с одноадресным назначением предназначен конкретному узлу.

Пример: узел с IP-адресом 192.168.1.5 (источник) запрашивает веб-страницу с сервера с IP-адресом 192.168.1.200 (адресат).

Для отправки и приема одноадресного пакета в заголовке IP-пакета должен указываться IP-адрес назначения. Кроме того, в заголовке кадра Ethernet должен быть MAC-адрес назначения. IP-адрес и MAC-адрес - это данные для доставки пакета одному узлу. IPv4 использует 32 битное адресное пространство, которое имеет размер 4 байта. Это означает, что общее количество IP адресов в интернете может быть 2 в 32 степени.

IPv6 использует адресное пространство размером 128 бит. Поэтому общее количество адресов будет 2 в 128 степени.

Адрес IPv6 отличается от адреса IPv4. Каждая группа разделяется двоеточием вместо точки и состоит из 16 бит, в виде четырех шестнадцатеричных цифр. Первые 64 бита содержат информацию о сетевом адресе, которая используется для маршрутизации, остальные 64 содержат подробную информацию о сетевом интерфейсе хоста.

IPv4 не предоставляет встроенную возможность безопасности.IPv6 адреса снабжены встроенной поддержкой IPSec.

IPv4 не имеет возможности определения одинаковых адресов (duplicateaddressdetection (DAD)).УIPv6 есть эта встроенная защитная функция, добавленная егоразработчиками.

Заголовок пакета IPv6 не содержит лишних полей. Он использует только 8 полей, по сравнению с 13 в случае с IPv4. Дополнительные поля теперь являются необязательными расширениями заголовка. Размер заголовка 40 байт, что в два раза больше, чем у IPv4.

IPv4 использует как статическое, так и динамическое назначение IP адресов. Назначение IP адреса статически требует определенного времени работы. DHCP иBOOTP используются для динамического назначения IP адресов. DHCP и BOOTP автоматически назначают IP адреса хостам, когда они загружаются в сети.

IPv6 использует протокол автоматической конфигурации (statelessautoconfiguration) сходный с DHCP. Таким образом, любой интерфейс маршрутизатора, который имеет назначенный адрес IPv6, становится поставщиком IP адресов в сети. Использование портов, DHCP, DNS, хост-файлов.

Порт — цифровой номер, который является программным адресом, используемым для взаимодействия различных конечных точек (сетевых устройств, хостов) в современных компьютерных сетях на транспортном уровне модели OSI. Порты используются в транспортных протоколах TCP, UDP, SCTP, DCCP и позволяют различным программам и сетевым службам на одном хосте получать данные в IP-пакетах независимо друг от друга.

Всякое взаимодействие двух хостов подразумевает использование как минимум одного порта получателя, ипорта источника. Номер порта, добавленный к IP-адресу компьютера, завершает идентификацию возможного сеанса связи. То есть, пакеты данных направляются по сети к определенному IP-адресу назначения, а затем, по достижении конечного компьютера, далее направляется конкретному процессу, связанному с номером порта назначения. Принцип использования портов зависит от протокола, который их использует. Порт хоста назначения конкретного сетевого взаимодействия обычно известен приложению заранее. Порт хоста-источника сетевого пакета может назначаться как динамически для каждого нового сеанса связи, так и быть постоянным, статическим.

Примером использования протоколов может служитьсетевая служба электронной почты (E-mail). Сервер используется для отправки и получения электронной почты, то есть нуждается в существовании двух служб. Первая служба используется для передачи сообщений на другие сервера. Это выполняется посредством простого протокола передачи почты (SMTP). Приложение-служба протокола SMTP обычно прослушивает TCP-порт 25 для входящих запросов. Вторая служба это, как правило, либо PostOfficeProtocol (POP), либо InternetMessageAccessProtocol (IMAP), которые используются клиентскими приложениями электронной почты на персональных компьютерах пользователей для получения сообщений электронной почты с сервера. Служба POP прослушивает TCP-порт под номером 110. Обе службы могут быть запущены на одном и том же хост-компьютере, и в этом случае по номеру порта можно определить сервис, который запрошен удаленным компьютером, будь то компьютер пользователя или другой почтовый сервер. Номера порта-источника (то есть клиента) для подключения к серверу SMTP и POP3 обычно определяется динамически. DHCP — сетевой протокол, позволяющий компьютерам автоматически получать IP-адрес и другие параметры, необходимые для работы в сети TCP/IP. Данный протокол работает по модели «клиент-сервер». Для автоматической конфигурации компьютер-клиент на этапе конфигурации сетевого устройства обращается к серверу DHCP и получает от него нужные параметры. Сетевой администратор может задать диапазон адресов, распределяемых сервером среди компьютеров. Протокол DHCP используется в большинстве сетей TCP/IP.

DHCP сервер может предоставлять настройки, используя следующие методы:

· Выделение вручную - подразумевает использование DHCP для определения уникального аппаратного адреса каждой сетевой карты, подключенной к сети, и продолжительного предоставления неизменной конфигурации каждый раз, когда DHCP клиент делает запрос на DHCP сервер, используя это сетевое устройство.

· Динамическое выделение - DHCP сервер будет выделять IP адрес из пула адресов на период времени, который настраивается на сервере, или пока клиент не проинформирует сервер, что больше вообще не нуждается в адресе. Клиенты получают свои настройки динамически по принципу «первый пришел - первый обслужился».

· Автоматическое выделение - DHCP автоматически присваивает постоянный IP адрес устройству, выбранный из пула доступных адресов.

2. Ответьте на контрольные вопросы

Контрольные вопросы

1. Дайте определение понятию «коммуникационная сеть» (ПК-11).
2. Дайте определение понятию «информационная сеть» (ПК-11).
3. Дайте определение понятию «вычислительная (компьютерная) сеть» (ПК-11).
4. Дайте определение понятию «локальная вычислительная сеть (лвс)» (ПК-11).
5. Дайте определение понятию «корпоративная вычислительная сеть» (ПК-11).
6. Дайте определение понятию «глобальная вычислительная сеть» (ПК-11).
7. Дайте определение понятию «рабочая группа» (ПК-11).
8. Дайте определение понятию «домен (domain)» (ПК-11).
9. Дайте определение понятию «узел (host)» (ПК-11).
10.. Дайте определение понятию «топология (topology)» (ПК-11).
11.. Дайте определение понятию «сетевая архитектура» (ПК-11).
12.. Дайте определение понятию «программное обеспечение компьютерных сетей»

Практическое занятие № 2 Изучение методов адресации в компьютерных сетях

DNS (система доменных имён) — компьютерная распределённая система для получения информации о доменах. Используется для получения IP-адреса по имени хоста, получения информации о маршрутизации почты, обслуживающих узлах для протоколов в домене.

Распределённая база данных DNS поддерживается с помощью иерархии DNS-серверов, взаимодействующих по определённому протоколу.

Основой DNS является представление об иерархической структуре доменного имени и зонах. Каждый сервер, отвечающий за имя, может делегировать ответственность за дальнейшую часть домена другому серверу, что позволяет возложить ответственность за актуальность информации на серверы различных организаций, отвечающих только за «свою» часть доменного имени.

DNS важна для работы Интернета, так как для соединения с узлом необходима информация о его IP-адресе, а для людей проще запоминать буквенные (обычно осмысленные) адреса, чем последовательность цифр IP-адреса. В некоторых случаях это позволяет использовать виртуальные серверы, например, HTTP-серверы, различая их по имени запроса.

Hosts — текстовый файл, содержащий базу данных доменных имен и использующийся при их трансляции в сетевые адреса узлов. Запрос к этому файлу имеет приоритет перед обращением к DNS-серверам. В отличие от системы DNS, содержимое файла контролируется администратором компьютера.Файл hostsне имеет видимого расширения, но его можно редактировать в любом текстовом редакторе, как обычный файл текстового формата.

Его местоположение может отличаться в зависимости от операционной системы, но по умолчанию файл hostsнаходится:

Windows 95/98/ME: WINDOWS\hosts

Windows NT/2000: WINNT\system32\drivers\etc\hosts

Windows XP/2003/Vista: WINDOWS\system32\drivers\etc\hosts

В Windows NT/2000/XP/2003 это местоположение также можно изменить с помощью следующего ключа реестра:

HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters

"DataBasePath"="%SystemRoot%\System32\drivers\etc" Технология NAT.

NAT (от англ.NetworkAddressTranslation — «преобразование сетевых адресов») — это механизм в сетях TCP/IP, позволяющий преобразовывать IP-адреса транзитных

пакетов. Такжеимеетназвания IP Masquerading, Network Masquerading и Native Address Translation.

Преобразование адреса методом NAT может производиться почти любым маршрутизирующим устройством — маршрутизатором, сервером доступа, межсетевым экраном. Наиболее популярным является SNAT, суть механизма которого состоит в замене адреса источника при прохождении пакета в одну сторону и обратной замене адреса назначения в ответном пакете. Наряду с адресами источник/назначение могут также заменяться номера портов источника и назначения.

Принимая пакет от локального компьютера, роутер смотрит на IP-адрес назначения. Если это локальный адрес, то пакет пересылается другому локальному компьютеру. Если нет, то пакет надо переслать наружу в интернет. Но ведь обратным адресом в пакете указан локальный адрес компьютера, который из интернета будет недоступен. Поэтому роутер «на лету» транслирует обратный IP-адрес пакета на свой внешний (видимый из интернета) IP-адрес и меняет номер порта. Комбинацию, нужную для обратной подстановки, роутер сохраняет у себя во временной таблице. Через некоторое время после того, как клиент и сервер закончат обмениваться пакетами, роутер сотрет у себя в таблице запись о n-ом порте за сроком давности.

Помимо SNATчасто применяется также destination NAT, когда обращения извне транслируются межсетевым экраном на компьютер пользователя в локальной сети, имеющий внутренний адрес и потому недоступный извне сети непосредственно (без NAT).

Существует 3 базовых концепции трансляции адресов: статическая, динамическая, маскарадная.

NAT выполняет три важных функции.

· Позволяет сэкономить IP-адреса (в случае использования NAT в режиме PAT), транслируя несколько внутренних IP-адресов в один внешний публичный IP-адрес.

· Позволяет предотвратить или ограничить обращение снаружи к внутренним хостам, оставляя возможность обращения изнутри наружу. При инициации соединения изнутри сети создаётся трансляция. Ответные пакеты, поступающие снаружи, соответствуют созданной трансляции и поэтому пропускаются. Если для пакетов, поступающих снаружи, соответствующей трансляции не существует, они не пропускаются.

· Позволяет скрыть определённые внутренние сервисы внутренних хостов/серверов. VPN — обобщённое название технологий, позволяющих обеспечить одно или несколько сетевых соединений (логическую сеть) поверх другой сети (например, Интернет).

Несмотря на то, что коммуникации осуществляются по сетям с меньшим или неизвестным уровнем доверия (например, по публичным сетям), уровень доверия к построенной логической сети не зависит от уровня доверия к базовым сетям благодаря использованию средств криптографии (шифрования, аутентификации, инфраструктуры

открытых ключей, средств для защиты от повторов и изменений передаваемых по логической сети сообщений).

В зависимости от применяемых протоколов и назначения, VPN может обеспечивать соединения трёх видов: узел-узел, узел-сеть и сеть-сеть.

Обычно VPN развёртывают на уровнях не выше сетевого, так как применение криптографии на этих уровнях позволяет использовать в неизменном виде транспортные протоколы (такие как TCP, UDP).

Пользователи MicrosoftWindows обозначают термином VPN одну из реализаций виртуальной сети — PPTP, причём используемую зачастую не для создания частных сетей.

При должном уровне реализации и использовании специального программного обеспечения сеть VPN может обеспечить высокий уровень шифрования передаваемой информации. При правильной настройке всех компонентов технология VPN обеспечивает анонимность в Сети.

VPN состоит из двух частей: «внутренняя» (подконтрольная) сеть, которых может быть несколько, и «внешняя» сеть, по которой проходит инкапсулированное соединение (обычно используется Интернет). Возможно также подключение к виртуальной сети отдельного компьютера. Подключение удалённого пользователя к VPN производится посредством сервера доступа, который подключён как к внутренней, так и к внешней (общедоступной) сети. При подключении удалённого пользователя (либо при установке соединения с другой защищённой сетью) сервер доступа требует прохождения процесса идентификации, а затем процесса аутентификации. После успешного прохождения обоих процессов, удалённый пользователь (удаленная сеть) наделяется полномочиями для работы в сети, то есть происходит процесс авторизации.

Взаимодействие IP-адреса и маски подсети.

IP-адрес — уникальный сетевой адрес узла в компьютерной сети, построенной по протоколу IP.

В сети Интернет требуется глобальная уникальность адреса; в случае работы в локальной сети требуется уникальность адреса в пределах сети. В версии протокола IPv4 IP-адрес имеет длину 4 байта, а в версии протокола IPv6 IP-адрес имеет длину 16 байт.

Маска подсети — битовая маска, определяющая, какая часть IP-адреса узла сети относится к адресу сети, а какая — к адресу самого узла в этой сети.

Другой вариант определения — это определение подсети IP-адресов. Например, с помощью маски подсети можно сказать, что один диапазон IP-адресов будет в одной подсети, а другой диапазон соответственно в другой подсети.

Как и IP-адрес, маска состоит из 32 бит.Маска сравнивается с IP-адресом побитно, слева направо. В маске подсети единицы соответствуют сетевой части, а нули - адресу узла. Отправляя пакет, узел сравнивает маску подсети со своим IP-адресом и адресом

назначения. Если биты сетевой части совпадают, значит, узлы источника и назначения находятся в одной и той же сети, и пакет доставляется локально. Если нет, отправляющий узел передает пакет на интерфейс локального маршрутизатора для отправки в другую сеть.

В домашних офисах и небольших компаниях чаще всего встречаются следующие маски подсети: 255.0.0.0 (8 бит), 255.255.0.0 (16 бит) и 255.255.255.0 (24 бита). В маске подсети 255.255.255.0 (десятичный вариант), или 11111111.11111111.1111111.00000000 (двоичный вариант) 24 бита идентифицируют сеть, а 8 - узлы в сети.

ЗАДАНИЕ НА ПРАКТИЧЕСКОЕ ЗАНЯТИЕ

Задание1. Выясните, каков будет порядок отправки информацию по адресам 192.168.193.31 и 192.167.192.3 для хоста с адресом 192.167.12.3 и маской подсети 255.255.0.0. Решение задачи запишите в отчет.

Как происходит передача данных

1. IP-адрес в двоичном представлении разбивается на 2 части - адрес сети (левая часть адреса) и адрес хоста (правая часть адреса).

Например, в адресе 190.167.34.2 первые 24 бита могут быть адресом сети, а последние 8 – адресом хоста.

Тогда наш адрес будет выглядеть как 10111110101001110010001000000010, где зеленым цветом выделена сетевая часть адреса (она одинакова для всех хостов локальной сети), а красным - часть адреса, адресующая хост внутри локальной сети.

Для того, чтобы быстро вычислять по IP-адресу адрес сети или хоста, используется понятие **маски подсети (subnet mask)**. Это двоичное число, в котором все биты адреса сетевой части адреса равны 1, а все остальные биты равны нулю. В нашем случае для адреса 10111110101001110010001000000010 получим маску подсети 11111111111111111111111100000000.

1. Маску подсети принято записывать в том же десятичном формате, что и IP-адрес. Для этого нужно каждый байт маски перевести в десятичное число и записать полученные десятичные числа через точки. Ванашемаслучае

Ответ:

$11111111_2=255$
$11111111_2=255$
$11111111_2=255$
$00000000_2=0$

255.255.255.0 - маска подсети.

Маску подсети в настоящее время все чаще называют маской сети, что точней отображает ее смысл.

3. Информационные пакеты пересылаются напрямую от компьютера-

отправителя к компьютеру-получателю только в пределах одной сети. Если компьютер-получатель находится в другой сети, то информация пересылается специальному компьютеру сети, который называется шлюзом (gateway). Его адрес всегда известен. Об этом заботится системный администратор. Компьютер-шлюз имеет связь с как минимум с одной другой сетью и ретранслирует информацию в нужном направлении. Этот процесс называется маршрутизацией (routing).

4. Если ваш компьютер, имеющий IP адрес 192.169.204.12 и маску подсети 255.255.192.0 должен отправить информацию компьютеру с адресом 192.169.198.15, то прежде всего ваш компьютер проверит, находится ли получатель информации в той же сети. Для этого двоичное представление адреса получателя он побитовоумножит на двоичное представление маски подсети, то в результате получится адрес сети:

```
11000000101010001100011000001100 (адрес компьютера - получателя)
*
11111111111111111100000000000000 (текущая маска подсети)
--------------------------------------------------
11000000101010001100000000000000 (адрес сети получателя)
```

Аналогичную процедуру компьютер проделает со своим адресом для того, чтобы узнать адрес своей собственной сети:

```
11000000101010011100110000001100 (адрес компьютера - отправителя)
*
11111111111111111100000000000000 (текущая маска подсети)
--------------------------------------------------
11000000101010011100000000000000 (адрес своей собственной сети)
```

Если адрес сети получателя совпадает с адресом собственной сети, следовательно, получатель находится в локальной сети, и информация может быть послана напрямую. Если бы совпадения нет, то информация будет отправлена шлюзу (с адресом, например,192.168.192.2) с указанием адреса получателя 192.169.204.12, а он переслал бы ее в другую сеть. Этот процесс будет продолжаться до тех пор, пока информация не дойдет до получателя.

Задание2. Определение настроек протокола IP вашего компьютера Для этого достаточно запустить программу ipconfig (в Windows 9X есть еще программа с графическим интерфейсом winipcfg). Получить доступ к командной строке и напечатайте ipconfig. Нажмите клавишу :

```
C:\>ipconfig

Настройка протокола IP для Windows

        DNS-суффикс этого подключения . . :
        IP-адрес  . . . . . . . . . . . . : 192.168.0.1
        Маска подсети . . . . . . . . . . : 255.255.255.0
        Основной шлюз . . . . . . . . . . : 192.168.0.10

C:\>_
```

Занесите полученные параметры в отчет.

Примечание. Настройка протокола IP на каждом компьютере локальной сети - одна из задач системного администратора. Он может в принципе задать все параметры вручную. Но если число компьютеров в сети больше десятка, то удобней назначать настройки автоматически в момент загрузки компьютера. Для этого разработан специальный протокол DHCP (Dynamic Host Configuration Protocol). Наличие у компьютера правильного IP-адреса является совершенно необходимым условием его работы в Интернет.

Задание 3. По IP -адресу определить идентификатор сети и идентификатор узла (подразумеваются стандартные классы IP-адресов):

192.168.1.1
126.15.25.5
221.186.52.65
125.14.7.8

Контрольные вопросы

1. Протокол TCP/IP.
2. IP-адреса.
3. Статический IP-адрес.
4. Автоматическое получение IP-адреса.
5. Управляющие протоколы Интернета.
6. Тестирование TCP/IP.
7. Маршрутизация пакетов в IP сетях.
8. Утилиты командной строки для работы с сетью.
9. Служба имен доменов.
10. Пространство имен домена.
11. Разрешение имени.
12. Прямой и обратный запросы.

Практическое занятие № 3 Вёрстка web страниц с применением html5 и CSS3

Практические работы по HTML

Задание № 1. *Создание простейшего файла HTML*

1. Создайте личную папку, куда вы будете сохранять все файлы своего сайта.
2. Запустите программу *Блокнот* (*Notepad*).
3. Наберите в окне программы простейший файл HTML.

```
<HTML>
     <HEAD>
          <TITLE>Учебный файл HTML</TITLE>
     </HEAD>
     <BODY>
        Расписание занятий на вторник
     </BODY>
</HTML>
```

4. Сохраните файл под именем **RASP.HTML** (обязательно укажите тип файла HTML при сохранении) в личной папке.
5. Для просмотра Web-страницы используйте любую программу браузера (*Internet Explorer*, *Opera*, *Mozilla Firefox* или другую). Для этого, не покидая программу Блокнот (сверните окно на панель задач), откройте личную папку и двойным кликом по файлу RASP.HTML откройте окно браузера.

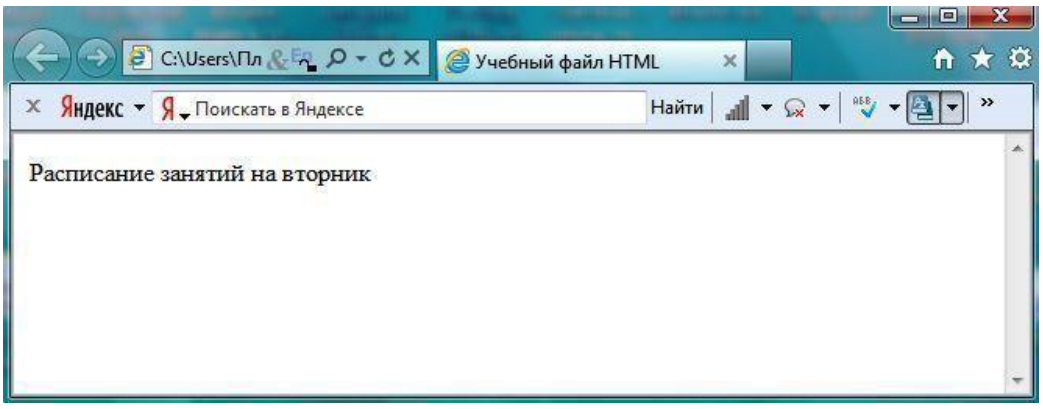


Рис.1

На экране вы увидите результат работы, изображенный на рисунке 1.

Задание № 2. *Управление расположением текста на экране*

1. При необходимости откройте текст Web-страницы в Блокноте (1 щелчок правой клавишей мыши по файлу RASP.HTML, в контекстном меню выбрать команду *Открыть с помощью...* и выбрать программу *Блокнот*). При необходимости открыть файл в браузере

– двойной клик по значку файла левой клавишей мыши.

2. Внести изменения в файл RASP.HTML, расположив слова *Расписание, занятий, на вторник* на разных строках.

```
<HTML>
     <HEAD>
          <TITLE>Учебный файл HTML</TITLE>
     </HEAD>
     <BODY>
        Расписан
        ие
        занятий
        на вторник
     </BODY>
</HTML>
```

3. Сохраните текст с внесенными изменениями в файле RASP.HTML (меню Файл | Сохранить). Если у вас уже отображается Web-страница, то вам достаточно переключиться на панели задач на программу браузера и обновить эту страницу (кнопка ↻). Изменилось ли отображение текста на экране?
   Не удивляйтесь тому, что внешний вид вашей Web-страницы не изменился.

Не забывайте каждый раз сохранять текст Web-страницы при ее корректировке в программе *Блокнот* и обновлять страницу при ее просмотре в программе браузера.

Задание № 3. *Некоторые специальные команды форматирования текста*

Существуют специальные команды, выполняющие перевод строки и задающие начало нового абзаца. Кроме того существует команда, запрещающая программе браузера изменять каким-либо образом изменять форматирование текста и позволяет точно воспроизвести на экране заданный фрагмент текстового файла.
Тег перевода строки **
** отделяет строку от последующего текста или графики.
Тег абзаца **<P>** тоже отделяет строку, но еще добавляет пустую строку, которая зрительно выделяет абзац.
Оба тега являются одноэлементными, тег <P> – двойной, т.е. требуется закрывающий тег.

1. Внесите изменения в файл RASP.HTML

```
<HTML>
     <HEAD>
          <TITLE>Учебный файл HTML</TITLE>
     </HEAD>
     <BODY>
        <P>Расписание</P>
        <BR>занятий<
        BR> на
        вторник
     </BODY>
```

```
</HTML>
```

2. Сохраните внесенные изменения, переключитесь на панели задач на программу браузера, обновите Web-страницу.
   Как изменилось отображение текста на экране? Выглядеть ваша Web-страница будет примерно так, как показано на рисунке 2.

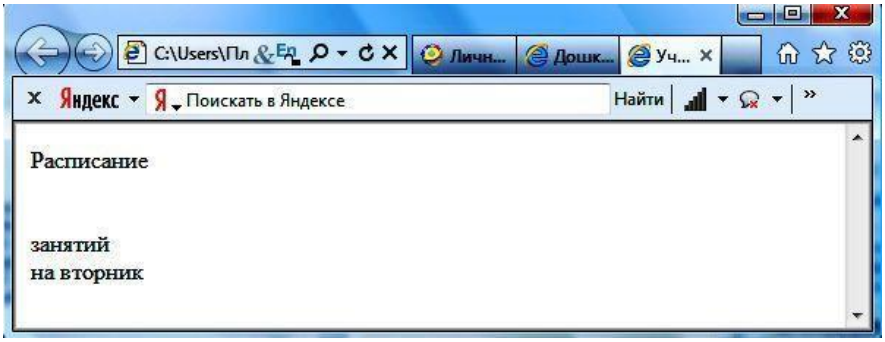


Рис. 2

Задание № 4. *Выделение фрагментов текста*

1. Внести изменения в текст файла RASP.HTML

```
<HTML>
     <HEAD>
          <TITLE>Учебный файл HTML</TITLE>
     </HEAD>
     <BODY>
        <B>Расписание</B>
        <I> занятий</I>
        <U> на вторник</U>
     </BODY>
</HTML>
```

2. Посмотрите полученную Web-страницу.

Возможно использование комбинированных выделений текста.

```
<I><B>Расписание</B></I>                    <I><U>
    занятий</U></I>        <U> на вторник</U>
```

Но при этом необходимо помнить следующее правило использования комбинированных тегов:

<Тег_1><Тег_2> ... </Тег_2></Тег_1> – ***правильная запись***.

<Тег_1><Тег_2> ... </Тег_1></Тег_2> – ошибочная запись.

Обратите внимание на «вложенность» тегов, она напоминает «вложенность» скобок.

Задание № 5. *Задание размеров символов Web-страницы*

Существует два способа управления размером текста, отображаемого браузером:

- использование стилей заголовка,
- задание размера шрифта основного документа или размера текущего шрифта.

Используется шесть тегов заголовков: от **<H1>** до **<H6>** (тег двойной, т.е. требует закрытия). Каждому тегу соответствует конкретный стиль, заданный параметрами настройки браузера.

1. Внесите изменения в файл RASP.HTML

```
<HTML>
     <HEAD>
          <TITLE>Учебный файл HTML</TITLE>
     </HEAD>
     <BODY>
        <P><H1>Расписание</H1></P>
        <I> занятий</I><U> на вторник</U>
     </BODY>
</HTML>
```

2. Просмотрите свою Web-страницу. На экране вы увидите то, что отображено на рисунке 3.

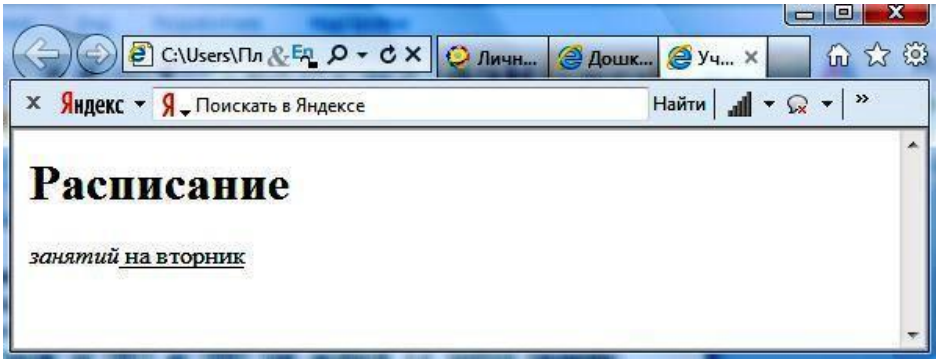


Рис. 3

Задание № 6. *Установка размера текущего шрифта*

Тег шрифта **<FONT>** позволяет задавать размер текущего шрифта в отдельных местах текста в диапазоне от 1 до 7.

1. Внесите изменения в текст RASP.HTML

```
<HTML>
     <HEAD>
          <TITLE>Учебный файл HTML</TITLE>
     </HEAD>
     <BODY>
        <FONT
        SIZE="7">Расписание</FONT>
        занятий на вторник
     </BODY>
</HTML>
```

2. Самостоятельно измените размер текста «занятий на вторник», используя тег <FONT>.

3. Измените оформление текста HTML-документа, используя тег выделения фрагментов и тег перевода строки и абзаца.

Задание № 7. *Установка гарнитуры и цвета шрифта*

Тег **<FONT>** предоставляет возможности управления гарнитурой, цветом и размером текста. Изменение гарнитуры текста выполняется простым добавлением к тегу **<FONT>** атрибута **FACE**. Например, для отображения текста шрифтом *Arial* необходимо записать:

```
<FONT FACE=”ARIAL”>
```

Для изменения цвета шрифта можно использовать в теге **<FONT>** атрибут **COLOR=”X”**. Вместо **“X”** надо подставить английское название цвета в кавычках (“ ”), либо его шестнадцатеричное значение. При задании цвета шестнадцатеричным числом необходимо представить этот цвет разложенным на три составляющие: красную (*R – Red*), зелѐную (*G – Green*), синюю (*B – blue*), каждая из которых имеет значение от **00** до **FF**. В этом случае мы имеем дело с так называемым форматом **RGB**.

Примеры записи текста в формате **RGB** приведены в Таблице 1:

Таблица 1

Запись текста в формате RGB

Цвет		RRGGBB	Цвет		RRGGBB	Цвет		RRGGBB
Black Черный		000000	Purple Фиолетовый		FF00FF	Green Зеленый		00FF00
White Белый		FFFFFF	Yellow Желтый		FFFF00	Azure Бирюзовый		00FFFF
Red Красный		FF0000	Brown Коричневый		996633	Blue Синий		0000FF
Orange Оранжевый		FF8000	Violet Лиловый		B000FF	Gray Серый		A0A0A0

1. Внесите изменения в файл RASP.HTML

```
<HTML>
    <HEAD>
        <TITLE>Учебный файл HTML</TITLE>
    </HEAD>
    <BODY>
    <U><I><B><FONT COLOR="#FF0000" FACE="ARIAL"
    SIZE="7">
       Расписание</FONT></B></I></U> занятий на
       вторник
    </BODY>
</HTML>
```

2. Самостоятельно измените размер, цвет, гарнитуру стиль текста документа.

Задание № 8. *Выравнивание текста по горизонтали*

1. Внесите изменения в файл RASP.HTML

```
<HTML>
     <HEAD>
          <TITLE>Учебный файл HTML</TITLE>
     </HEAD>
     <BODY>
        <P ALIGN="CENTER">
        <FONT COLOR="#008080" SIZE="7">
        <B>Расписание</B></FONT><BR>
        <FONT SIZE="6"><I> занятий на
        вторник</I></FONT>
        </P>
     </BODY>
</HTML>
```

2. Просмотрите изменения в браузере. На экране вы увидите то, что показано на рисунке 4.

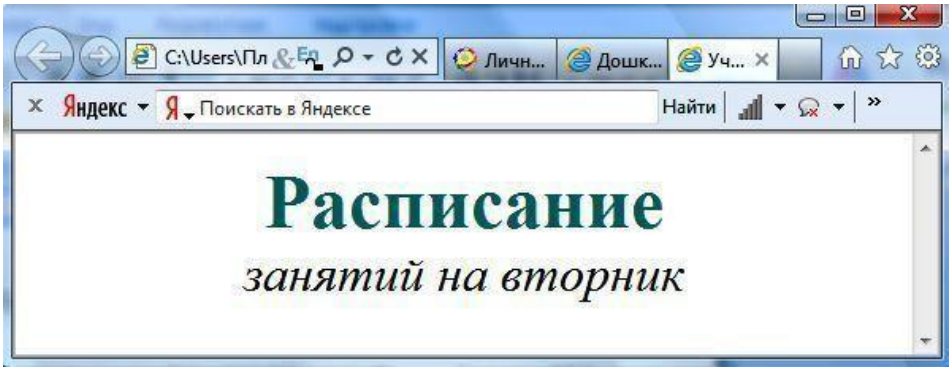


Рис. 4

Задание № 9. *Задание цвета фона и текста*

При изображении фона и цвета браузеры используют цвета, установленные по умолчанию, – они заданы параметрами настройки браузера. Если вы хотите задать другие цвета, то это надо сделать в начале файла HTML в теге **<BODY>**. Атрибут **BGCOLOR=** определяет цвет фона страницы, атрибут **TEXT=** задает цвет текста для всей страницы, атрибуты **LINK=** и **VLINK=** определяют соответственно цвета непросмотренных и просмотренных ссылок (последние два примера будут рассмотрены позже).

1. Внесите изменения в файл RASP.HTML

```
<HTML>
     <HEAD>
          <TITLE>Учебный файл HTML</TITLE>
     </HEAD>
     <BODY BGCOLOR="#FFFFCC" TEXT="#330066">
        <P ALIGN="CENTER">
        <FONT COLOR="#008080" SIZE="7">
        <B>Расписание</B></FONT><BR>
```

```
        <FONT SIZE="6"><I> занятий на
        вторник</I></FONT>
        </P>
    </BODY>
</HTML>
```

2. Просмотрите изменения Web-страницы в браузере.

Задание № 10. *Размещение графики на Web-странице*

Тег **<IMG>** позволяет вставить изображение на Web-страницу. Оно появится в том месте документа, где находится этот тег. Тег **<IMG>** является одиночным.
Необходимо помнить, что графические файлы должны находиться в той же папке, что и файл HTML, описывающий страницу. Графика в Web, как правило, распространяется в трех форматах: GIF, JPG, PNG.

Для выполнения следующего задания поместите файл с именем CLOCK.JPG (или другим именем) в рабочую папку.

Следует помнить, что для браузера важно, в каком регистре вы задаете описание имени и типа файла. Выработайте для себя определенное правило и строго следуйте ему. Если вы размещаете файл графического изображения во вложенной папке, то при описании изображения необходимо указывать путь доступа к файлу изображения, отображая вложенность папок.

1. Внесите изменения в файл RASP.HTML

```
<HTML>
    <HEAD>
        <TITLE>Учебный файл HTML</TITLE>
    </HEAD>
    <BODY BGCOLOR="#FFFFCC" TEXT="#330066">
        <P ALIGN="CENTER">
        <FONT COLOR="#008080" SIZE="7">
        <B>Расписание</B></FONT><BR>
        <FONT SIZE="6"><I> занятий на
        вторник</I></FONT>
        <BR><BR>
        <IMG SRC="CLOCK.PNG">
        </P>
    </BODY>
</HTML>
```

2. Просмотрите изменения вашей Web-страницы в браузере.

На экране вы увидите те, что показано на рисунке 5.

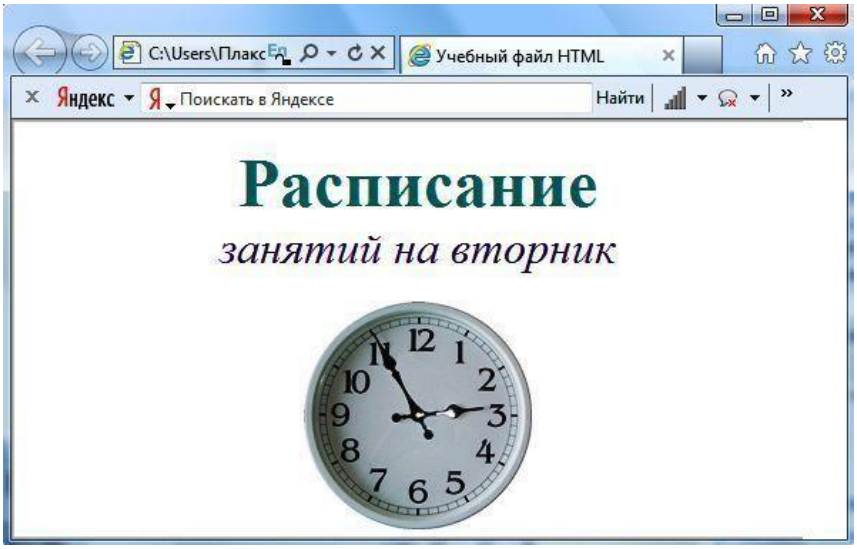


Рис. 5

Тег <IMG> имеет немало атрибутов, описанных в таблице 2. Эти атрибуты можно задавать дополнительно и располагаться они могут в любом месте тега после кода **IMG**.

Таблица 2

Атрибуты изображения

Атрибут	Формат	Описание
ALT	<IMG SRC="CLOCK.JPG" ALT="Картинка">	Задает текст, заменяющий изображение в том случае, если браузер не воспринимает изображение
BORDER	<IMG SRC="CLOCK.JPG" BORDER="3">	Задает толщину рамки вокруг изображения. Измеряется в пикселях
ALIGN	<IMG SRC="CLOCK.JPG" ALIGN="TOP">	Задает выравнивание изображения относительно текста: • относительно текста выровнена верхняя часть изображения – **"TOP"**, • относительно текста выровнена нижняя часть изображения – **"BOTTOM"**, • относительно текста выровнена средняя часть изображения – **"MIDDLE"**.
HEIGHT	<IMG SRC="CLOCK.JPG" HEIGHT="111">	Задает вертикальный размер изображения внутри окна браузера
WIDTH	<IMG SRC="CLOCK.JPG" WIDTH="210">	Задает горизонтальный размер изображения внутри окна браузера
VSPACE	<IMG SRC="CLOCK.JPG" VSPACE="8">	Задает добавление верхнего и нижнего пустых полей
HSPACE	<IMG SRC="CLOCK.JPG" HSPACE="8">	Задает добавление левого и правого пустых полей

Задание № 11. *Использование атрибутов изображения*

1. Самостоятельно внесите изменения в текст файла RASP.HTML: опробуйте использование таких атрибутов графики, как **ALT**, **BORDER**, **ALIGN**, **HEIGHT**, **WIDTH**, **VSPACE**, **HSPACE**.

Всегда обращайте внимание на размер графического файла (в байтах), так как это влияет на время загрузки Web-страницы.

2. Просмотрите изменения вашей Web-страницы в браузере.

Задание № 12. *Установка фонового изображения на Web-странице*

Фоновое изображение – это графический файл с небольшим рисунком, который многократно повторяется, заполняя все окно браузера независимо от его размеров. Графика, используемая в качестве фоновой, задается в теге **<BODY>**.

1. Внесите изменения в файл RASP.HTML, предварительно подготовив и сохранив в рабочей папке графический файл фонового рисунка (FON.PNG).

```
<HTML>
     <HEAD>
          <TITLE>Учебный файл HTML</TITLE>
     </HEAD>
     <BODY BACKGROUND="FON.PNG"
     TEXT="#330066">
        <P ALIGN="CENTER">
        <FONT COLOR="#008080" SIZE="7">
        <B>Расписание</B></FONT><BR>
        <FONT SIZE="6"><I> занятий на
        вторник</I></FONT>
        <BR><BR>
        <IMG SRC="CLOCK.PNG" ALIGN="MIDDLE">
        </P>
     </BODY>
</HTML>
```

На экране вы увидите то, что изображено на рисунке 6.

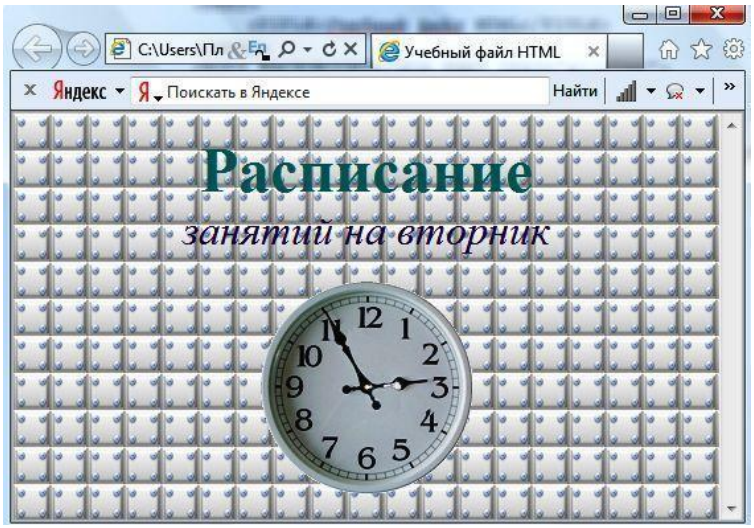


Рис. 6

Рисунок, который использовался в качестве фонового, имеет вид

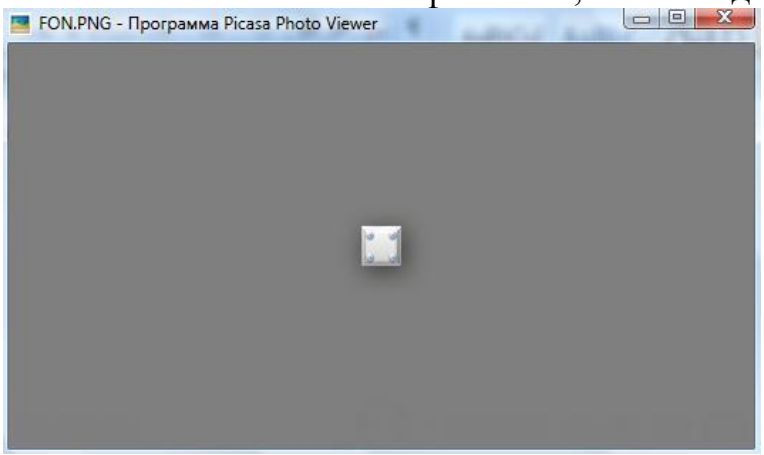


Рис. 7

2. Поэкспериментируйте с фоновым рисунком Web-страницы и выберите оптимальный с вашей точки зрения.

Задание № 13. *Создание таблицы*

Таблица является частью HTML-документа. Она представляет собой прямоугольную сетку, состоящую из вертикальных столбцов и горизонтальных строк. Пересечение строки и столбца называется ячейкой таблицы. Ячейка может содержать в себе текст, графику или другую таблицу.

Таблица состоит из трех основных частей:

- названия таблицы,
- заголовков столбцов,
- ячеек таблицы.

Таблица в Web-документе заполняется по строкам (слева направо по строке, затем переход на новую строку). **Каждая ячейка таблицы должна быть заполнена** (хотя бы пробелом, которые используются для создания пустых ячеек).

1. Запустите программу *Блокнот* и наберите текст следующей Web-страницы. Применяйте приемы копирования при создании таблицы, работая в программе *Блокнот*.

```
<HTML>
    <HEAD>
        <TITLE>Расписание занятий 5
        классов</TITLE>
    </HEAD>
    <BODY BGCOLOR="FFFFFF">
       <P ALIGN="CENTER">
       <FONT COLOR="RED" SIZE="6"
       FACE="ARIAL">
       <B>5 класс</B></FONT><BR></P>
       <FONT COLOR="BLUE" SIZE="4"
       FACE="COURIER">
       <B>Понедельник</B></FONT><BR>
           <TABLE BORDER="1" WIDTH=100%
                BGCOLOR="#99CCCC">
         <TR BGCOLOR="#CCCCFF" ALIGN="CENTER">
              <TD>Урок</TD> <TD>5 "А"</TD>
              <TD>5 "Б"</TD>
              <TD>5 "В"</TD>
              </TR>
              <TD>1</TD> <TD>Русский язык</TD>
              <TD>Литература</TD>
                                  <TD>История</
              TD>
              </TR>
              <TD>2</TD> <TD>Математика</TD>
              <TD>Информатика</TD>
              <TD>Английский язык</TD>
              </TR>
```

```
                    <TD>3</TD> <TD>История</TD>
                    <TD>Математика</TD>
                    <TD>Информатика</TD>
                </TR>
            </TABL
              E>
        </BODY>
</HTML>
```

2. Сохраните файл в личной рабочей папке под именем 5.HTML
3. Для просмотра созданной Web-страницы в окне личной рабочей папки двойным щелчком левой клавиши мыши загрузите браузер.

На экране вы увидите то, что показано на рисунке 8.

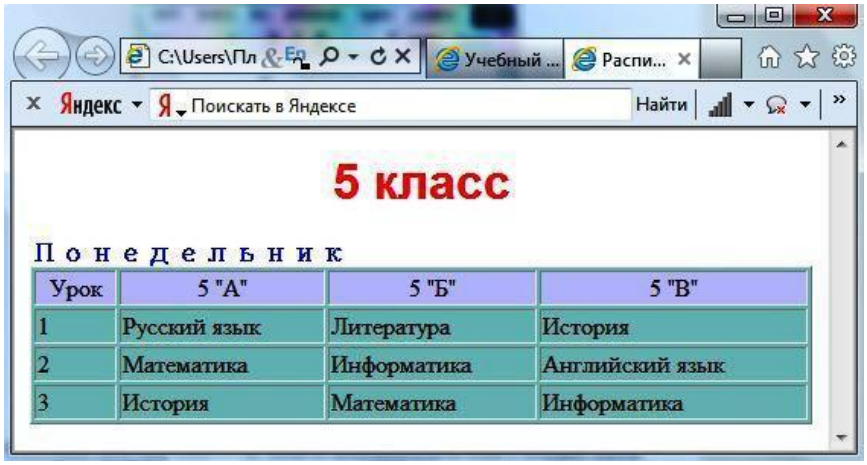


Рис. 8

Задание № 14. *Построение гипертекстовых связей*

Важнейшим средством языка HTML является возможность включения в документ ссылок на другие документы.
Возможны ссылки:

- на удаленный HTML-файл,
- на некоторую точку в текущем HTML-документе,
- на любой файл, не являющийся HTML-документом.

В качестве ссылки можно использовать любой текст или графику.

Ссылки в пределах одного документа

Такие ссылки требуют двух частей: метки и самой ссылки. Метка определяет точку, на которую происходит переход по ссылке. Ссылка использует имя метки. Ссылки выделяют цветом или подчеркиванием в зависимости от того, как настроен браузер. Для изменения цвета ссылки используются атрибуты **LINK=** и **VLINK=** тега **<BODY>**.

Описание ссылки

<A HREF="#ПН">Понедельник</A>

Перед именем метки (ПН), указывающей, куда надо перейти по ссылке, ставится символ #. Между символами “>” и “<” располагается текст (“Понедельник”), на котором должен быть произведен щелчок для перехода по ссылке.

Определим метку

<A NAME="ПН">Понедельник</A>

1. Дополните файл 5.HTML описанием таблицы, содержащей названия дней недели, поместив его в начало Web-страницы.

```
...
<TABLE WIDTH=100%>
  <TR>
        <TD>Понедельник</TD>
        <TD>Вторник</TD>
        <TD>Среда</TD>
        <TD>Четверг</TD>
        <TD>Пятница</TD>
        <TD>Суббота</TD>
    </TR>
  </TABLE>
  <BR>
...
```

2. Вставьте в файл 5.HTML метку, указывающую на понедельник.

```
...
<FONT COLOR="BLUE" SIZE="4"
FACE="COURIER"><B>
<A
NAME="ПН">Понедельник</A></B></FONT><BR>
...
```

3. Вставьте в таблицу с названиями дней недели ссылку для выбранной метки:

```
...
<TABLE WIDTH=100%>
  <TR>
 <TD><A HREF="#ПН">Понедельник</A></TD>
 <TD>Вторник</TD>
  <TD>Среда</TD>
```

4. Создайте таблицы расписаний для других дней недели.
5. Сохраните файл 5.HTML в личной рабочей папке.

6. Просмотрите полученную Web-страницу.

На экране вы увидите то, что изображено на рисунке 9.

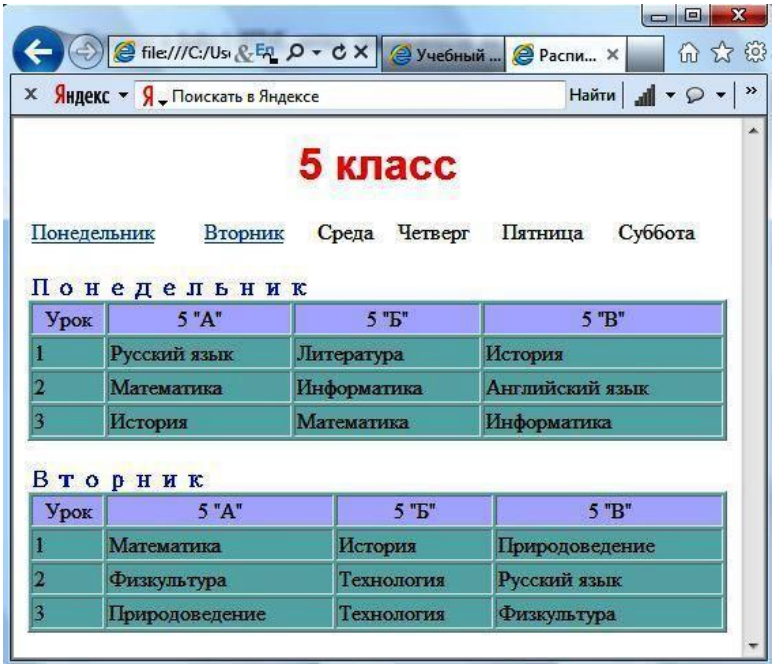


Рис. 9

Задание № 15. *Создание ссылки на другой HTML-документ*

Ссылки позволяют щелчком на выделенном слове или фразе

перейти к другому файлу. Опишем ссылку:

```
<A HREF="5.HTML">5 класс</A>
```

После имени файла (5.HTML) между символами «>» и «<» располагается текст («5 класс»), на котором должен быть произведен щелчок для перехода к этому файлу.

1. Внесите изменения в файл RASP.HTML

```
<HTML>
    <HEAD>
        <TITLE>Учебный файл HTML</TITLE>
    </HEAD>
    <BODY BGCOLOR="#FFFFFF" TEXT="#330066">
      <P ALIGN="CENTER">
      <FONT COLOR="#008080" SIZE="7">
      <B>Расписание</B></FONT><BR>
      <FONT SIZE="6"><I> занятий на
      вторник</I></FONT>
      <BR><BR>
      <IMG SRC="CLOCK.PNG" ALIGN="TOP">
```

```
        </P>
        <CENTER>
          <TABLE WIDTH=60%>
          <TR><TD><A HREF="5.HTML">5
          класс</A></TD>
          <TD>6 класс</TD></TR>
          <TR><TD>7 класс</TD>
          <TD>8 класс</TD></TR>
          <TR><TD>9 класс</TD>

          <TD>10 класс</TD></TR>
          <TR><TD>11 класс</TD>   </TR>
          </TABLE>
        </CENTER>
      </BODY>
</HTML>
```

2. Сохраните файл RASP.HTML
3. Просмотрите полученную Web-страницу.

На экране вы увидите то, что изображено на рисунке 10.

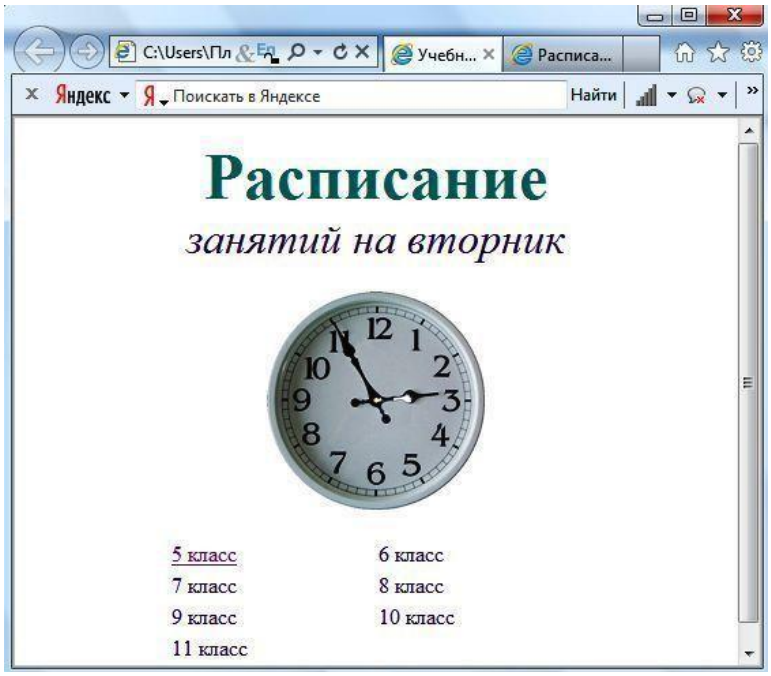


Рис. 10

Задание № 16. *Создание ссылки на другой HTML-документ*

1. Внесите изменения в файл 5.HTML так, чтобы в конце страницы была ссылка на головную страницу Расписание занятий 5 классов (RASP.HTML). В качестве ссылки используйте графический файл (HOME.GIF) следующим образом:

```
. . .
</TABLE><BR>
```

```
<CENTER>
<A HREF="RASP.HTML"><IMG SRC="HOME.PNG"
BORDER="0"></A>
</CENTER>
...
```

2. Просмотрите полученную Web- страницу.

На экране вы увидите то, что показано на рисунке 11.

В качестве ссылки выступает рисунок – стрелка

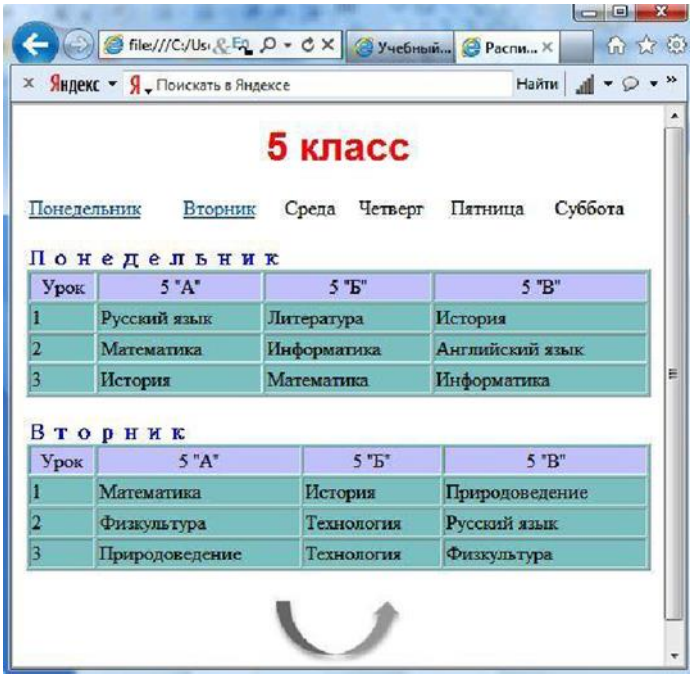


Рисунок 11

Задание № 17. *Самостоятельное итоговое задание*

Разработайте Web-страницы, рассказывающие о вашем классе. На головной странице разместите рассказ о классе, классном руководителе. Рассказы об учениках разместите на отдельных Web- страницах. Укажите ссылки на страницы учеников с головной Web-страницы. Не забудьте разместить ссылки возврата на головную страницу.

Как подготовить хорошую Web-страницу

1. Следует обратить внимание на простоту и логичность расположения информации на ваших страницах. Один из способов сделать информацию более легкой для восприятия – оставить на странице достаточно свободного места, не содержащего ни текста, ни рисунков. Страница, содержащая много информации, только отпугнет посетителя.

2. Постарайтесь представить информацию в виде списков или таблиц так, чтобы можно было достаточно легко найти важные сведения.
3. Не размещайте одно изображение сразу за другим. Попробуйте распределить их по документу, оставив достаточно свободного пространства.
4. Информация должна размещаться частями, легкими для восприятия. Обратите внимание на длину абзацев. Если абзац слишком длинный, разбейте его на несколько небольших абзацев.
5. Если Web-страница имеет большой объем, то, возможно, вам следует вставить ссылки, позволяющие пользователю быстро перемещаться между частями одного документа. Иногда имеет смысл вместо одного длинного документа подготовить одну страницу, содержащую перечень тем, каждую из которых раскрыть на отдельной Web-странице, и установить ссылки на соответствующие Web-страницы.
6. Использование графики может дополнительно привлечь пользователей. Но необходимо помнить о времени загрузки вашей страницы, которое определяется количеством и объемом графической информации. Красивая картинка не произведет никакого впечатления, ели для того, чтобы ее увидеть, придется долго ждать, пока она загрузится.

Тестирование

Перед тем как выставлять свои Web-страницы на сервер необходимо их протестировать. Созданные документы должны пройти «локальную проверку» в пределах вашего жесткого диска. При проверке используйте разные браузеры. Вы увидите различия, которые могут оказаться довольно существенными.

В рамках тестирования необходимо сделать следующее:

1. *Проверить правописание.* Выполните автоматизированную проверку правописания текста (для этого можно использовать Microsoft Word) или попросите кого-нибудь выполнить корректуру.
2. *Проверит навигацию.* Убедитесь, что на каждой странице присутствуют необходимые средства навигации, все ссылки работают правильно.
3. *Проверить доступ к внешним файлам.* Выясните, размещены ли графические, звуковые или видеофайлы там, где они могут быть найдены и откуда их можно загрузить (должен быть правильно указан путь доступа). Для неграфических браузеров требуется задать подменяющие текстовые сообщения.
4. *Проверить, допустимо ли время загрузки.*
5. *Осуществить проверку ваших Web-страниц посторонним лицом.* Попросите кого-нибудь, кто не знаком с вашими документами, пройти их от начала до конца. Иногда при этом выясняются такие факты, каких вы сами ни за что бы не заметили.

Контрольные вопросы

1. Что такое блочная вёрстка?
2. Как создать вложенный плавающий блок?.
3. Что задаёт свойство display?
4. Как создать панель навигации?
5. Чем обеспечивается выравнивание плавающих элементов?
6. С чего начинается создание макета web страницы?
7. Какие виды позиционирования могут применяться на web странице?

8. Когда применяется фиксированное позиционирование?

Практическое занятие № 4 Программирование клиентских приложений на языке JavaScript

Цель: Изучить приемы создания документов, исользующих Drag & Drop

Задание:

1. Ознакомиться с теоретическим материалом по теме.
2. Создайте документ со следующим сценарием. Пользователь нажал клавишу мыши в каком-либо месте на окне браузера. Наш скрипт должен зафиксировать это событие и вычислить, с каким объектом (то есть слоем) это было связано.
3. Создайте документ, в котором демонстрируется применение MouseMove - текущие координаты курсора мыши отображаются в окне состояния.
4. Объедините оба последних задания. Нужно, чтобы были представлены координаты указателя мыши, когда пользователь перемещает мышь, нажав на клавишу.
5. Создать документ, в котором нужно определить, по какому именно слою пользователь щелкнул клавишей мыши. И затем этот объект должен двигаться вслед за мышью.

Необходимые приборы: ПК, текстовый редактор Блокнот, браузер

Методические рекомендации к выполнению лабораторной работы:

Методические рекомендации к выполнению задания 1

С помощью новой модели событий в языке JavaScript, 1.2 и механизма слоев можно реализовать на web-странице схему drag & drop ("перетащил и оставил"). Для этого понадобится, по крайней мере, Netscape Navigator 4.0, поскольку нужно пользоваться особенностями языка JavaScript 1.2.

Механизм drag & drop, который мы хотим здесь реализовать, ограничивается web-страницей. Поэтому нельзя использовать представленный здесь код, чтобы переносить объекты с HTML-страницы на жесткий диск вашего компьютера или другие подобные действия.

Язык JavaScript не поддерживает напрямую механизм drag & drop. Это значит, что нет возможности назначить объекту image свойство dragable (перемещаемый) или что-либо в этом роде. Поэтому мы должны сами писать необходимый для этого код. Это не так сложно. Для этого нужны две вещи. Во-вервых, нужнозарегистрировать определенные события, связанные с работой мышью, то есть нужно понять, каким образом, можно узнать, какой объект необходимо переместить и на какую позицию? Затем нужно подумать, каким именно образом будет показываться перемещение объектов по экрану. Конечно же,

желательно пользоваться слоями при создании объектов и перемещении их по экрану. Каждый объект представлен собственным слоем.

События при работе с мышью в JavaScript 1.2

Какие события, происходящие при работе с мышью, следует использовать? Нет такого события, как*MouseDrag*, однако того же самого можно достичь, отслеживая события *MouseDown, MouseMove* и *MouseUp*. В версии 1.2языка JavaScript используется новая модель событий. И без нее нельзя решить эту задачу. Взглянем на некоторые важные ее части еще раз.

MouseDown, MouseMove и MouseUp

Вязыке JavaScript нетсобытия MouseDrag. Поэтому мы должны пользоваться событиями MouseDown, MouseMove и MouseUp, реализуя механизм drag & drop.

"Оставляемые" объекты

Предположим, нужно создать онлайновый магазин. Есть несколько изделий, которые можно поместить в корзину. Пользователь должен переносить эти изделия в корзинку и оставлять их там. Это означает, что нужно регистрировать моменты, когда пользователь опускает некий объект в корзину - иными словами, что он хочет купить его.

Какую часть кода мы должны изменить, чтобы сделать такое? Нужно проверить, в какой месте оказался объект после того, как было зафиксировано событие *MouseUp* - то есть нужно сделать некоторые добавления к функции *endDrag()*. Например мы могли бы проверять, попадает ли в этот момент курсор мыши в границы некого прямоугольника. Если это так, то вызывается функция, регистрирующая все изделия, которые необходимо купить (например, можно поместить их в некий массив). Ну и после этого можное показывать это изделие уже в корзинке.

Реализации

Есть несколько путей для совершенствования скрипта. Во-первых, мы могли бы изменять порядок следования слоев, как только пользователь щелкает клавишей мыши по какому-либо объекту. Иначе выглядело бы странным, если бы вы перемещали объект, а он при этом прятался от вас за окружающие предметы. Очевидно, что эту проблему можно решить, меняя лишь порядок следования слоев в функции *startDrag()*.

Методические рекомендации к выполнению задания 2

Создайте документ со следующим сценарием. Пользователь нажал клавишу мыши в каком-либо месте на окне браузера. Наш скрипт должен зафиксировать это событие и вычислить, с каким объектом (то есть слоем) это было связано. Нам необходимо знать координаты точки, где произошло это событие. В JavaScript 1.2 реализован новый объект Event, который сохраняет координаты этой точки (а также еще и другую информацию о событии).

Другой важный момент заключается в перехвате событий. Если пользователь, например, щелкает по клавише мыши, то сигнал о соответствующем событии посылается непосредственно объекту button. Однако в нашем примере необходимо, чтобы событие обрабатывалось объектом window (окно). Поэтому мы позволяем объекту окна **перехватывать** сигнал о событии, связанном с мышью, т.е. чтобы именно объект window фиксировал это событие и имел возможность на него реагировать. Это демонстрируется в следующем примере (на примере события *Click*). Вы можете щелкнуть в любом месте окна

браузера. При этом возникнет окно сообщения, где будут показаны координаты точки, где это событие имело место.

Код:

```
<html>
<script language="JavaScript">
<!--
  window.captureEvents(Event.CLICK);
  window.onclick= displayCoords;
  function displayCoords(e) {
    alert("x: " + e.pageX + " y: " + e.pageY);
}
// -->
</script>
"Кликните" клавишей мыши где-нибудь в этом окне.
</html>
```

Сперва мы сообщаем, что объект window перехватывает сигнал о событии *Click*. Для этого мы пользуемся методом *captureEvent()*. Строка

```
  window.onclick= displayCoords;
```

говорит о том, что должно происходить, когда случается событие *Click*. Конкретнее, здесь сообщается, что в качестве реакции на событие *Click* браузер должен вызвать процедуру *displayCoords()* (Заметим, что Вам при этом **не следует**ставить скобки после слова *displayCoords*). В свою очередь, *displayCoords()* - это функция, которая определяется следующим образом:

```
function displayCoords(e) {
   alert("x: " + e.pageX + " y: " + e.pageY);
}
```

Как видите, эта функция имеет аргумент (мы назвали его *e*). На самом деле это объект Event, который передается на обработку функции *displayCoords()*. Объект Event имеет свойства *pageX* и *pageY* (наряду с другими), из которых моно получить координаты точки, где произошло событие. Окно с сообщением лишь показывает эти значения.

Методические рекомендации к выполнению задания 3

Создайте документ, в котором демонстрируется применение *MouseMove* - текущие координаты курсора мыши отображаются в окне состояния.

од скрипта почти такой же, как и в предыдущем задании:

```
<html>
<script language="JavaScript">
<!--
  window.captureEvents(Event.MOUSEMOVE);
  window.onmousemove= displayCoords;
  function displayCoords(e) {
    status= "x: " + e.pageX + " y: " + e.pageY;
}
// -->
</script>
```

Координаты мыши отображаются в строке состояния.

</html>

Заметьте, что нуно написать именно *Event.MOUSEMOVE*, где слово *MOUSEMOVE* обязательно должно быть написано заглавными буквами. А указывая, какая функция должна быть вызвана, когда произойдет событие MouseMove, Вы должны писать ее строчными буквами: *window.onmousemove=...*

Методические рекомендации к выполнению задания 4

Объедините оба последних задания. Нужно, чтобы были представлены координаты указателя мыши, когда пользователь **перемещает мышь, нажав на клавишу**.

Код этого примера выглядит следующим образом:

```
<html>
<script language="JavaScript">
<!--
window.captureEvents(Event.MOUSEDOWN | Event.MOUSEUP);
window.onmousedown= startDrag;
window.onmouseup= endDrag;
window.onmousemove= moveIt;
function startDrag(e) {
  window.captureEvents(Event.MOUSEMOVE);}
function moveIt(e) {
  // показыватькоординаты
  status= "x: " + e.pageX + " y: " + e.pageY;}
function endDrag(e) {
  window.releaseEvents(Event.MOUSEMOVE);}
// -->
</script>
```

Нажмите на клавишу мыши и, не отпуская ее, передвиньте саму мышь. Координаты курсора будут отображаться в строке состояния.

</html>

Во-первых, мы заставляем объект window перехватывать сигналы о событиях *MouseDown* and *MouseUp*:

window.captureEvents(Event.MOUSEDOWN | Event.MOUSEUP);

Как видно, мы пользуемся символом | (или), чтобы сказать, что объект window должен перехватывать несколько указанных событий. Следующие две строки описывают, что именно должно происходить, когда указанные события имеют место:

window.onmousedown= startDrag;

window.onmouseup= endDrag;

В следующей строке кода определяется, что происходит, когда объект window получает сигнал о событии *MouseMove*:

window.onmousemove= moveIt;

Однако постойте, мы же не определили *Event.MOUSEMOVE* в *window.captureEvents()*! Это означает, что данное событие не будет перехватываться объектом window. Тогда почему мы указываем объекту window вызывать *moveIt()*, раз сигнал об этом событии никогда не достигает объекта window? Ответ на этот вопрос можно

найти в функции *startDrag()*, которая вызывается сразу после того, как произойдет событие *MouseDown*:

```
function startDrag(e) {
  window.captureEvents(Event.MOUSEMOVE);}
```

Это означает, что объект window начнет перехватывать событие *MouseMove*, как только будет нажата клавиша кнопка мыши. И мы должны прекратить перехватывать событие *MouseMove*, если произойдет событие *MouseUp*. Это делается в функции *endDrag()* с помощью метода *releaseEvents()*:

```
function endDrag(e) {
  window.releaseEvents(Event.MOUSEMOVE);}
```

Функция *moveIt()* записывает координаты мыши в окно состояния.

Теперь у нс есть все элементы скрипта, необходимые для регистрации событий, связанных с реализацией механизма drag & drop. И мы можем приступить к рисованию на экране наших объектов.

Методические рекомендации к выполнению задания 5

На предыдущих занятиях мы видели, как с помощью слоев можно создать перемещающиеся объекты. Все, что мы должны теперь сделать - это определить, по какому именно слою пользователь щелкнул клавишей мыши. И затем этот объект должен двигаться вслед за мышью.

Код задания:

```
<html>
<head>
<script language="JavaScript">
<!--
var dragObj= new Array();
var dx, dy;
window.captureEvents(Event.MOUSEDOWN | Event.MOUSEUP);
window.onmousedown= startDrag;
window.onmouseup= endDrag;
window.onmousemove= moveIt;
function startDrag(e) {
  currentObj= whichObj(e);
  window.captureEvents(Event.MOUSEMOVE);}
function moveIt(e) {
  if (currentObj != null) {
    dragObj[currentObj].left= e.pageX - dx;
    dragObj[currentObj].top= e.pageY - dy;  }}
function endDrag(e) {
  currentObj= null;
  window.releaseEvents(Event.MOUSEMOVE);}
function init() {
  // задать 'перемещаемые' слои
  dragObj[0]= document.layers["layer0"];
  dragObj[1]= document.layers["layer1"];
```

```
  dragObj[2]= document.layers["layer2"];}
function whichObj(e) {
  // определить, по какому объекту был произведен щелчок
var hit= null;
  for (var i= 0; i < dragObj.length; i++) {
    if ((dragObj[i].left < e.pageX) &&
        (dragObj[i].left + dragObj[i].clip.width > e.pageX) &&
        (dragObj[i].top < e.pageY) &&
        (dragObj[i].top + dragObj[i].clip.height > e.pageY)) {
          hit= i;
          dx= e.pageX- dragObj[i].left;
          dy= e.pageY- dragObj[i].top;
          break;    }
}
  return hit;}
// -->
</script>
</head>
<body onLoad="init()">
<layer name="layer0" left=100 top=200 clip="100,100" bgcolor="#0000ff">
<font size=+1>Object 0</font>
</layer>
<layer name="layer1" left=300 top=200 clip="100,100" bgcolor="#00ff00">
<font size=+1>Object 1</font>
</layer>
<layer name="layer2" left=500 top=200 clip="100,100" bgcolor="#ff0000">
<font size=+1>Object 2</font>
</layer>
</body>
</html>
```

Можно видеть, что в разделе <body> нашей HTML-страницы мы определяем три слоя. После того, как была загружена вся страница, при помощи программы обработки события *onLoad*, указанной в тэге <body>, вызывается функция *init()*:

```
function init() {
  // задать 'перемещаемые' слои
  dragObj[0]= document.layers["layer0"];
  dragObj[1]= document.layers["layer1"];
  dragObj[2]= document.layers["layer2"];}
```

Массив *dragObj* включает все слои, которые пользователь может перемещать. Каждый такой слой получает в множестве *dragObj* некий номер. Его мы рассмотрим попозже.

Можно видеть, что мы используем тот же самый код, что использовался ранее для перехвата событий, связанных с мышью:

```
window.captureEvents(Event.MOUSEDOWN | Event.MOUSEUP);
window.onmousedown= startDrag;
```

window.onmouseup= endDrag;

window.onmousemove= moveIt;

К функции *startDrag()* я добавил следующую строку:

currentObj= whichObj(e);

Функция *whichObj()* определяет, по какому объекту был произведен щелчок. Возвращает она номер соответствующего слоя. Если ни один слой не был выделен, то возвращается значение *null*. Полученное значение хранится в переменной *currentObj*. Это означает, что из *currentObj* можно извлечь номер слоя, который в данный момент необходимо перемещать (либо это будет *null*, если никакого слоя перемещать не надо).

В функции *whichObj()* для каждого слоя мы проверяем свойства *left*, *top*, *width* и *height*. По этим значеням мы и можем проверять, по которому из объектов пользователь щелкнул клавишей.

Контрольные вопросы

1. Что такое динамический html?
2. В чем сущность объектной модели программирования?
3. Какими свойствами обладают селекторы потомков?
4. Что такое селекторы дочерних элементов?
5. Что такое DOM (Document Object Model)?
6. Перечислите основные узлы дерева HTML документа.
7. Как применять программный интерфейс HTML DOM?
8. Как изменить свойство узлов?
9. Как применять регулярные выражения в JavaScript ?
10. Где могут быть размещены выражения JavaScript ?
11. В чем сущность объектной модели браузера?
12. Для чего применяется библиотека jQuery?

Практическое занятие № 5 Создание форм

Тема:

HTML формы — сложные элементы интерфейса. Они включают в себя разные функциональные элементы: поля ввода <input> и <textarea>, списки <select>, подсказки и т.д. Весь код формы заключается в элемент <form>.

Большая часть информации веб-форм передаётся с помощью элемента <input>. Для ввода одной строки текста применяется элемент <input type="text">, для нескольких строк — элемент <textarea>. Элемент <select> создает выпадающий список.

Элемент <label> создаёт надписи к полям формы. Существует два способа группировки надписи и поля. Если поле находится внутри элемента <label>, то атрибут for указывать не нужно.

<label for="lastname">Last Name</label><input type="text" id="lastname">

<input type="text" id="lastname"><label for="lastname">Last Name</label>

<label>Last Name<input type="text" name="lastname"></label>

HTML

Поля формы можно разделять на логические блоки с помощью элемента <fieldset>. Каждому разделу можно присвоить название с помощью элемента <legend>.

<fieldset>
 <legend>Контактная информация</legend>
 <label>Имя<input type="text" required></label>
 <label>E-mail<input type="email" required></label>
</fieldset>

HTML

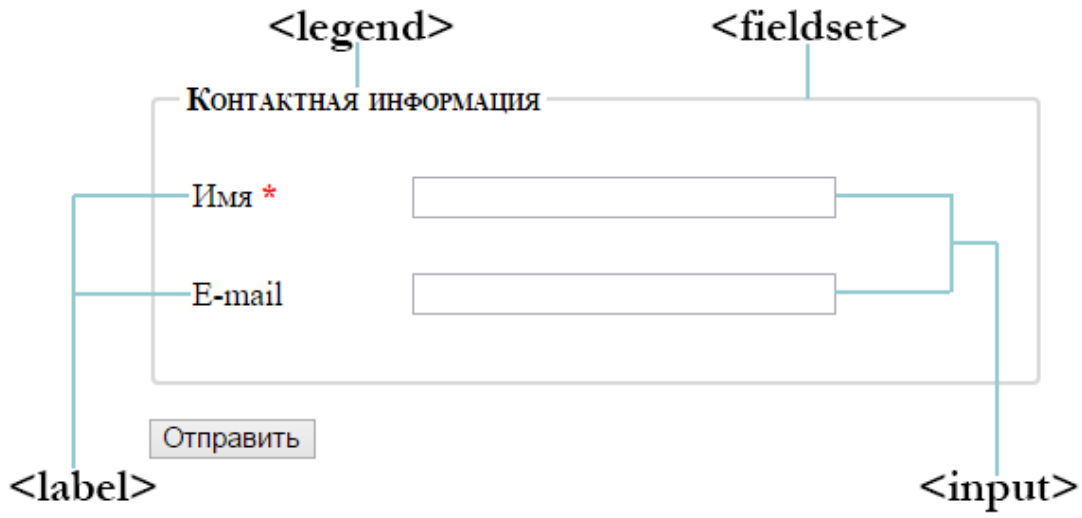


РИС. 1. ГРУППИРОВКА ПОЛЕЙ ФОРМЫ

Чтобы сделать форму более понятной для пользователей, в поля формы добавляют текст, содержащий пример вводимых данных. Такой текст называется подстановочным и создаётся с помощью атрибута placeholder.

Обязательные для заполнения поля также необходимо выделять. До появления HTML5 использовался символ звездочки *, установленный возле названия поля. В новой спецификации появился специальный атрибут required, который позволяет отметить

обязательное поле на уровне разметки. Этот атрибут дает указание браузеру (при условии, что тот поддерживает HTML5), указание не отправлять данные после нажатия пользователем кнопки отправить, пока указанные поля не заполнены.

<input type="text" required placeholder="Ваше имя">

HTML

Для изменения внешний вид текстового поля при получении фокуса, используется псевдокласс focus. Например, можно сделать фон текущего поля более темным или добавить цветную рамку, чтобы оно выделялось среди остальных:

input:focus {
background: #eaeaea;
}

CSS

Ещё один полезный html5-атрибут — атрибут autofocus. Он позволяет автоматически установить фокус на нужном начальном поле для элементов <input> и <textarea> (только в один элемент каждой формы).

Пример создания формы регистрации

```
<div class="form-wrap">
  <div class="profile"><img src="https://html5book.ru/wp-content/uploads/2016/10/profile-image.png">
    <h1>Регистрация</h1>
  </div>
  <form method="post" action="form.php">
    <div>
      <label for="name">Имя</label>
      <input type="text" name="name" required>
    </div>
    <div class="radio">
      <span>Пол</span>
      <label>
        <input type="radio" name="sex" value="мужской">мужской
        <div class="radio-control male"></div>
      </label>
      <label>
        <input type="radio" name="sex" value="женский">женский
        <div class="radio-control female"></div>
      </label>
    </div>
    <div>
      <label for="email">E-mail</label>
      <input type="email" name="email" required>
    </div>
    <div>
      <label for="country">Страна</label>
      <select name="country">
        <option>Выберите страну проживания</option>
        <option value="Россия">Россия</option>
        <option value="Украина">Украина</option>
        <option value="Беларусь">Беларусь</option>
      </select>
      <div class="select-arrow"></div>
    </div>
    <button type="submit">Отправить</button>
```

```
  </form>
</div>
HTML
```

Примечание

action="form.php" — ссылка на файл обработчика формы. Создайте файл в кодировке UTF-8, закачайте его на сервер и замените action="form.php" на путь к файлу на вашем сервере.

Контрольные вопросы

1. Что такое веб-приложение?
2. Что такое браузер?
3. Опишите цикл обработки запроса к веб-приложению от клиента.
4. Для чего необходимы технологии разработки веб-приложений (такие как ASP.NET, PHP, Ruby On Rails и др.).
5. Как работает протокол HTTP и для чего он нужен?
6. Что такое заголовки HTTP-сообщения и для чего они нужны?
7. Что такое тело HTTP-сообщения?
8. Каким образом в HTTP-сообщении заголовки отделяются от тела сообщения?
9. Что такое метод HTTP-запроса?
10. Что такое статусный код HTTP-ответа?
11. Приведите примеры HTTP-заголовков HTTP-запроса и HTTP-ответа.
12. Что такое веб-сервер?

Практическое занятие № 6 Создание базы данных MySQL

Цель работы:

Что такое MySQL.

MySQL – компактный многопоточный сервер баз данных. MySQL характеризуется большой скоростью, устойчивостью и легкостью в использовании.

MySQL был разработан компанией TcX для внутренних нужд, которые заключались в быстрой обработке очень больших баз данных. Компания утверждает, что использует MySQL с 1996 года на сервере с более чем 40 БД, которые содержат 10,000 таблиц, из которых более 24.01.2004 чем 500 имеют более 7 миллионов строк.

MySQL является идеальным решением для малых и средних приложений. Исходники сервера компилируются на множестве платформ. Наиболее полно возможности сервера проявляются на Unix-серверах, где есть поддержка многопоточности, что дает значительный прирост производительности.

На текущий момент MySQL все еще в стадии разработки, хотя версии 3.22 полностью работоспособны.

MySQL-сервер является бесплатным для некоммерческого использования. Иначе необходимо приобретение лицензии, стоимость которой составляет 190 EUR.

Возможности MySQL.

MySQL поддерживает язык запросов SQL в стандарте ANSI 92, и кроме этого имеет множество расширений к этому стандарту, которых нет ни в одной другой СУБД.

Краткий перечень возможностей MySQL.

Поддерживается неограниченное количество пользователей, одновременно работающих с базой данных.

Количество строк в таблицах может достигать 50 млн.

Быстрое выполнение команд. Возможно MySQL самый быстрый сервер из существующих.

Простая и эффективная система безопасности.

MySQL действительно очень быстрый сервер, но для достижения этого разработчикам пришлось пожертвовать некоторыми требованиями к реляционным СУБД. В MySQL отсутствуют:

Поддержка вложенных запросов, типа SELECT * FROM table1 WHERE id IN (SELECT id FROM table2).

Не реализована поддержка транзакций. Взамен предлагается использовать LOCK/UNLOCK TABLE.

Нет поддержки триггеров и хранимых процедур.

По словам создателей именно эти пункты дали возможность достичь высокого быстродействия. Их реализация существенно снижает скорость сервера. Эти возможности

не являются критичными при создании Web-приложений, что в сочетании с высоким быстродействием и малой ценой позволило серверу приобрести большую популярность

Работа с MySQL (сохранение данных в базе данных).

Для начала создаем базу данных и таблицу. Входим в MySQL, и выполняем команды:

>CREATE DATABASE products;

>CREATE TABLE clients (name VARCHAR(25), email VARCHAR(25), choise VARCHAR(8));

Для общения с MySQL из PHP понадобятся следующие функции.

int mysql_connect(string hostname, string username, string password);

Создать соединение с MySQL.

Параметры:

Hostname – имя хоста, на котором находится база данных.

Username – имя пользователя.

Password – пароль пользователя.

Функция возвращает параметр типа int, который больше 0, если соединение прошло успешно, и равен 0 в противном случае.

int mysql_select_db(string database_name, int link_identifier);

Выбрать базу данных для работы.

Параметры:

Database_name – имя базы данных.

link_identifier – ID соединения, которое получено в функции mysql_connect. (параметр необязательный, если он не указывается, то используется ID от последнего вызова mysql_connect)

Функция возвращает значение true или false

int mysql_query(string query, int link_identifier);

Функция выполняет запрос к базе данных.

Параметры:

Query – строка, содержащая запрос

link_identifier – см. предыдущую функцию.

Функция возвращает ID результата или 0, если произошла ошибка.

int mysql_close(int link_identifier);

Функция закрывает соединение с MySQL.

Параметры:

link_identifier – см. выше.

Функция возвращает значение true или false

1. **Создадим таблицу Customer**
2.

Поля создаваемой таблицы:

id, FirstName, LastName, Login, Password. Поле id – автоинкрементный первичный ключ.

Код программы:

create.php

```
<?php
$hostname='DOC200';
$database='________';
@mysql_connect($hostname,'_________________','____________________');
@mysql_select_db($database);
$q="CREATE TABLE `Customer_номер студенческого`
 (`id` INT (3) UNSIGNED DEFAULT '0',
`FirstName` VARCHAR (128),
PRIMARY KEY(`id`)
) ";
$r=mysql_query($q);
@mysql_free_result($r);
mysql_close();
?>
```

2. Выведем список имеющихся таблиц в БД

show_tables.php

```
<?php
$hostname='DOC200';
$database='_______';
@mysql_connect($hostname,'_______','______________');
@mysql_select_db($database);
$r = mysql_listtables ($database);
$i = 0;
while ($i < @mysql_num_rows ($r)) {
   $ names[$i] = mysql_tablename ($r, $i);
   print $names[$i] . "<BR>";
   $i++;
}
@mysql_free_result($r);
```

```
mysql_close();
?>
```

3. Удалим таблицу Customer

delete_table.php

```
<?php
$hostname='DOC200';
$database='_____________';
@mysql_connect($hostname,'_____________','_____________');
@mysql_select_db($database);
$q="DROP TABLE `id301`.`customer`";
$r=mysql_query($q);
mysql_free_result($r);
mysql_close();
?>
```

Упражнение. Создайте административный скрипт для интерактивного создания и удаления таблиц БД.

Шаг 1. Создайте меню(**admin.html**).

- Показать список таблиц в БД
- Удалить таблицу
- Создать таблицу

Расположите на форме кнопку «Далее >>». При нажатии кнопки перейдите на **admin.php**

Шаг 2. Создайте обработчик формы(**admin.php**).

Используйте конструкцию switch.

1. Если выбран пункт меню 1, выполняйте **show_tables.php**. В конце расположите форму с копкой «Назад», возвращающую администратора к основному меню (**admin.html**).

2. Если выбран пункт меню 2, выполняйте **delete_table.php.** Для пункта «Удалить таблицу» предлагайте список имен всех имеющихся таблиц в БД. В конце расположите форму с копкой «Назад», возвращающую администратора к основному меню (**admin.html**).

3. Если выбран пункт меню 3, выполняйте **create.php**.

Если и флаг создания полей (имя произвольное) не установлен, то предложите форму:

Выберите имя БД список

Количество полей в таблице поле ввода

Расположите на форме кнопку «Далее >>». При нажатии кнопки перейдите на **admin.php**, установив флаг выбора меню в «3» и флаг создания полей в «1».

В скрипте проверьте, если флаг создания полей установлен в «1», то предложите пользователю таблицу:

Имя поля 1	Тип поля 1	Размер	По умолчанию	Первичный ключ	Уникальный столбец	Автоинкремент
поле ввода	список	поле ввода	поле ввода	флажок	флажок	флажок

.
.
.

Имя поля N	Тип Поля N	Размер	По умолчанию	Первичный ключ	Уникальный столбец	Автоинкремент
поле ввода	список	поле ввода	поле ввода	флажок	флажок	флажок

Расположите на форме кнопку «Далее >>». При нажатии кнопки перейдите на **admin.php**, установив флаг выбора меню в «3» и флаг создания полей в «2».

В скрипте проверьте, если флаг создания полей установлен в «2», то сформируйте запрос на создание таблицы и выполните его. Выведите пользователю сообщение «Таблица создана».

Выведите форму с копкой «Назад», возвращающую администратора к основному меню (**admin.html**).

Замечание. Типы полей списка «Тип поля»: INT, BIGINT, SMALLINT, FLOAT, DOUBLE, DATE, DATETIME, CHAR, VARCHAR

Контрольные вопросы

1. Какие базы данных применяются в web – программировании?
2. Дайте сравнительный анализ MySQL, PostgreSQL.
3. Опишите программный интерфейс, выполнение запросов.
4. Охарактеризуйте базы данных Microsoft. Microsoft SQL.
5. Как осуществляется доступ к базе данных MySQL?
6. Опишите управление базой данных с помощью веб-приложения PHPMyAdmin.
7. Как создать таблицы базы данных?
8. Как выполнить вывод списка имеющихся таблиц в БД?
9. Как осуществить удаление таблиц?
10. Как осуществить редактирование атрибутов таблиц?
11. Как осуществить редактирование содержания таблиц?
12. Как выполнить SQL запрос?

Практическое занятие № 7 События и их обработка

В Java **событие** - это специальный объект, описывающий изменение состояния источника. Это может быть, например, щелчок кнопки, ввод символа с клавиатуры, выбор элемента в списке и т.д. Событие может происходит и без участия пользователя, например, при использовании таймера. Также можно создавать собственные события.

Хочется отметить, что любые операционные системы, поддерживающие графический пользовательский интерфейс, непрерывно отслеживают такие события, как нажатие клавиш или щелчок мыши, а затем сообщают о них выполняемой программе. Каждая программа решает, как реагировать на эти события (если это соответсвенно предусмотрено ее кодом).

В Java при обработке событий, предусмотренных в библиотеке AWT, программист

полностью контролирует передачу событий от источников событий (event sources)

(например, кнопок или полос прокруток) к слушателю событий (event listener). Любой

объект можно считать слушателем некоего события — на практике все объекты так или

иначе реагируют на события.

А **слушатель** (listener) - это объект, уведомляемый о событии. Он должен быть зарегистрирован источником событий и реализовывать методы для получения и обработки уведомлений.

Таким образом, обработка событий основана на модели делегирования событий (delegation event model) - источник извещает о событии одного или несколько слушателей (listener).

Источники событий содержат методы, позволяющие связывать их со слушателями. Когда происходит соответствующее событие, источник посылает извещение всем объектам слушателя, зарегистрированным для этого.

Поскольку язык Java является объектно-ориентированным, информация о событии

инкапсулируется в объекте события (event object). В языке Java все объекты событий представляют собой объекты классов, производных от класса java.util.EventObject.

Разумеется, для каждого типа событий существует свой подкласс, например подклассы ActionEvent и WindowEvent.

Различные источники могут порождать разные виды событий. Например, кнопка может посылать объекты класса ActionEvent, а окно — объекты класса WindowEvent.

Для регистрации объекта слушателя источником события применяется следую-

щий оператор:

*объектИсточникаСобытия.*addCo6brrMeListener*(объектСлушателяСобытия)*

Например при создании кнопки:

```
ActionListener listener = ...;
Jbutton button = new JButton("Ok");
button.addActionListener(listener) ;
```

Теперь объект слушателя извещается о "наступлении события", связанного с кнопкой. Это может быть, например, простой щелчок.

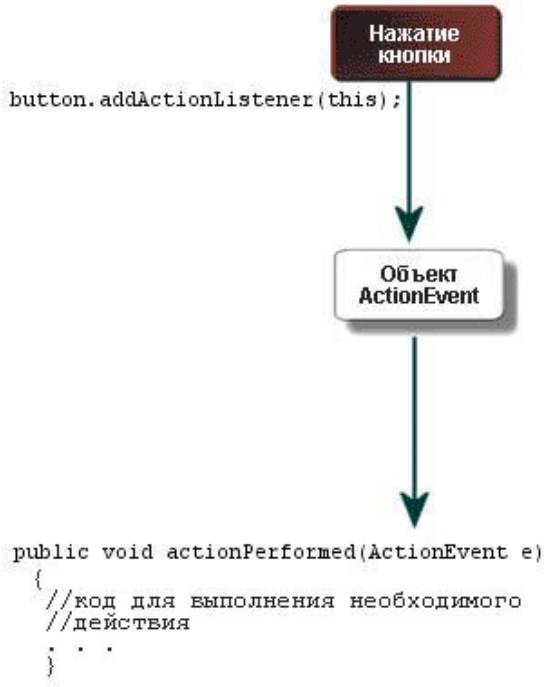
Код, подобный приведенному выше, требует, чтобы класс, которому принадлежит объект слушателя, реализовывал соответствующий интерфейс (в данном случае — ActionListener).

Для того чтобы реализовать интерфейс ActionListener, класс слушателя должен иметь метод с именем actionPerformed, получающий объект класса ActionEvent в качестве параметра.

```
class MyListener implements ActionListener
{
public void actionPerformed(ActionEvent event)
{
        ...
}
}
```

Когда пользователь, к примеру, щелкает на кнопку мыши, объект класса JButton создает объект класса ActionEvent и вызывает метод listener.actionPerformed (event), передавая ему этот объект события.

Более наочно можете увидеть это на схеме ниже:



При этом интерфейс ActionListener, который мы рассматривали выше, не ограничивается отслеживанием щелчков на кнопках.

Он используется во многих ситуациях:

• при двойном щелчке на пункте списка;

• при выборе пункта меню;

• при нажатии клавиши <ENTER> в поле ввода текста;

• по истечении заданного отрезка времени, отслеживаемого компонентом Timer.

- **Типы событий**

После того как мы разобались с механизмом обработки событий, давайте рассмотрим какие типы событий существуют в Java.

В библиотеке AWT проведено разделение событий на **низкоуровневые** (lowlevel events) и **семантические** (semantic events).

Семантические события описывают действия пользователя, например, "щелчок на кнопке". Низкоуровневые события же обеспечивают возможность таких действий. Например перестаскивание обьекта мышью, нажатие кнопки мыши (а не самой кнопки на экране)

В пакете java.awt.event предусмотрено четыре класса семантических событий:

• ActionEvent (щелчок на кнопке, выбор пункта меню, выбор пункта в списке, а также нажатие клавиши <ENTER> в поле ввода текста).

• AdjustmentEvent (перемещение бегунка на панели прокрутки).

• ItemEvent (установка флажка или выбор пункта в списке).

• TextEvent (изменение содержания текстового поля).

Кроме того, существуют семь классов низкоуровневых событий:

• ComponentEvent (изменение размера компонента, его перемещение, показ или сокрытие; этот класс является базовым для всех низкоуровневых событий).

• KeyEvent (нажатие или высвобождение клавиши).

• MouseEvent (нажатие, высвобождение клавиши мыши, а также перемещение курсора и буксировка объекта).

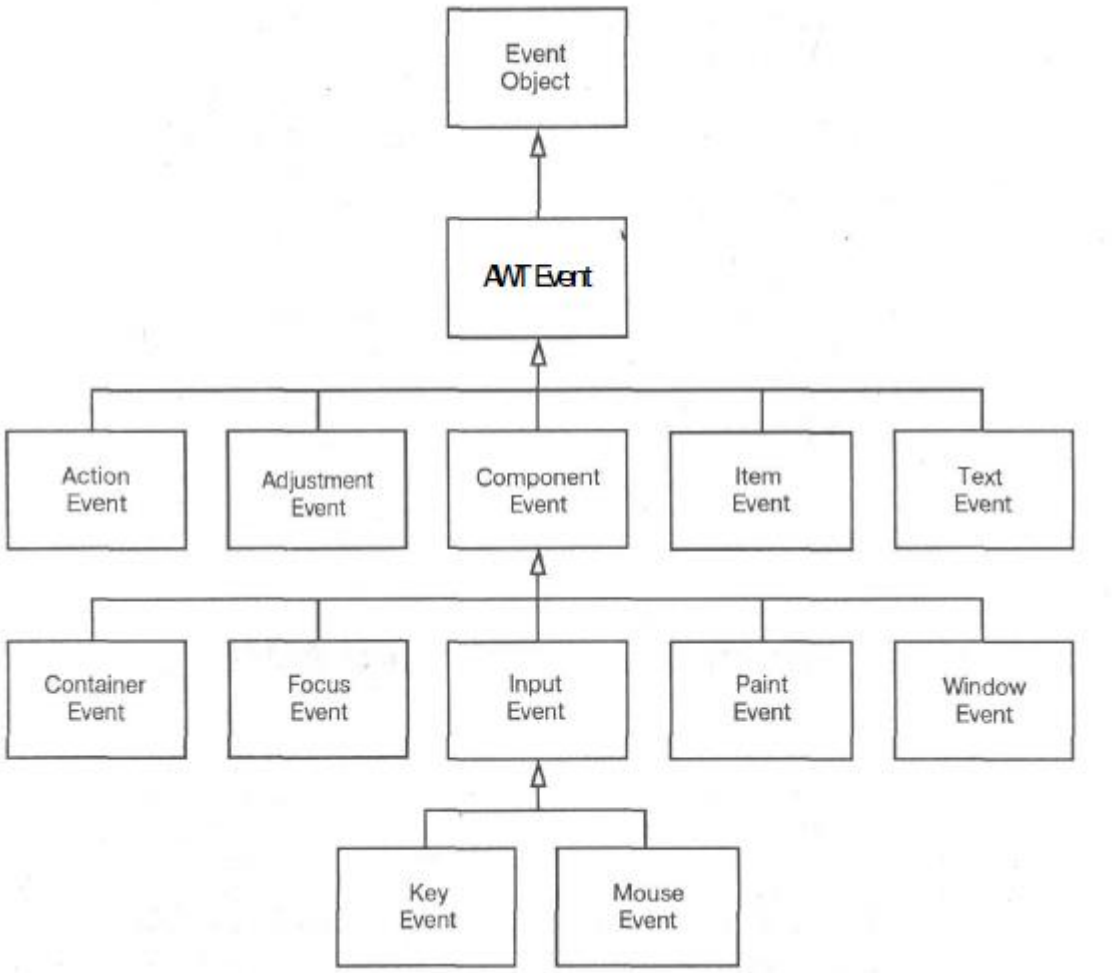
• MouseWheelEvent (вращение шарика мыши).

• FocusEvent (перемещение фокуса внимания на компонент и снятие фокуса с компонента).

• WindowEvent (активирование, деактивирование, сворачивание, разворачивание и закрытие окна).

• ContainerEvent (добавление и удаление компонента).

Ниже вы можете посмотреть диаграмму наследования классов событий из пакета AWT



- **Классы адаптеров на примере оконного события**

Прежде чем мы перейдем к различным типам обработки событий, нам нужно ознакомиться с таким понятием как адаптер. Для того чтобы, вы поняли, что такое адаптер, давайте рассмотрим с вами событие WindowEvent, который возникает при совершении каких либо действий с окном.

Приемник этого события должен быть экземпляром класса, реализующего интерфейс WindowListener. При этом, в интерфейсе WindowListener имеется семь методов.

Как обычно, любой класс, реализующий какой-нибудь интерфейс, должен реализовать все его методы.

Но иногда нам не нужны все семь методов и потому не хочется их все прописывать.

И тут на помощь нам приходят классы адаптеров, например WindowAdapter. Они реализуют все методы, причем тела их пусты. Но при этом класс адаптера можно расширить и уточнить некоторые методы.

- **MouseListener**

Как уже было указано для обработки события, возникающего при нажатии на кнопку можно использовать ActionListener, так как операции с мышью автоматически обрабатываются компонентами пользовательского интерфейса.

Но иногда возникают ситуации, когда надо прослушивать не только щелчки, а например, просто нажатие кнопки мыши или ее перемещение.

Для того, чтобы прослушивать события мыши необходимо реализовать интерфейс MouseListener

По аналогии с другими слушателями swing, MouseListener — это интерфейс, методы которого необходимо реализовать. Всего их 5.

Начнем рассмотрение с **mouseEntered.** Данный метод будет вызываться системой у слушателя каждый раз, когда курсор мыши будет оказываться над компонентом. В противоположность этому методу — mouseExited. Он срабатывает, когда убираем курсор мыши с компонента

Пример: у нас есть компонент. Мы добавили к нему слушателя MouseListener. Начинаем водить мышкой. Как только «залезли» курсором на компонент — вызвался mouseEntered, уводим курсор с компонента — вызвался mouseExited.

Что ж , идем дальше. Каждый раз при нажатии одной из кнопок мыши будет срабатывать mousePressed. Навели на компонент, зажали кнопку — система вызвала mousePressed. Отпускаем кнопку — mouseReleased

И последний метод mouseClicked – щелчок кнопки мыши.

Также у мыши еще есть интерфейс MouseMotionListener, который отслеживает ее движение. Он содержит два метода: mouseMoved(), mouseDragged().

MouseMoved() отслеживает просто перемещение мыши, а mouseDragged() ее перемещение при нажатой кнопке мыши.

Также предусмотрен **MouseAdapter** и мы соответственно можем переопределить только нужный нам метод.

- **KeyListener**

Работая с приложением, которое имеет графический интерфейс, пользователь прибегает к помощи не только мыши, но и клавиатуры. Давайте посмотрим, что необходимо сделать, чтобы иметь возможность слушать события клавиатуры. Для этого рассмотрим интерфейс KeyListener из пакета java.awt.event.

KeyListener имеет три метода: keyTyped, keyPressed и keyReleased

Метод keyTyped вызывается системой каждый раз, когда пользователь нажимает на клавиатуре клавиши символы Unicode.

Метод keyPressed вызывается системой в случае нажатия любой клавиши на клавиатуре

Метод keyReleased вызывается при отпускании любой клавиши на клавиатуре.

Чтобы добавить слушателя KeyListener к интересуемому компоненту для прослушивания событий клавиатуры, используется метод addKeyListener.

Также предусмотрен **KeyAdapter**, который содержит все три метода но с пустыми методами keyTyped, keyPressed и keyReleased и мы соответсвенно можем переопределить только нужный нам метод.

Когда клавиши нажата, то узнать ее код можно с помощью метода getKeyCode. Класс KeyEvent содержит большой набор констант. Каждая константа содержит код соответствующей клавиши.

Например KeyEvent. VK_ENTER или KeyEvent.VK_F.

Стоит сказать, что события от клавиатуры будут генерироваться системой только тогда, когда компонент, который мы слушаем, находится в фокусе.

Здесь происходит обработки нажатия клавиш компонентом JPanel. Вообще по умолчанию JPanel не должен получать фокуса, однако это можно зделать, если очень захотеть при помощи метода setFocusable и передать этому методу true в качестве параметра.

- **TextListener**

Иногда возникает ситуация, когда требуется обработать ввод пользователя. Событие TextEvent происходит только по одной причине — изменению текста.

Соответствующий интерфейс имеет только один метод:

```
public interface TextListener extends EventListener{
public void textValueChanged(TextEvent e) ;
}
```

От аргумента e этого метода можно получить ссылку на объект-источник события методом getSource, унаследованным от класса EventObject, например, так:

```
TextComponent tc = (TextComponent)e.getSource();
```

После чего извлекаем текст:

```
String s = tc.getText() ;
```

- **WindowListener**

Мы уже с вами немного рассматривали это событие. WindowEvent служит для действий, связанных с окном и может произойти по семи причинам:

- окно открылось — идентификатор WINDOW_OPENED;
- окно закрылось — идентификатор WINDOW_CLOSED;
- попытка закрытия окна — идентификатор WINDOW_CLOSING;
- окно получило фокус — идентификатор WINDOW_ACTIVATED;
- окно потеряло фокус — идентификатор WINDOW_DEACTIVATED;
- окно свернулось в ярлык — идентификатор WINDOW_ICONIFIED;
- окно развернулось — идентификатор WINDOW_DEICONIFIED.

Соответствующий интерфейс содержит семь методов:

```
public interface WindowListener extends EventListener {
public void windowOpened(WindowEvent e);
public void windowClosing(WindowEvent e);
public void windowClosed(WindowEvent e);
public void windowlconified(WindowEvent e);
public void windowDeiconified(WindowEvent e);
public void windowActivated(WindowEvent e);
public void windowDeactivated(WindowEvent e); }
```

- **Событие ComponentEvent**

Данное событие происходит в компоненте, причем по четырем причинам:

- компонент перемещается — идентификатор COMPONENT_MOVED;
- компонент меняет размер — идентификатор COMPONENT_RESIZED;
- компонент убран с экрана — идентификатор COMPONENT_HIDDEN;
- компонент появился на экране — идентификатор COMPONENT_SHOWN.

Соответствующий интерфейс содержит описания четырех методов:

```
public interface ComponentListener extends EventListener{
public void componentResized(ComponentEvent e);
public void componentMoved(Comp©nentEvent e);
public void componentShown(ComponentEvent e);
public void componentHidden(ComponentEvent e);
```

}

- **Событие ContainerEvent**

Это событие происходит в контейнере, причем по двум причинам:

- в контейнер добавлен компонент — идентификатор COMPONENT_ADDED;
- из контейнера удален компонент — идентификатор COMPONENT_REMOVED.

Этим причинам соответствуют методы интерфейса:

```
public interface ContainerListener extends EventListener{
public void componentAdded(ContainerEvent e) ;
public void componentRemoved(ContainerEvent e);
}
```

- **Событие ItemEvent**

Это событие возникает при выборе или отказе от выбора элемента в списке List, choice или флажка checkbox и отмечается идентификаторомITEM_STATE_CHANGED.

Соответствующий интерфейс очень прост:

```
public interface ItemListener extends EventListener{
void itemStateChanged(ItemEvent e);
}
```

- **Событие AdjustmentEvent**

Это событие возникает для полосы прокрутки ScrollBar при всяком изменении ее бегунка и отмечается идентификатором ADJUSTMENT_VALUE_CHANGED.

Соответствующий интерфейс описывает один метод:

```
public interface AdjustmentListener extends EventListener{
public void adjustmentValueChanged(AdjustmentEvent e);
}
```

Контрольные вопросы

1. Опишите принцип функционирования апплета.
2. Как осуществляется передача параметров апплету.
3. Как производится загрузка и вывод графических изображений.
4. Что такое апплеты двойного назначения?
5. Какие директивы JSP вы знаете?
6. Перечислите элементы JSP скриптов.
7. Как применяются атрибуты JSP страницы и границы видимости?
8. Как работают обработчики событий от мыши?
9. Опишите апплет, обрабатывающий события
10. Перечислите основные методы класса Applet.
11. Опишите различия между Java апплетом и Java приложением.
12. В чем сущность переопределения методов апплета.

Практическое занятие №8 Изучение методов построения WEB сервисов

SOAP - протокол обмена структурированными сообщениями в распределённой вычислительной среде. Поддерживается консорциумом W3C (http://www.w3.org/TR/soap/).

Протокол SOAP создан в 1998 году командой разработчиков под руководством Дейва Винера (Dave Winer), работавшей в корпорации Microsoft и фирме Userland, но затем передан в консорциум W3C.

Последняя версия стандарта на сегодняшний день - SOAP 1.2. В версии 1.1 SOAP расшифровывался как Simple Object Access Protocol — простой протокол доступа к объектам. Это название отражало его первоначальное назначение — обращаться к методам удаленных объектов. Сейчас назначение SOAP изменилось, поэтому разные разработчики предлагали свои варианты расшифровки. Поэтому в версии 1.2 аббревиатуру решили никак не расшифровывать.

Протокол SOAP не различает вызов процедуры и ответ на него, а просто определяет формат послания (message) в виде документа XML. Послание может содержать вызов процедуры, ответ на него, запрос на выполнение каких-то других действий или просто текст. Спецификацию SOAP не интересует содержимое послания, она задает только его оформление.

SOAP основан на языке XML и расширяет некоторый протокол прикладного уровня — HTTP, FTP, SMTP и т.д. Как правило чаще всего используется HTTP. Вместо использования HTTP для запроса HTML-страницы, которая будет показана в браузере, SOAP отправляет посредством HTTP-запроса XML-сообщение и получает результат в HTTP-отклике. Для правильной обработки XML-сообщения процесс-«слушатель» HTTP (напр.

Apache или Microsoft IIS) должен предоставить SOAP-процессор, или, другими словами, должен иметь возможность обрабатывать XML.

SOAP является самой главной частью технологии Web-сервисов. Он осуществляет перенос данных по сети из одного места в другое.

SOAP обеспечивает доставку данных веб-сервисов. Он позволяет отправителю и получателю XML-документов поддерживать общий протокол передачи данных, что обеспечивает эффективность сетевой связи.

SOAP – это базовая однонаправленная модель соединения, обеспечивающая согласованную передачу сообщения от отправителя к получателю, потенциально допускающая наличие посредников, которые могут обрабатывать часть сообщения или добавлять к нему дополнительные элементы. Спецификация SOAP содержит соглашения по преобразованию однонаправленного обмена сообщениями в соответствии с принципом «запрос/ответ», а также определяет как осуществлять передачу всего XML-документа.

Как видно из рис.1 SOAP предназначен для поддержания независимого абстрактного протокола связи, обеспечивающего коммуникацию двух и более приложений, сайтов, предприятий и т.п., реализованных на разных технологиях и аппаратных средств.

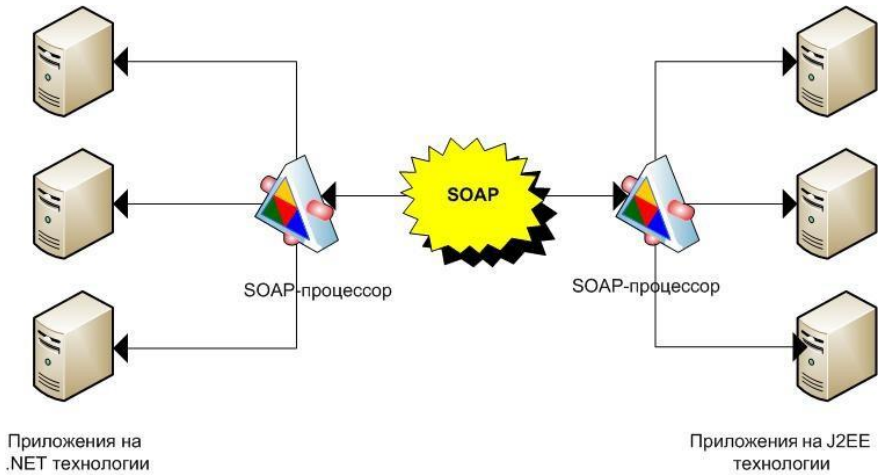


Рисунок 1.

Общая структура SOAP сообщения

SOAP-сообщение представляет собой XML-документ; сообщение состоит из трех основных элементов: конверт (SOAP Envelope), заголовок (SOAP Header) и тело (SOAP Body).

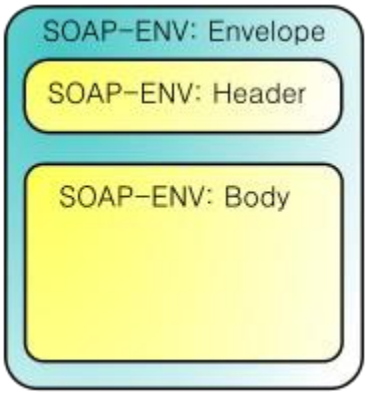


Рисунок 2. Структура SOAP сообщения

Пример SOAP сообщения:

```
<SOAP-ENV:Envelope xmlns:SOAP-
    ENV="http://www.w3.org/2003/05/soap-envelope"
    xmlns:t="www.example.com">
    <SOAP-ENV:Header>
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
        <t:CurrentDate>
            <Year>2011</Year>
            <Month>February</Month>
            <Day>12</Day>
            <Time>18:02:00</Time>
        </t:CurrentDate>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Конверт (SOAP Envelope)

Является самым «верхним» элементом SOAP сообщения. Содержит корневой элемент XML-документа. Описывается с помощью элемента `Envelope` с обязательным пространством имен http://www.w3.org/2003/05/soap-envelope для версии 1.2 и http://schemas.xmlsoap.org/soap/ для версии 1.1.

У элемента `Envelope` могут быть атрибуты `xmlns`, определяющие пространства имен, и другие атрибуты, снабженные префиксами.

`Envelope` может иметь необязательный дочерний элемент `Header` с тем же пространством имен — заголовок. Если этот элемент присутствует, то он должен быть первым прямым дочерним элементом конверта.

Следующий дочерний элемент конверта должен иметь имя `Body` и то же самое пространство имен - тело. Это обязательный элемент и он должен быть вторым прямым дочерним элементом конверта, если есть заголовок, или первым — если заголовка нет.

Версия 1.1 позволяла после тела сообщения записывать произвольные элементы, снабженные префиксами. Версия 1.2 это запрещает.

Элементы `Header` и `Body` могут содержать элементы из различных пространств

имен.

Конверт изменяется от версии к версии. SOAP-процессоры, совместимые с версией 1.1, при получении сообщения, содержащего конверт с пространством имен версии 1.2, будут генерировать сообщение об ошибке. Аналогично для SOAP-процессоров, совместимых с версией 1.2. Ошибка — VersionMismatch.

Заголовок SOAP (SOAP Header)

Первый прямой дочерний элемент конверта. Не обязательный. Заголовок кроме атрибутов `xmlns` может содержать 0 или более стандартных атрибутов:

- `encodingStyle`
- `actor` (или `role` для версии 1.2)
- `mustUnderstand`
- `relay`

Атрибут encodingStyle

В SOAP-сообщениях могут передаваться данные различных типов (числа, даты, массивы, строки и т.п.). Определение этих типов данных выполняется в схемах XML (обычно
— XSD). Типы, определенные в схеме, заносятся в пространство имен, идентификатор которого служит значением атрибута `encodingStyle`. Атрибут `encodingStyle` может появиться в любом элементе SOAP-сообщения, но версия SOAP 1.2 запрещает его появление в корневом элементе `Envelope`. Указанное атрибутом `encodingStyle` пространство имен будет известно в том элементе, в котором записан атрибут, и во всех вложенных в него элементах. Какие-то из вложенных элементов могут изменить пространство имен своим атрибутом `encodingStyle`.

Стандартное пространство имен, в котором расположены имена типов данных SOAP 1.1, называется `http://schemas.xmlsoap.org/soap/encoding/`. У версии 1.2 — `http://www.w3.org/2003/05/soap-encoding`. Идентификатор того или иного пространства имен, в котором определены типы данных, обычно получает префикс enc или SOAP-ENC.

Атрибут `actor`

Тип данных URI. Задает адрес конкретного SOAP-сервера, которому предназначено сообщение.

SOAP-сообщение может пройти через несколько SOAP-серверов или через несколько приложений на одном сервере. Эти приложения выполняют предварительную обработку блоков заголовка послания и передают его друг другу. Все эти серверы и/или приложения называются SOAP-узлами (SOAP nodes). Спецификация SOAP не определяет правила прохождения послания по цепочке серверов. Для этого разрабатываются другие протоколы, например, Microsoft WS-Routing.

Атрибут actor задает целевой SOAP-узел — тот, который расположен в конце цепочки и будет обрабатывать заголовок полностью. Значение `http://schemas.xmlsoap.org/soap/actor/next` атрибута actor показывает, что обрабатывать заголовок будет первый же сервер, получивший его. Атрибут actor может встречаться в отдельных блоках заголовка, указывая узел-обработчик этого блока. После обработки блок удаляется из SOAP-сообщения.

В версии 1.2 атрибут actor заменен атрибутом `role`, потому что в этой версии SOAP каждый узел играет одну или несколько ролей. Спецификация пока определяет три роли SOAP-узла:

- Роль `http://www.w3.org/2003/05/soap-envelope/role/ultimateReceiver` играет конечный, целевой узел, который будет обрабатывать заголовок.
- Роль `http://www.w3.org/2003/05/soap-envelope/role/next` играет промежуточный или целевой узел. Такой узел может играть и другие, дополнительные роли.

Роль `http://www.w3.org/2003/05/soap-envelope/role/none` не должен играть ни один SOAP-узел.

Распределенные приложения, исходя из своих нужд, могут добавить к этим ролям другие роли, например, ввести промежуточный сервер, проверяющий цифровую подпись и определить для него эту роль какой-нибудь строкой URI.

Значением атрибута role может быть любая строка URI, показывающая роль узла, которому предназначен данный блок заголовка. Значением по умолчанию для этого атрибута служит пустое значение, то есть, просто пара кавычек, или строка URI `http://www.w3.org/2003/05/soap-envelope/role/ultimateReceiver`.

Значение атрибута role показывает, что блок должен быть обработан узлом, играющим роль, определенную такой же строкой.

Атрибут mustUnderstand

Тип данных — boolean. По умолчанию 0. Если значение равно 1, то SOAP-узел при обработке элемента обязательно должен учитывать его синтаксис, определенный в схеме документа, или совсем не обрабатывать сообщение. Это повышает точность обработки сообщения.

В версии SOAP 1.2 вместо цифр нужно писать true или false.

Атрибут `relay`

Тип данных — boolean. Показывает, что заголовочный блок, адресованный SOAP- посреднику, должен быть передан дальше, если он *не* был обработан. Необходимо отметить, что, если заголовочный блок обработан, правила обработки SOAP требуют, чтобы он был удален из уходящего сообщения. По умолчанию, необработанный заголовочный блок,

предназначенный роли, которую исполняет SOAP-посредником, должен быть удален перед отправкой сообщения.

Все прямые дочерние элементы заголовка называются блоками заголовка (в версии - статьями). Блоки заголовка используются для расширения сообщений децентрализованным способом путем добавления таких функций как аутентификация, администрирование транзакций и т. п. Их имена обязательно должны помечаться префиками. В блоках заголовка могут быть атрибуты role, actor и mustUnderstand. Действие этих атрибутов относится только к данному блоку. Это позволяет обрабатывать отдельные блоки заголовка промежуточными SOAP-узлами, чья роль совпадает с ролью, указанной атрибутом role. Ниже дан пример такого блока:

```
<env:Header>
    <t:Transaction
        xmlns:t="http://example.com/tra
        nsaction"
        env:role="http://www.w3.org/200
        3/05/soap-
envelope/role/ultimateReceiver"
        env:mustUnderst
        and="true"> 5
    </t:Transaction>
</env:Header>
```

Элементы, вложенные в блоки заголовка, уже не называются блоками. Они не могут содержать атрибуты `role`, `actor` и `mustUnderstand`.

Тело SOAP (SOAP Body)

Элемент `Body` обязательно записывается сразу за элементом `Header`, если он есть в сообщении, или первым в SOAP-сообщении, если заголовок отсутствует. В элемент `Body` можно вложить произвольные элементы, спецификация никак не определяет их структуру. Определен

только один стандартный элемент, который может быть в теле сообщения - Fault, содержащий сообщение об ошибке.

Обработка ошибок в SOAP-сообщениях

Если SOAP-сервер, обрабатывая поступившее SOAP-сообщение, обнаружит ошибку, то он прекратит обработку и отправит клиенту SOAP-сообщение, содержащее один элемент Fault с сообщением об ошибке.

В версии 1.1 элемент Fault имел 4 дочерних элемента:

- код ошибки faultcode — предназначено для программы, обрабатывающей ошибки;
- описание ошибки faultstring – словесное описание типа ошибки, предназначено для человека;
- место обнаружения ошибки faultactor – адрес URI сервера, заметившего ошибку. Промежуточные SOAP-узлы обязательно записывают этот элемент, целевой SOAP- сервер не обязан это делать;
- Детали ошибки detail — описывают ошибки, встреченные в теле Body послания, но не в его заголовке. Если при обработке тела ошибки не обнаружены, то этот элемент отсутствует.

Пример сообщения об ошибке:

```
<env:Envelope
xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
    <env:Body>
        <env:Fault>
            <faultcode>env:MustUnderstand</faultcode>
            <faultstring>SOAP Must Understand
            Error</faultstring>
        </env:Fault>
    </env:Body>
</env:Envelope>
```

В версии SOAP 1.2 содержание элемента `Fault` изменилось . Как описано в пространстве имен `http://www.w3.org/2003/05/soap-envelope`, в него входят два обязательных элемента и три необязательных элемента.

Обязательные элементы:

- Код ошибки `code`. Он содержит обязательный вложенный элемент `value` с кодом ошибки и необязательный вложенный элемент `subcode`, также содержащий элемент `value` с уточняющим кодом ошибки и элемент `subcode`, и далее все повторяется рекурсивно.
- Причина ошибки `Reason`. Содержит необязательный атрибут `xml:lang`, указывающий язык сообщения, и произвольное число вложенных элементов с описанием ошибки.

Необязательные элементы:

- `Node` — адрес URI промежуточного SOAP-узла, заметившего ошибку.
- `Role` — роль SOAP-узла, заметившего ошибку.
- `Detail` — описание ошибки, замеченной при обработке тела `Body` послания, но не его заголовка.

В следующем примере показана ошибка, возникающая при попытке выполнения процедуры — имена аргументов процедуры неправильно записаны.

```
<?xml version='1.0' ?>
<env:Envelope
    xmlns:env="http://www.w3.org/2003/05/so
    ap-envelope"
    xmlns:rpc='http://www.w3.org/2002/06/so
    ap-rpc1>
```

```
    <env:Body>
        <env:Fault>
            <env:Code>
                <env:Value>env:Sender</env:Value>
                <env:Subcode>
                    <env:Value>rpc:BadArguments</env:
                    Value>
                </env:Subcode>
            </env:Code>
            <env:Reason>Processing Error</env:Reason>
            <env:Detail>
                <e:myfaul
tdetails
xmlns:e="http://www.exampl
e.org/faults">
                    <message>Name does not
                    match</message>
                    <errorcode>999</errorcode>
                </e:myfaultdetails>
            </env:Detail>
        </env:Fault>
    </env:Body>
</env:Envelope>
```

Контрольные вопросы

1. В чем сущность технологии Web-сервиса?
2. В чём сущность технологии JMS?
3. Опишите модели обмена сообщения-ми: «точка - точка» и «издатель-подписчик».
4. Что такое XML приложение?
5. Что лежит в основе Web-сервисов?
6. Перечислите свойства Web-сервисов.
7. Типы взаимодействия Web-сервисов.
8. В чем сущность протокола SOAP?

9. Что такое WSDL?

Рекомендуемая литература

1. Основная литература

Код	Авторы, составители	Заглавие	Издательство, год	Кол.
Л1.1	Гуриков С. Р.	Интернет-технологии: Учебное пособие/ -, 500 экз	М.: Форум, НИЦ ИНФРА-М, 2015. - 184 с.: 70x100 1/16. - (Высшее образование: Бакалавриат) ISBN 978-5-00091-001-	Э1
Л1.2	Т.И. Немцова, Т.В. Казанкова, А.В. Шнякин / под ред. Л.Г. Гагариной	Компьютерная графика и web-дизайн : учеб. пособие /. —. + Доп. материалы	М. : ИД «ФОРУМ» : ИНФРА-М, 2019. — 400 с	Э2
Л1.3	Бенкен Е.С.	PHP, MySQL, XML: программирование для Интернета. - 3-е изд., перераб. и доп. -	СПб:БХВ-Петербург, 2011. - 304 с. ISBN 978-5-9775-0724-0	Э3
Л1.4	Будилов В.А.	Интернет-программирование на Java: Пособие / -	СПб:БХВ-Петербург, 2014. - 698 с. ISBN 978-5-9775-1931-	Э4
Л1.5	Соколова В.В.	Разработка мобильных приложений: Учебное пособие / -	Томск:Изд-во Томского политех. университета, 2014. - 176 с.: ISBN 978-5-4387-0369-3	Э8

2 Дополнительная литература

Код	Авторы, составители	Заглавие	Издательство, год	Кол.
Л2.1	Зиангирова Л.Ф.	Сетевые технологии учебно-методическое пособие	Саратов: Вузовское образование, 2019.— 100 с	Э5
Л2.2	Будилов В.А.	Основы программирования для Интернета: Пособие / -	СПб:БХВ-Петербург, 2014. - 733 с. ISBN 978-5-9775-1917-	Э6
Л2.3	Кисленко Н.П.	Интернет-программирование на PHP учебное пособие	Новосибирск: Новосибирский государственный архитектурно-строительный	Э7

			университет (Сибстрин),	
3 Методическое обеспечение для самостоятельной работы обучающихся				
Код	**Авторы, составители**	**Заглавие**	**Издательство, год**	**Кол.**
Л4.1	П.Б. Храмцов [и др.].	Основы Web-технологий учебное пособие	Москва, Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2019.— 375 с.—	Э8
Л4.2	Семенов Ю.А.	Протоколы и алгоритмы маршрутизации в «Интернет»	М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 998 с.—	Э9
Л4.3	Дубаков А.А.	Сетевое программирование учебное пособие	СПб.: Университет ИТМО, 2014.— 249 с.—	Э10
Л4.3	Зиангирова Л.Ф.	Технологии облачных вычислений учебное пособие	Саратов: Вузовское образование, 2016.— 300 с.	Э11
Электронные образовательные ресурсы				
Э1	http://znanium.com/bookread2.php?book=488074			
Э2	http://znanium.com/bookread2.php?book=894969			
Э3	http://znanium.com/bookread2.php?book=350558			
Э4	http://znanium.com/bookread2.php?book=940239			
Э5	http://www.iprbookshop.ru/62065.html.— ЭБС «IPRbooks»			
Э6	http://znanium.com/bookread2.php?book=940218			
Э7	http://www.iprbookshop.ru/68769.html.— ЭБС «IPRbooks»			
Э8	http://www.iprbookshop.ru/67384.html.— ЭБС «IPRbooks»			
Э9	http://www.iprbookshop.ru/62826.html.— ЭБС «IPRbooks			
Э10	http://www.iprbookshop.ru/68118.html.— ЭБС «IPRbooks»			
Э11	http://www.iprbookshop.ru/41948.html.— ЭБС «IPRbooks»			
Программное обеспечение				
П.1	MS Windows			
П.2	Система визуального программирования Lazarus			
П.3	Пакет программ для проведения тестирования по изученным темам			